

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

Mô phỏng hệ thống truyền thông (Simulation of Communication Systems)

Bộ môn Tín hiệu và Hệ thống

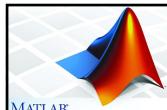
1

Giới thiệu môn học

- **Thời lượng môn học:**
 - 3 ĐVHT (20LT + 4BT + 6TH + 15 Tự Học)
- **Mục tiêu:**
 - Kiến thức: Cung cấp cho người học những khái niệm và kiến thức cơ bản về mô hình hóa và mô phỏng. Nội dung của môn học sẽ tập trung vào phương pháp luận cũng như công cụ mô phỏng hệ thống truyền thông làm cơ sở cho các môn học chuyên sâu khác và hỗ trợ cho làm đồ án tốt nghiệp.
 - Kỹ năng: Rèn cho sinh viên có kỹ năng sử dụng bộ công cụ chương trình MATLAB và Simulink, và các phương pháp cơ bản áp dụng cho việc mô phỏng các hệ thống truyền thông.
- **Đánh giá:**

<ul style="list-style-type: none"> – Tham gia học tập trên lớp: – Thực hành-Thí nghiệm – Bài tập/Thảo luận: – Kiểm tra giữa kỳ: – Kiểm tra cuối kỳ: 	10% 10% 20 % 10% 50%
--	----------------------------------

2

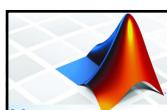


Giới thiệu môn học

- Nội dung:
 - Chương 1: **Tổng quan về kỹ thuật mô phỏng**
 - Chương 2: **Giới thiệu về MATLAB**
 - Chương 3: **Giới thiệu về Simulink**
 - Chương 4: **Mô phỏng tín hiệu và quá trình thu phát**
 - Chương 5: **Mô phỏng kênh thông tin**
 - Chương 6: **Ước tính tham số và đánh giá hiệu năng**

 3

3

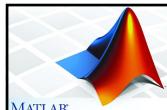


Giới thiệu môn học

- Tài liệu tham khảo:
 - [1] Michel C. Jeruchim, Philip Balaban, *Simulation of Communication Systems: Modeling, Methodology and Techniques*, 2nd ed., Kluwer Academic/Plenum Publishers, 2000.
 - [2] Nguyễn Viết Đảm, *Mô phỏng hệ thống viễn thông và ứng dụng MATLAB*, NXB Bưu Điện, 2007.
 - [3] J. G. Proakis, M. Salehi, G. Bauch, *Contemporary Communication Systems Using MATLAB and Simulink*, 3rd ed., Cengage Learning, 2012.
 - [4] O. Beucher, M. Weeks, *Introduction to MATLAB and Simulink: A Project Approach*, 3rd ed., Infinity Science Press, 2008.
 - [5] Mathworks Inc., *MATLAB and Simulink Student Version: Getting Started With MATLAB*, 2007.
 - [6] Steven C. Chapra, R. P. Canale, *Numerical Methods for Engineers*, 6th ed., McGraw-Hill, 2010.
 - [7] Dennis Silages, *Digital Communication Systems using MATLAB and Simulink*, Bookstand Publishing, 2009.
 - [8] K. C. Raveendranathan, *Communication Systems Modeling and Simulation using MATLAB and Simulink*, Universities Press, 2011.
 - [9] Mohsen Guizani, Ammar Rayes, Bilal Khan, Ala Al-Fuqaha, *Network Modeling and Simulation: A Practical Perspective*, Wiley, 2010.

 4

4

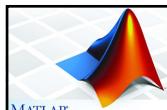


Tổng quan về kỹ thuật mô phỏng

- Giới thiệu chung
- Phương pháp luận mô phỏng
- Các khái niệm cơ bản về mô hình hóa
- Kỹ thuật đánh giá hiệu năng
- Các nguồn lối trong mô phỏng
- Vai trò mô phỏng trong thiết kế hệ thống truyền thông

 5

5

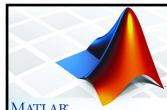


Giới thiệu chung

- Độ phức tạp của hệ thống truyền thông hiện đại:
 - Ngày càng tăng lên
 - Tính phức tạp do:
 - Cấu trúc phức tạp của hệ thống
 - Môi trường được triển khai
 - Yêu cầu về đồng bộ do hoạt động tại tốc độ cao
 - động lực thúc đẩy sử dụng mô phỏng (simulation)
- Sự phát triển của máy tính số
 - Khả năng xử lý, giá thành, độ thân thiện, ...
- Ứng dụng mô phỏng
 - Giúp hiểu biết sâu cù xử của hệ thống
 - Cho phép triển khai thí nghiệm tương tự như hệ thống thực → giảm thiểu chi phí và thời gian cho việc thiết kế hệ thống

 6

6



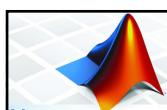
Phương pháp luận

- Bài toán mô phỏng: gồm 4 bước cơ bản
 - Ánh xạ bài toán đã cho thành mô hình mô phỏng
 - Phân giải bài toán tổng thể thành một tập các bài toán nhỏ hơn
 - Lựa chọn các kỹ thuật mô hình hóa, mô phỏng, ước tính phù hợp và áp dụng chúng để giải quyết các bài toán nhỏ của chúng
 - Kết hợp các kết quả của các bài toán con → xử lý tạo ra nghiệm cho bài toán tổng thể.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

7

7



Phương pháp luận

- Gần đúng bài toán → dễ dàng cho phân tích:
 - Phân tích
 - Mô phỏng
- Phân tích
 - Tính toán một số đặc trưng cho một đại lượng quan tâm
- Mô phỏng
 - Sao chép hệ thống quan tâm: xử lý một đại lượng động → giám sát hệ thống tại các điểm khác nhau
 - Sử dụng mô hình thực
 - Có thể thay đổi mô tả của bất kỳ một phần tử trong hệ thống (tính module)
- Hệ thống thông tin thực:
 - Quá phức tạp để đặc trưng và mô phỏng → Đơn giản hóa một số mặt của bài toán (Giảm độ phức tạp) → dễ dàng hơn cho việc tính toán

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

8

8



Phương pháp luận

- Ví dụ:
 - Dạng sóng đầu ra V_t của hệ thống: $V_t = g(\Omega)$
 - g – đặc tính truyền đạt hệ thống; $\Omega = (z_1, z_2, \dots, z_k)$ – tập các quá trình đầu vào (rồi rạc thời gian)
 - Chức năng mô phỏng:
 - Tạo ra chuỗi giá trị $\{V_t\}$ tại $t = kT_s$, $k = \pm 1, \pm 2, \dots$; T_s – chu kỳ lấy mẫu
 - Chuỗi được xử lý → thu được đại lượng hiệu năng hoặc thông tin phù hợp
 - Thí nghiệm điều kiện:
 - Tạo ra $V_t = g(\Omega')$ với $\Omega' = (z_1, \dots, z_k, \dots, z_{k+1}=\zeta_{k+1}, \dots, z_k=\zeta_k)$
 - k quá trình đầu tiên được mô phỏng, còn lại được giữ tại giá trị cố định
 - Thí nghiệm lặp lại cho một tập các điều kiện
 - Thí nghiệm mô phỏng:
 - Tạo ra $V_t = g'(\Omega')$, g' - đặc tính truyền đạt hệ thống được đơn giản hóa

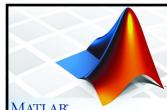
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

9

9



10

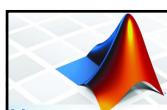


Phương pháp luận

- Các mặt ảnh hưởng:
 - Lý thuyết truyền thông:
 - Cấu trúc hệ thống, hoạt động của các phân hệ (bộ điều chế, bộ cân bằng, ...)
 - Xử lý tín hiệu số:
 - Lấy mẫu, kỹ thuật khai triển tín hiệu, lọc ...
 - Phương pháp số:
 - Kỹ thuật tích phân, nội suy, tính gần đúng ...
 - Lý thuyết xác suất:
 - Biên ngẫu nhiên, hàm mật độ xác suất, ...
 - Lý thuyết số:
 - Chuỗi số, chuỗi ngẫu nhiên, ...

11

11

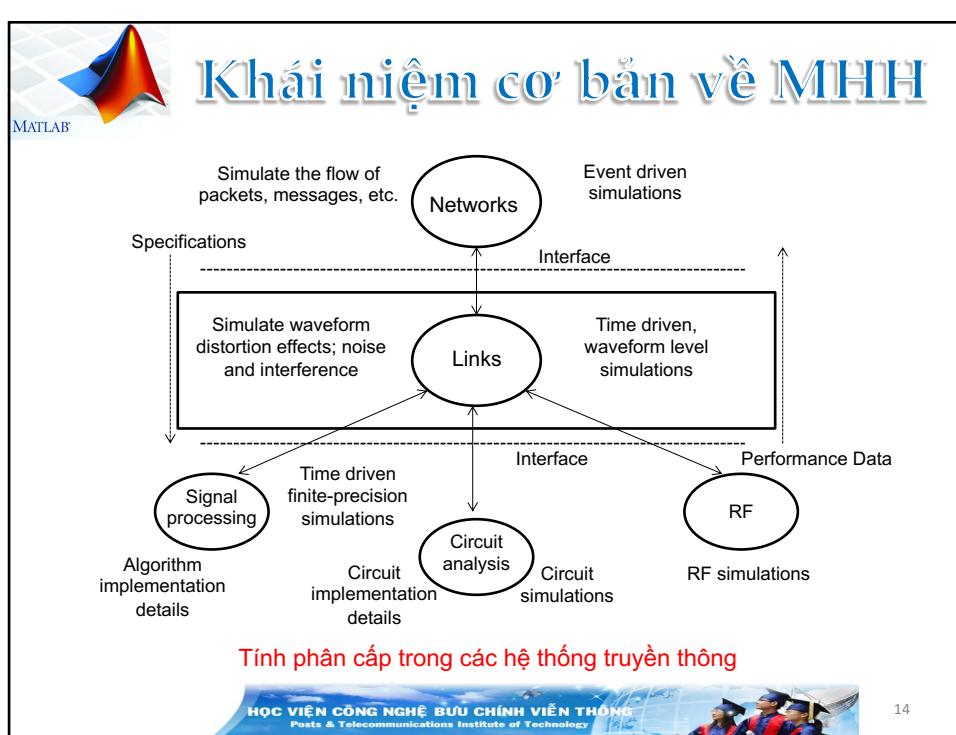
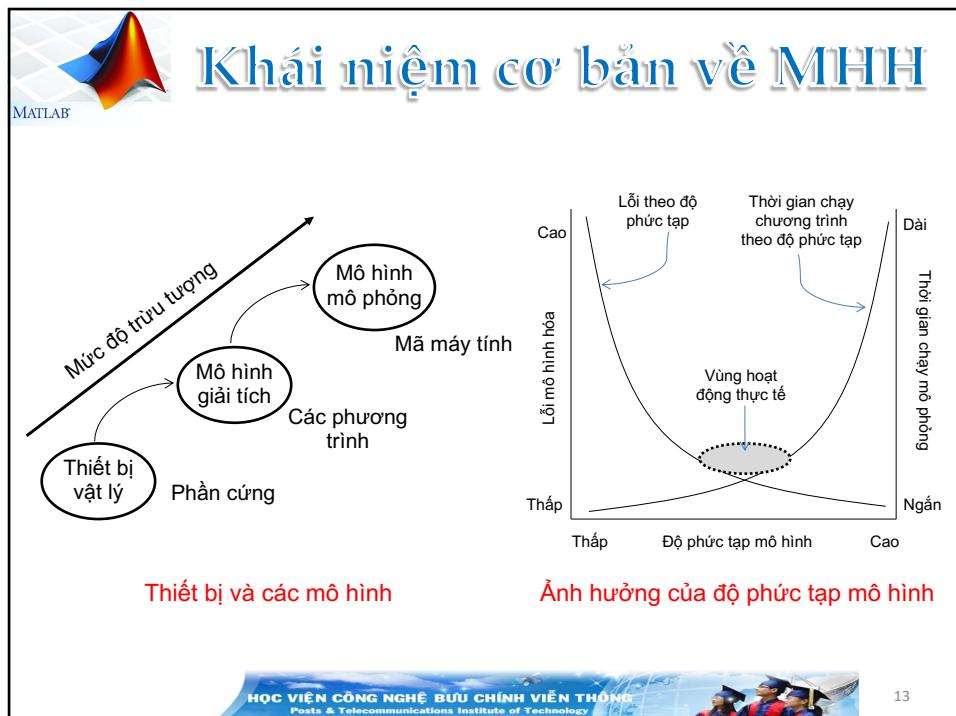


Phương pháp luận

- Các mặt ảnh hưởng:
 - Khoa học máy tính:
 - Kỹ thuật lập trình, đồ họa, ...
 - Lý thuyết ước tính:
 - Ước tính các tham số kết hợp thống kê và xử lý tín hiệu
 - Lý thuyết quá trình ngẫu nhiên:
 - Hàm phân bố, hàm tương quan, ...
 - Lý thuyết hệ thống:
 - Quan hệ vào/ra, đáp ứng xung, hàm truyền đạt ...

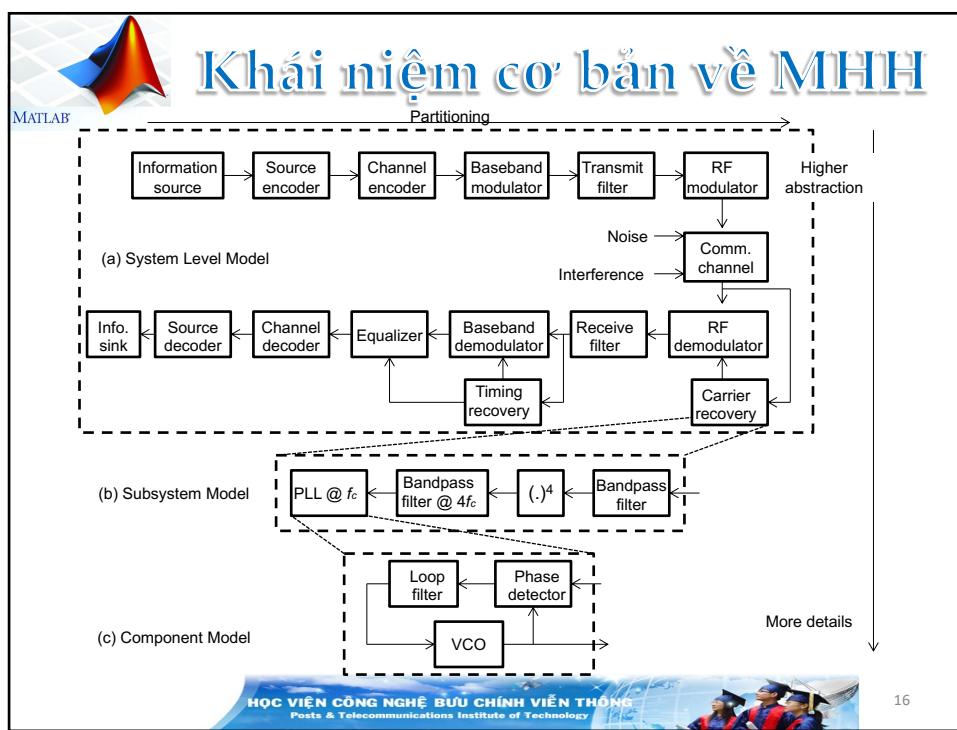
12

12

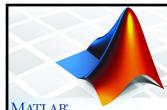




15



16



Khái niệm cơ bản về MHH

- Mô hình hóa hệ thống:
 - Hệ thống: mức cao nhất của mô tả, đặc trưng bởi sơ đồ khối các phân hệ
 - Vấn đề mô hình hóa hệ thống: vấn đề cấu hình
 - Sơ đồ khối mô phỏng càng sát với hệ thống thực → Mô hình hệ thống càng chính xác
 - Giảm mức độ phức tạp mô hình hóa → sử dụng tập con các khối ở cùng mức phân cấp
- Mô hình hóa thành phần linh kiện
 - Linh kiện: một khối tại mức phân hệ chứa những đặc điểm mà nhà thiết kế hệ thống mong muốn
 - Kiểu mô tả: một phương trình, một tập phương trình, một thuật toán, hoặc một lookup table
 - Mô hình phân hệ tốt: có các tham số đầu vào có thể biến đổi → phản ánh cùi xử thực của linh kiện

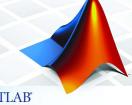
17



Khái niệm cơ bản về MHH

- Mô hình hóa quá trình ngẫu nhiên:
 - Đầu vào và đầu ra của hệ thống và các phân hệ: các quá trình ngẫu nhiên mong muốn (thông tin) và không mong muốn (nhiều và giao thoa)
 - Các quá trình được mô phỏng: sao chép các tính chất của các quá trình thực
 - Nhiệm vụ mô hình hóa: Sao chép quá trình ngẫu nhiên → tạo ra đặc tính đầu ra chính xác
 - Mô hình quá trình ngẫu nhiên tương đương: tiết kiệm thời gian tính toán
- Mô hình hóa hệ thống giả định
 - Trong thiết kế hệ thống: Đặc tính kỹ thuật của hệ thống chưa được biết → hệ thống giả định
 - Giả sử một số lượng nhỏ nhất các thành phần mà vẫn thu được một hệ thống hợp lý

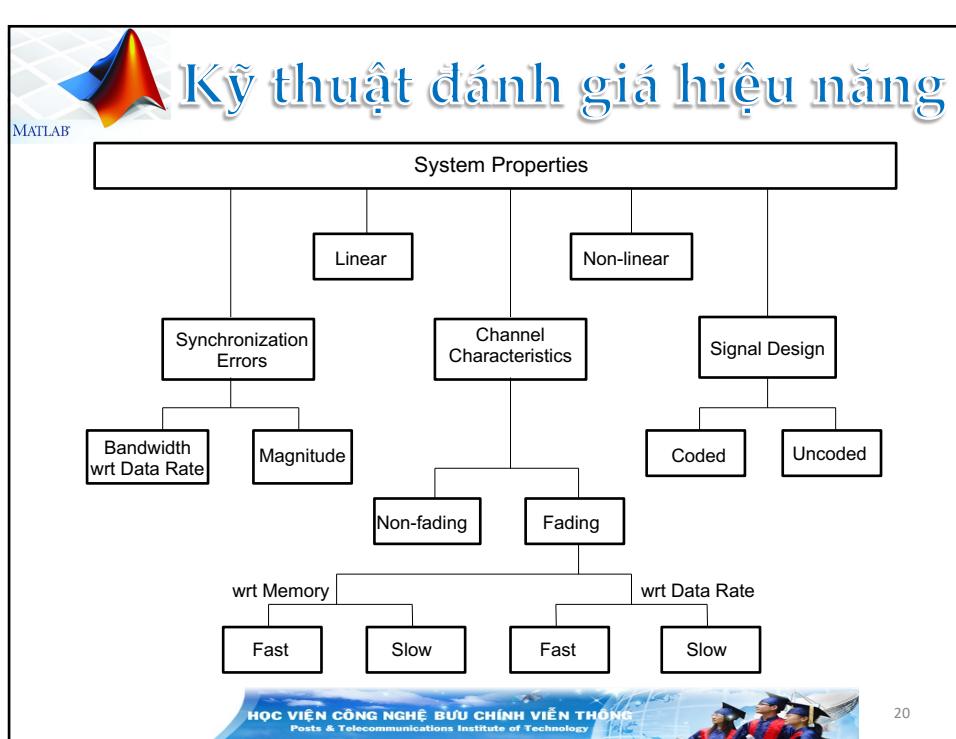
18

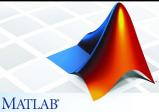


Kỹ thuật đánh giá hiệu năng

- Kỹ thuật đánh giá hiệu năng:
 - Tập hợp các công cụ giải tích và các giả định → ước tính hiệu quả đại lượng hiệu năng
- Mô phỏng Monte Carlo
 - Tỉ số lỗi bit BER được ước tính: cho N bit qua hệ thống và đếm lỗi
 - Đảm bảo độ tin cậy: Số bit cần để quan sát trong phạm vi $10/p$ đến $100/p$, $p = \text{BER}$ thực.
- Một số kỹ thuật PET thay thế
 - Kỹ thuật bán giải tích (quasianalytical estimation)
 - Kỹ thuật lấy mẫu quan trọng (Importance sampling)
 - Kỹ thuật ngoại suy (Extrapolation)

19





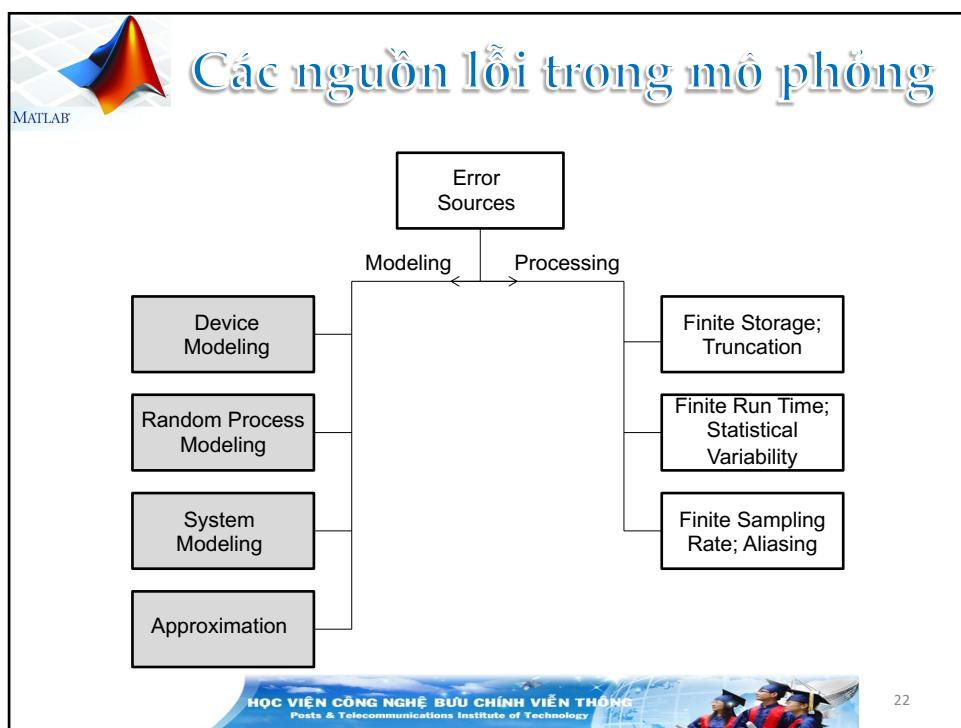
Các nguồn lỗi trong mô phỏng

- Độ chính xác của mô phỏng bị giới hạn bởi:
 - Ba kiểu sai số mô hình hóa:
 - Mô hình hóa hệ thống
 - Mô hình hóa thành phần linh kiện
 - Mô hình hóa quá trình ngẫu nhiên
 - Sai số xử lý

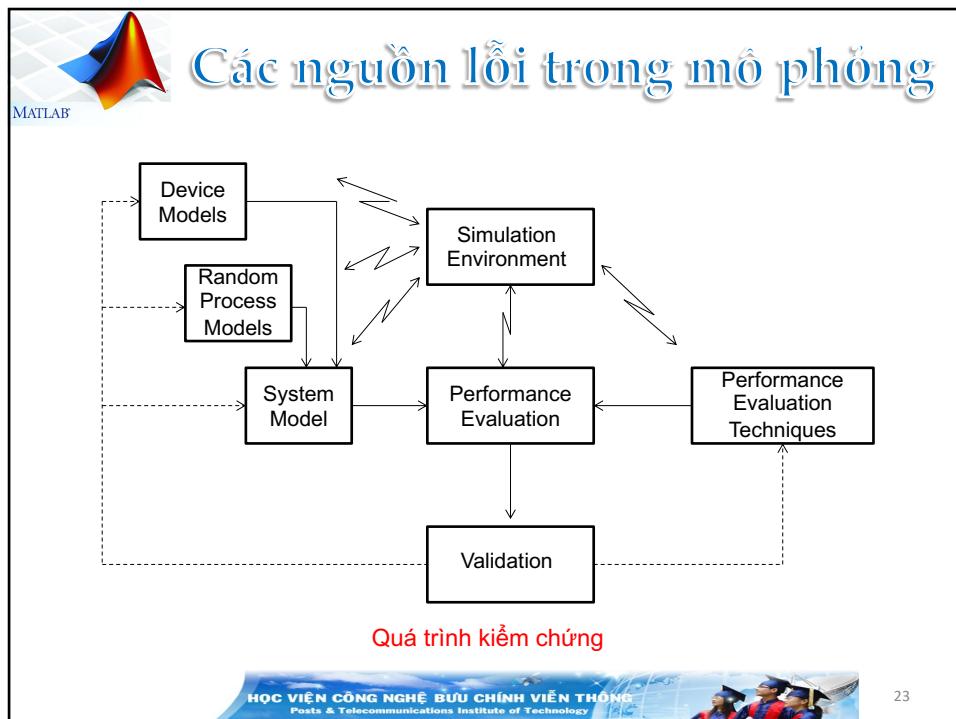
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

21

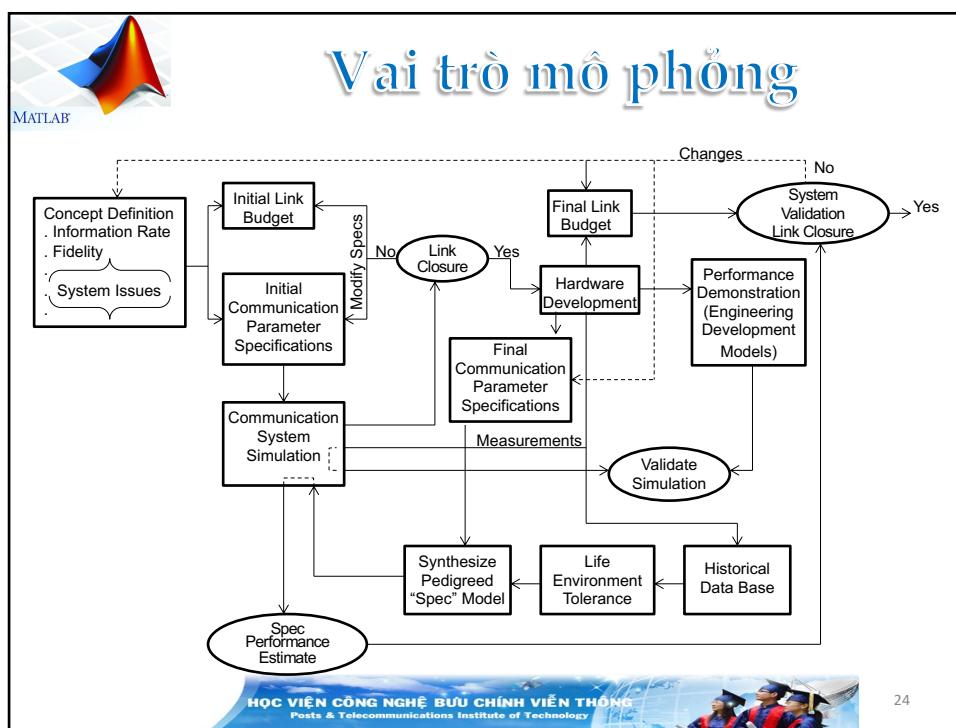
21



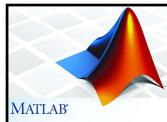
22



23



24



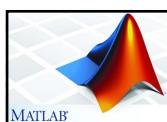
Giới thiệu về MATLAB

- Giới thiệu chung
- Các cấu trúc cơ bản trong MATLAB
- Hoạt động ma trận và vectơ
- Lập trình trong MATLAB
- Phương trình vi phân

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

25

25



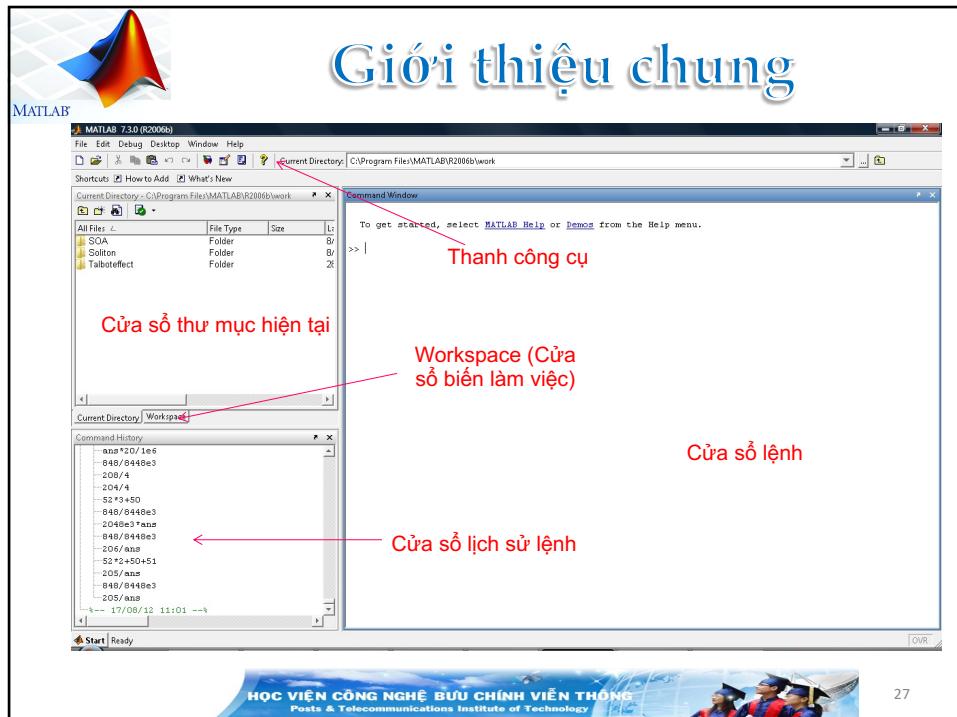
Giới thiệu chung

- MATLAB: MATrix LABoratory
 - Là một công cụ mô phỏng và tính toán số
 - Các hoạt động tính toán dựa trên cấu trúc dữ liệu đơn hay *matrix* → cú pháp trong MATLAB đơn giản, chương trình dễ viết hơn các ngôn ngữ lập trình bậc cao hoặc các chương trình đại số máy tính khác.
 - MATLAB là một ngôn ngữ dịch, tất cả các lệnh có thể được thực hiện trực tiếp
 - Được bổ sung thêm “symbolics” toolbox → cho phép thực hiện tính toán dạng “symbolic” như các chương trình MAPLE hoặc MATHEMATICA.
 - Khả năng tương tác với *Simulink*, một toolbox đặc biệt – công cụ để xây dựng chương trình mô phỏng dựa trên giao diện đồ họa.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

26

26



27

27

The screenshot shows the MATLAB desktop environment. The title bar says "MATLAB 7.3.0 (R2006b)". The main window contains several panes:

- Các cấu trúc cơ bản:** A list of basic MATLAB structures.
- Command Window:** Displays the message "HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG Posts & Telecommunications Institute of Technology" and the command ">> x = 2.4".

The list of basic MATLAB structures includes:

- Các biến MATLAB
 - Kiểu dữ liệu cơ bản: matrix
 - Định nghĩa các biến MATLAB:

```

>> x = 2.4
x =
2.4000

>> vector = [1 5 -3]
vector =
    1      5      -3

>> thematrix = [3 1+2*i 2;4 0 -5]

thematrix =
3.0000      1.0000 + 2.0000i  2.0000
4.0000          0            -5.0000

```

28

28

Các cấu trúc cơ bản

- Các biến MATLAB
 - Workspace:

Sử dụng lệnh `who` hoặc `whos` để kiểm tra biến
Để xóa biến sử dụng lệnh `clear`

The screenshot shows the MATLAB interface with two windows open. The top window is the 'Workspace' browser, showing variables like ans, colvector, thematrix, vector, and x. The bottom window is the 'Array Editor' for the variable 'thematrix', displaying a 2x7 matrix with values 3, 1, +, ..., 2 in row 1 and 4, 0, -5 in row 2. A banner at the bottom of the slide reads 'HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG Posts & Telecommunications Institute of Technology'.

29

29

Các cấu trúc cơ bản

- Các biến MATLAB
 - Xử lý các biến:

```
>> A = [1 2 3; 4 5 6; 7 8 9]
```

A =	>> B = A(2,:)
1 2 3	B =
4 5 6	4 5 6
7 8 9	

```
>> A(:,1) = []
```

A =	>> A(2,:) = []
2 3	A =
5 6	1 2 3
8 9	7 8 9

The screenshot shows the MATLAB interface with code examples demonstrating matrix manipulation. It includes the workspace browser and command window. A banner at the bottom of the slide reads 'HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG Posts & Telecommunications Institute of Technology'.

30

30



Các cấu trúc cơ bản

- Các biến MATLAB:
 - Bài tập:

1. Tạo các vectơ và ma trận trong MATLAB với các biến:

$$M = \begin{pmatrix} 1 & 0 & 0 \\ 0 & j & 1 \\ j & j+1 & -3 \end{pmatrix},$$

$$k = 2.75,$$

$$\vec{v} = \begin{pmatrix} 1 \\ 3 \\ -7 \\ -0.5 \end{pmatrix},$$

$$\vec{w} = (1 \quad -5.5 \quad -1.7 \quad -1.5 \quad 3 \quad -10.7),$$

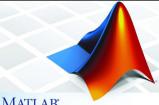
$$\vec{y} = (1 \quad 1.5 \quad 2 \quad 2.5 \quad \dots \quad 100.5).$$

2. Khai triển ma trận M thành ma trận V 6×6 : $V = \begin{pmatrix} M & M \\ M & M \end{pmatrix}$.
Xóa hàng 2 và cột 3 từ ma trận V
Tạo vectơ z từ hàng 4 của ma trận V
Biến đổi giá trị tại V(4,2) thành j+5

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

31

31



Các cấu trúc cơ bản

- Các hoạt động số học
 - Các phép tính ma trận:

```
>> M = [1 2 3; 4 -1 2] % define 2x3-Matrix M
M =
1     2     3
4    -1     2

>> N = [1 2 -1 ; 4 -1 1; 2 0 1] % define 3x3-Matrix N
N =
1     2     -1
4    -1      1
2     0      1

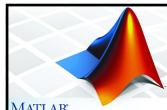
>> V = M*N % trying the product M*N
V =
15     0     4
4     9    -3

>> W = N*M % trying the product N*M
??? Error using ==> mtimes
Inner matrix dimensions must agree.
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

32

32



Các cấu trúc cơ bản

- Các hoạt động số học
 - Các phép tính theo phần tử: sử dụng . (dot) để phân biệt

```
>> M = [1 2 3; 4 -1 2] % define 2x3-Matrix M
M =
1     2     3
4    -1     2

>> N = [1 -1 0; 2 1 -1] % define 2x3-Matrix N
N =
1     -1     0
2      1    -1

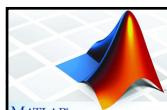
>> M*N % Matrix product M*N
??? Error using ==> mtimes
Inner matrix dimensions must agree.

>> M.*N % term-by-term multiplication
```



33

33



Các cấu trúc cơ bản

- Các hoạt động số học
 - Các phép tính chia: phân biệt chia trái (\) và chia phải (/)

```
>> A = [2 1 ;1 1] % Matrix A
A =
2     1
1     1

>> B = [- 1 1;1 1] % Matrix B
B =
-1     1
1     1

>> X1 = A/B % right division
X1 =
-0.5000    1.5000
0         1.0000

>> Y1 = A\B % left division
Y1 =
-2.0000   -0.0000
3.0000    1.0000

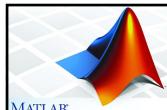
>> X2 = A*Binv % checking
X2 =
-0.5000    1.5000
0         1.0000

>> Y2 = Ainv*B % checking
Y2 =
-2         0
3         1
```



34

34



Các cấu trúc cơ bản

- Các hoạt động số học
 - Các phép tính chia: phân biệt chia trái (\) và chia phải (/)

```

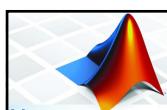
>> A = [2 1 ;1 1]           % Matrix A
A =
2     1
1     1
>> x = A\b                % left division of A "by" b
x =
1.0000
0.0000
>> b=[2; 1]                % column vector b
b =
2
1
>> y = A/b                % right division of A "by" b
??? Error using ==> mrdivide
Matrix dimensions must agree.

```

Here “left division,” $\vec{x} = A \setminus \vec{b}$, obviously yields a *solution* (in this case unique) of the linear system of equations $A\vec{x} = \vec{b}$, as the following test shows:

35

35



Các cấu trúc cơ bản

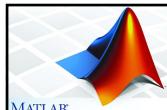
- Các hoạt động số học
 - Bài tập:

3. Tính tích 2 ma trận: $A = \begin{pmatrix} -1 & 3.5 & 2 \\ 0 & 1 & -1.3 \\ 1.1 & 2 & 1.9 \end{pmatrix}$ và $B = \begin{pmatrix} 1 & 0 & -1 \\ -1.5 & 1.5 & -3 \\ 1 & 1 & 1 \end{pmatrix}$,
4. Dùng hoạt động ma trận để biến đổi từ $A = \begin{pmatrix} -1 & 3.5 & 2 \\ 0 & 1 & -1.3 \\ 1.1 & 2 & 1.9 \end{pmatrix}$. thành $C = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1.9 \end{pmatrix}$
5. Tính ma trận đảo của M bằng phép chia

$$M = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ -1 & 0 & 0 \end{pmatrix}$$

36

36



Các cấu trúc cơ bản

- Các hoạt động logic
 - Các hoạt động logic cho ra kết quả *true* (1) hoặc *false* (0)

```
>> A=[1 -3 ;0 0]           >> B=[0 5 ;0 1]           >> res=A&B
A =
1   -3
0   0
B =
0   5
0   1
res =
0   0

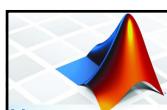
>> comp = A>B           help ops
comp =
1   0
0   0
```



37

37

37



Các cấu trúc cơ bản

- Các hoạt động logic
 - Các hoạt động logic cho ra kết quả *true* (1) hoặc *false* (0)

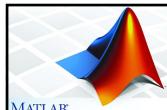
```
>> vect=[-2, 3, 0, 4, 5, 19, 22, 17, 1]
vect =
-2    3    0    4    5    19   22   17   1
>> compvect=2*ones(1, 9)
compvect =
2    2    2    2    2    2    2    2    2
>> comp=vect>compvect           >> res=vect(comp)
comp =
0    1    0    1    1    1    1    0    3    4    5    19   22   17
res =
0    1    0    1    1    1    1    0    3    4    5    19   22   17
```



38

38

38



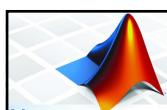
Các cấu trúc cơ bản

- Các hoạt động logic
 - Các hoạt động logic cho ra kết quả *true* (1) hoặc *false* (0)

```
>> logiField1 = [true, true, false, true, false, true]
logiField1 =
    1     1     0     1     0     1
>> numField = [1, 1, 0, 1, 0, 1]
numField =
    1     1     0     1     0     1
>> logiField2 = logical(numField)
logiField2 =
    1     1     0     1     0     1
>> whos
  Name          Size            Bytes  Class
  logiField1    1x6              6  logical array
  logiField2    1x6              6  logical array
  numField      1x6             48  double array
```

 39

39



Các cấu trúc cơ bản

- Các hoạt động logic
 - Bài tập:
 - Kiểm tra và giải thích kết quả hoạt động logic AND và OR giữa 2 ma trận trong bài tập 3.
 - Kiểm tra và giải thích kết quả hoạt động quan hệ giữa 2 vecto:
 $\vec{x} = (1 \ -3 \ 3 \ 14 \ -10 \ 12)$ và $\vec{y} = (12 \ 6 \ 0 \ -1 \ -10 \ 2)$.
 - Cho ma trận:

$$C = \begin{pmatrix} 1 & 2 & 3 & 4 & 10 \\ -22 & 1 & 11 & -12 & 4 \\ 8 & 1 & 6 & -11 & 5 \\ 18 & 1 & 11 & 6 & 4 \end{pmatrix}$$

Sử dụng các toán tử quan hệ để đặt các số hạng trong ma trận có giá trị > 10 và < -10 bằng 0.

 40

40



Các cấu trúc cơ bản

- Các hàm toán học
 - Các hoạt động được thực hiện theo từng phần tử help elfun

```

>> help asin
>> t=(0:1:5)
ASIN Inverse sine.
ASIN(X) is the arcsine of the elements of X. Complex
results are obtained if ABS(X) > 1.0 for some element.
t =
See also sin, asind.

0    1    2    3    4    5
Overloaded functions or methods
(ones with the same name in other directories)
help sym/asin.m

>> s=sin(t)
Reference page in Help browser
doc asin

s =
0    0.8415    0.9093    0.1411   -0.7568   -0.9589

```

41

41



Các cấu trúc cơ bản

- Các hàm toán học
 - Các hoạt động được thực hiện theo từng phần tử

```

>> cnum=[1+j, j, 2*j, 3+j, 2-2*j, -j]                                >> magn=abs(cnum)
cnum =                                                               magn =
Columns 1 through 3
1.0000 + 1.0000i   0 + 1.0000i   0 + 2.0000i   >> phase=angle(cnum)
Columns 4 through 6
3.0000 + 1.0000i   2.0000 - 2.0000i   0 - 1.0000i   phase =
0.7854   1.5708   1.5708   0.3218   -0.7854   -1.5708
>> deg=angle(cnum)*360/(2*pi)

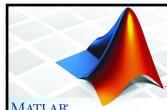
deg =
>> meas=[25.5 16.3 18.0; ...
2.0   6.9   3.0; ...
0.05  4.9   1.1]   >> dBmeas=20*log10(meas)

meas =                                                               dBmeas =
25.5000   16.3000   18.0000   28.1308   24.2438   25.1055
2.0000   6.9000   3.0000   6.0206   16.7770   9.5424
0.0500   4.9000   1.1000   -26.0206   13.8039   0.8279

```

42

42



Các cấu trúc cơ bản

- Các hàm toán học
 - Bài tập:

9. Tính giá trị của tín hiệu: $s(t) = \sin(2\pi 5t) \cos(2\pi 3t) + e^{-0.1t}$ với vectơ thời gian từ 0 đến 10 có cỡ bước 0,1.

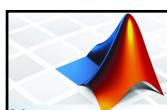
10. Tính giá trị của tín hiệu: $s(t) = \sin(2\pi 5.3t) \sin(2\pi 5.3t)$ theo vectơ thời gian của bài 9.

11. Làm tròn giá trị của vectơ: $s(t) = 20 \sin(2\pi 5t)$ về giá trị nguyên gần nhất theo vectơ thời gian của bài 9.

12. Tính logarithm cơ số 2 và 10 của vectơ:
 $\vec{b} = (1024 \quad 1000 \quad 100 \quad 2 \quad 1)$

43

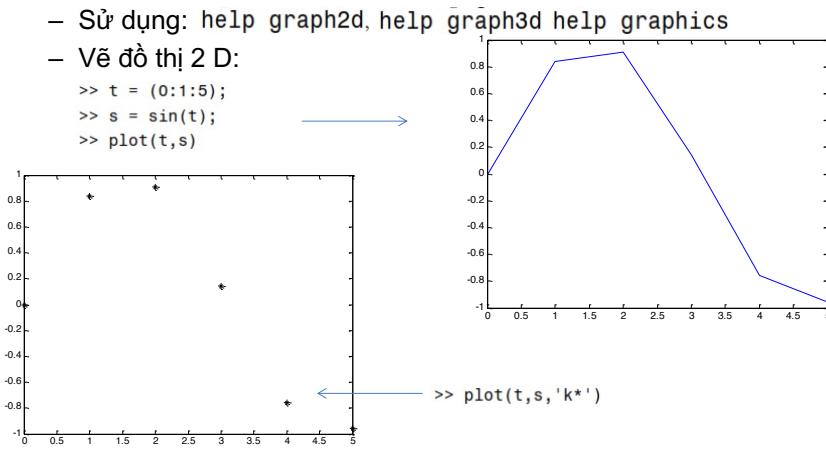
43



Các cấu trúc cơ bản

- Các hàm đồ họa
 - Sử dụng: `help graph2d, help graph3d help graphics`
 - Vẽ đồ thị 2 D:

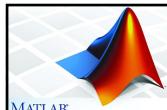
```
>> t = (0:1:5);
>> s = sin(t);
>> plot(t,s)
```



```
>> plot(t,s,'k*')
```

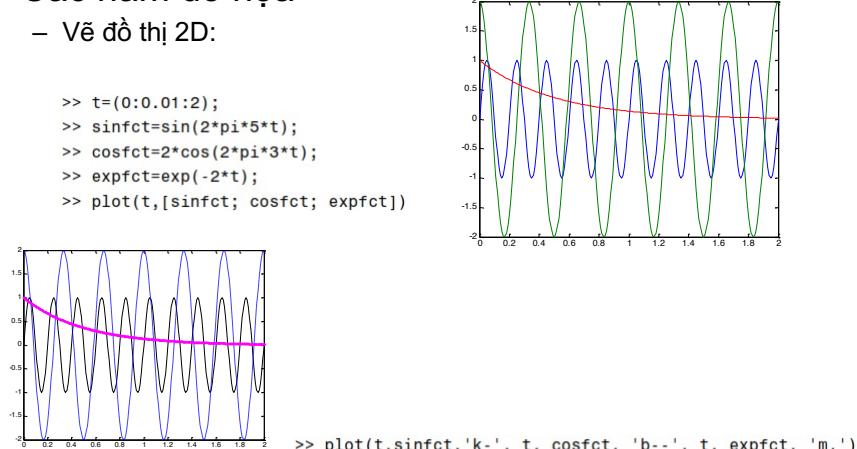
44

44

 **Các cấu trúc cơ bản**

- Các hàm đồ họa
 - Vẽ đồ thị 2D:

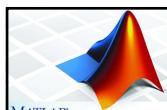
```
>> t=(0:0.01:2);
>> sinfct=sin(2*pi*5*t);
>> cosfct=2*cos(2*pi*3*t);
>> expfct=exp(-2*t);
>> plot(t,[sinfct; cosfct; expfct])
```



```
>> plot(t,sinfct,'k-', t, cosfct, 'b--', t, expfct, 'm.')
```

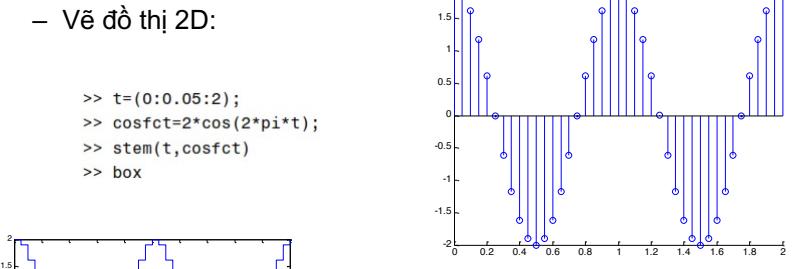
 45

45

 **Các cấu trúc cơ bản**

- Các hàm đồ họa
 - Vẽ đồ thị 2D:

```
>> t=(0:0.05:2);
>> cosfct=2*cos(2*pi*t);
>> stem(t,cosfct)
>> box
```



```
>> stairs(t,cosfct)
```

 46

46

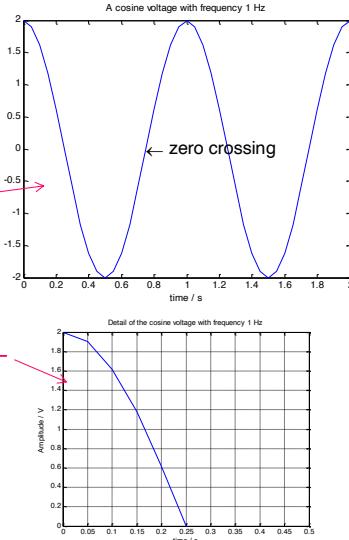
Các cấu trúc cơ bản



• Các hàm đồ họa

- Các hàm

```
>> t=(0:0.05:2);
>> cosfct=2*cos(2*pi*t);
>> plot(t,cosfct)
>> xlabel('time / s')
>> ylabel('Amplitude / V')
>> text (0.75,0,'<-- zero crossing','FontSize',18)
>> title('A cosine voltage with frequency 1 Hz')
>> figure % open a new window !
>> plot(t,cosfct)
>> xlabel('time / s')
>> ylabel('Amplitude / V')
>> grid % set grid frame
>> axis([0, 0.5, 0, 2]) % detail within interval [0, 0.5],
% but only amplitudes within
% the interval [0, 2]
>> title('Detail of the cosine voltage with frequency 1 Hz')
```

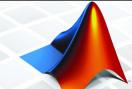


HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

47

47

Các cấu trúc cơ bản

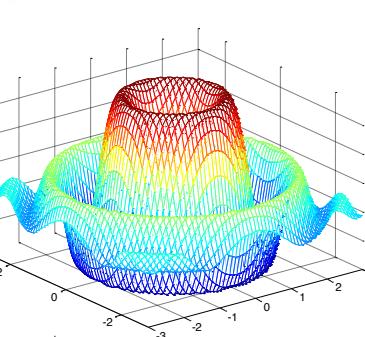


• Các hàm đồ họa

- Vẽ đồ thị 3 D: sử dụng *mesh* hoặc *surf*

$$f(x,y) = \sin(x^2 + y^2)e^{-0.2 \cdot (x^2+y^2)}$$

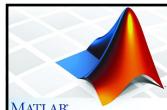
```
>> x=(-3:0.1:3); % grid frame in x direction
>> y=(-3:0.1:3)'; % grid frame in y direction
>> v=ones(length(x),1); % auxiliary vector
>> X=v*x; % grid matrix of the x values
>> Y=y*v'; % grid matrix of the y values
>> f=sin(X.^2+Y.^2).*exp(-0.2*(X.^2+Y.^2)); % function value
>> mesh(x,y,f) % mesh plot with mesh
>> mxnf = max(max(f)); % maximum value of the function
>> mif = min(min(f)); % minimum value of the function
>> axis([-3,3,-3,3,mif,mxf]) % adjust axes
>> xlabel('x-axis'); % label axes
>> ylabel('y-axis');
```



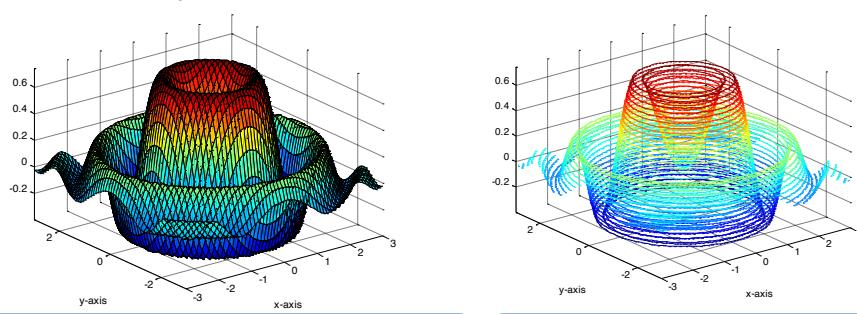
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

48

48

 **Các cấu trúc cơ bản**

- Các hàm đồ họa
 - Vẽ đồ thị 3 D:

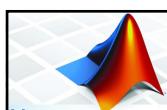


```
>> figure % new plot
>> surf(x,y,f) % surface mesh plot with surf
>> axis([-3,3,-3,3,mif,mxf]) % adjust axes
>> xlabel('x-axis'); % label axes
>> ylabel('y-axis');
```

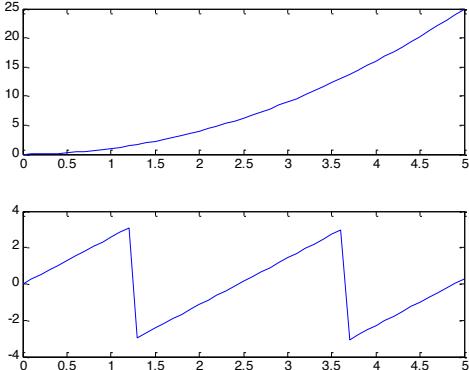
```
>> contour3(x,y,f,30)
>> axis([-3,3,-3,3,mif,mxf]) % adjust axes
>> xlabel('x-axis'); % label axes
>> ylabel('y-axis');
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology 49

49

 **Các cấu trúc cơ bản**

- Các hàm đồ họa
 - Vẽ nhiều đồ thị:

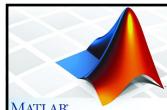


$$f(t) = t^2 e^{jt} f^t$$

```
>> t=(0:0.1:5);
>> f=(t.^2).*exp(j*t).*(j.^t); % * and ^ are field operations.
>> subplot(211) % plot the top graph
>> plot(t,abs(f))
>> subplot(212) % plot the bottom graph
>> plot(t,angle(f))
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology 50

50



Các cấu trúc cơ bản

- Các hàm đồ họa
 - Bài tập:

13. Cho vecto tần số: $\omega = (0.01, 0.02, 0.03, 0.04, \dots, 5)$ rad/s, và các hàm truyền của một bộ tích phân và của một phần tử trễ thời gian bậc 1 tương ứng:

$$H(j\omega) = \frac{1}{j\omega} \quad H(j\omega) = \frac{1}{1+j\omega},$$

thường gặp trong xử lý tín hiệu và kỹ thuật điều khiển. Hãy vẽ đồ thị biên độ của các hàm truyền này trên 2 hình riêng biệt.

Sử dụng các hàm semilogx, semilogy và loglog để thay đổi kết quả biểu diễn đồ thị theo các kiểu trực khác nhau. Xác định kiểu biểu diễn nào là tốt nhất.

14. Vẽ biên độ và pha của các hàm truyền cho ở bài 13 trên cùng một hình.

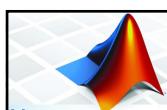
15. Tính và vẽ hàm $x^2 + y^2$ trong dải $[-2,2] \times [-1,1]$ sử dụng lưới có cỡ bước 0.2 theo chiều x và 0.1 theo chiều y.

16. Vẽ hình cầu có bán kính $R = 3$.



51

51



Các cấu trúc cơ bản

- Các hoạt động I/O
 - Sử dụng các lệnh `save` và `load`: để lưu hoặc nạp các dữ liệu từ file trong MATLAB.

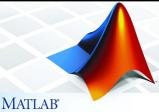
```
>> save wsbackup                               help iofun,
>> save 'C:\ndnhnhan\matlab7\thevars' var1 var2 -V6

>> save thevarX.txt X -ASCII
>> load -ASCII thevarX.txt
```



52

52



Các cấu trúc cơ bản

- Điều khiển ma trận
 - Xem: help elmat

```

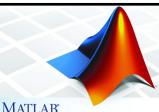
>> M = zeros(2,2)           >> N = ones(3,2)           >> x1 = [1,2,3,4,5,6];
                                         >> v=zeros(length(x1),1)
M =
N =
v =
0     0          1     1
0     0          1     1          0
                  1     1          0
                                         0
>> E5 = eye(5) % Tạo ma trận đơn vị
E5 =
1   0   0   0   0           >> length(x1) % Xác định độ dài vector
0   1   0   0   0           ans =
0   0   1   0   0           6
0   0   0   1   0
0   0   0   0   1

```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

53

53



Các cấu trúc cơ bản

- Điều khiển ma trận

```

>> B = E5;                 % store the matrix E5      >> [rows, columns] = size(B) % determine sizes
>> B(:,2) = [ ]            % empty the second column
B =
1   0   0   0
0   0   0   0
0   1   0   0
0   0   1   0
0   0   0   1

```

```

>> M = [1 2; 3 -2; -1 4]    % a 3x2 matrix      >> N = M'
M =
1   2
3  -2
-1   4

```

```

                                         >> N = % Chuyển vị ma trận
                                         % Xác định kích thước ma trận
                                         rows =
                                         5
                                         columns =
                                         4
                                         1   3   -1
                                         2  -2   4

```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

54

54



Các cấu trúc cơ bản

- Điều khiển ma trận

```

>> M = [i 2; 3 -j]           % a 2x2 matrix with complex terms    >> K = M.'
M =
0 + 1.0000i 2.0000
3.0000      0 - 1.0000i
>> N = M'                  % the transposed matrix
N =
0 - 1.0000i 3.0000
2.0000      0 + 1.0000i
>> M = [1 2; 3 -2; -1 4]   % a 3x2 matrix
M =
1     2
3     -2
-1    4
>> K =
0 + 1.0000i 2.0000
3.0000      0 - 1.0000i
>> zVec = [0, 3, -1, 0, 1, 99]
zVec =
0     3     -1     0     1     99
>> mVec = M(:)
>> N = repmat(M,2,2)
zVec =
0     3     -1     0     1
0     3     -1     0     1

```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

55

55



Các cấu trúc cơ bản

- Các cấu trúc – structures

- Ví dụ: Định nghĩa structure Graphic

```

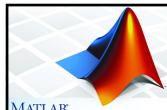
>> Graphic.title = 'Example';
>> Graphic.xlabel = 'time / s';
>> Graphic.ylabel = 'voltage / V';
>> Graphic.num = 2;
>> Graphic.color = ['r', 'b'];
>> Graphic.grid = 1;
>> Graphic.xVals = [0,5];
>> Graphic.yVals = [-1,1];
>> whos
  Name          Size            Bytes  Class
  Graphic       1x1             1096  struct array
>> Graphic = setfield(Graphic, 'title', 'The next example')
>> Graphicempty = struct('title', [], 'xlabel', [], ...
  'ylabel', [], 'num', [], ...
  'color', [], 'grid', [], ...
  'xVals', [], 'yVals', [])
  >> Grfarray = repmat(Graphicempty, 10, 1)
  >> Thirdarray = Grfarray(3)
  >> ttlString = Graphic.title
  >> Graphic.title = 'Another example'
  >> Grfarray(3).xVals = [0,10];

```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

56

56



Các cấu trúc cơ bản

- Điều khiển ma trận và cấu trúc
 - Bài tập:

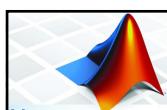
17. Tạo vectơ $y = (1, 1.5, 2, \dots, 4.5, 5)$. Sử dụng hoạt động điều khiển ma trận phù hợp để đảo trật tự các số hạng của vectơ y để tạo ra vectơ $yr = (5, 4.5, \dots, 1.5, 1)$.

Tạo vectơ z chỉ chứa các số nguyên từ vectơ y .

18. Định nghĩa cấu trúc **color** với các trường dữ liệu *red*, *blue* và *green*. Sau đó định nghĩa một trường 1×20 của các cấu trúc kiểu này và khởi tạo thành phần *red* bằng giá trị 'yes', thành phần *blue* bằng giá trị 'no' và thành phần *green* với giá trị [0,256,0]

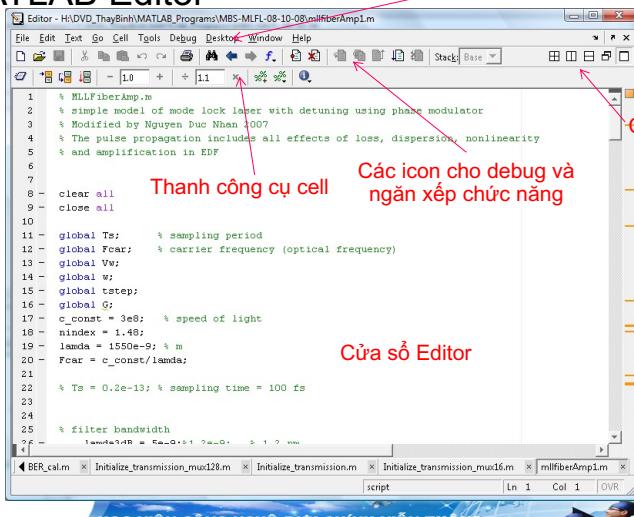
57

57



Lập trình trong MATLAB

- MATLAB Editor

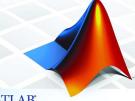


HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

58

58

29



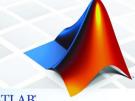
Lập trình trong MATLAB

- Các thủ tục
 - Cung cấp các tập lệnh được thực hiện trong cửa sổ lệnh bằng một lệnh đơn giản.
 - Các chuỗi lệnh được viết bằng Editor và được lưu trong một *m-file* với tên sẽ được sử dụng để chạy trong cửa sổ lệnh.
 - Sử dụng lệnh *help* để kiểm tra sự tồn tại của hàm.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

59

59



Lập trình trong MATLAB

- Các thủ tục

```
>> vidu1
```

```

1 % Procedure vidu1
2 %
3 % call: vidu1
4 %
5 % Vi du đầu tiên về việc viết chương trình m-file
6 %
7 % Môn học: Mô phỏng hệ thống truyền thông
8 %
9 % Giảng viên: TS. Nguyễn Đức Nhân
10 % Khoa Viễn thông 1
11 %
12 % Năm học 2012
13
14 - t=(0:0.01:2);
15 - sinfct=sin(2*pi*5*t);
16 - cosfct=2*cos(2*pi*3*t);
17 - expfct=exp(-2*t);
18 - plot(t,[sinfct;cosfct;expfct])
19 - xlabel('time / s')
20 - ylabel('Amplitude')
21 - title('three gorgeous signals')
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

60

60



Lập trình trong MATLAB

- Các function
 - Cấu trúc:

```
function [out1, out2, ...] = funname(in1,in2, ...)
    ↑
    Các tham số đầu ra   ↑
    Các tham số đầu vào
    ↑
    Tên hàm
```

Lưu ý: Tên hàm phải trùng tên của *m-file* chứa hàm

Các biến trong *function* là các biến cục bộ

Gọi hàm ở cửa sổ lệnh:

```
>> [y1,y2,...] = funname(x1,x2,...) hoặc
>> funname(x1,x2,...)
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

61

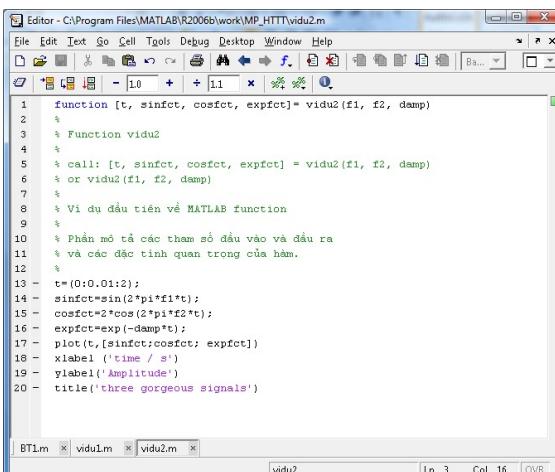
61



Lập trình trong MATLAB

- Các function

```
>> [t,s1,c1,e1] = vidu2(3,5,4);
```



The screenshot shows the MATLAB Editor window with the code for *vidu2.m*. The code defines a function *vidu2* that takes three input parameters (f1, f2, damp) and returns four output variables (t, s1, c1, e1). The code includes comments explaining the purpose of each part of the function.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

62

62



Lập trình trong MATLAB

- Các thủ tục
 - Bài tập:

19. Viết một chương trình MATLAB có tên *circle_prog.m* để thực hiện các hoạt động sau: vẽ đường tròn có bán kính $r = 3$, trả về các kết quả tính chu vi và diện tích hình tròn.(Hint: sử dụng lệnh *axis equal* để hiển thị đồ thị tốt hơn)

20. Thay đổi chương trình trên để hiển thị kết quả với 5 số sau dấu phẩy. (Hint: có thể dùng lệnh *sprintf*)


63

63



Lập trình trong MATLAB

- Các cấu trúc ngôn ngữ MATLAB

<p>Programming language constructs.</p> <p>Control flow.</p> <pre> if - Conditionally execute statements. else - IF statement condition. elseif - IF statement condition. end - Terminate scope of FOR, WHILE, SWITCH, TRY and IF statements. for - Repeat statements a specific number of times. while - Repeat statements an indefinite number of times. break - Terminate execution of WHILE or FOR loop. continue - Pass control to the next iteration of FOR or WHILE loop. switch - Switch among several cases based on expression. case - SWITCH statement case. otherwise - Default SWITCH statement case. try - Begin TRY block. catch - Begin CATCH block. return - Return to invoking function. </pre> <p>Evaluation and execution.</p> <pre> ... eval - Execute string with MATLAB expression. feval - Execute function specified by string. ... </pre>	<p>help lang</p> <p>Scripts, functions, and variables.</p> <pre> ... </pre> <p>Argument handling.</p> <pre> ... nargin - Number of function input arguments. nargout - Number of function output arguments. varargin - Variable length input argument list. varargout - Variable length output argument list. </pre> <p>Interactive input.</p> <pre> ... pause - Wait for user response. ...</pre>
--	--


64

64



Lập trình trong MATLAB

- Các cấu trúc ngôn ngữ MATLAB
 - Câu lệnh *if*:

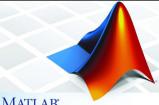
```
if expression1
    statements1
elseif expression2
    statements2
else
    statements3
end
```

```
if A > B
    disp('A lon hon B');
elseif A == B
    disp('A bang B');
else
    disp('A nho hon B');
end
```

 65

65

65



Lập trình trong MATLAB

- Các cấu trúc ngôn ngữ MATLAB
 - Câu lệnh *for*:

```
for variable = expression
    statement
    ...
    statement
end
```

Ví dụ tính giá trị phần tử trong ma trận

```
k = 10;
a = zeros(k,k) % Preallocate matrix
for m = 1:k
    for n = 1:k
        a(m,n) = 1/(m+n -1);
    end
end
```

 66

66

66



Lập trình trong MATLAB

- Các cấu trúc ngôn ngữ MATLAB
 - Câu lệnh *while*:

```
while expression
    statements
end
```

Ví dụ tìm nghiệm của một đa thức bằng phương pháp *bisection*

```
a = 0; fa = -Inf;
b = 3; fb = Inf;
while b-a > eps*b
    x = (a+b)/2;
    fx = x^3-2*x-5; ← Đa thức
    if sign(fx) == sign(fa)
        a = x; fa = fx;
    else
        b = x; fb = fx;
    end
end
x
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

67

67



Lập trình trong MATLAB

- Các cấu trúc ngôn ngữ MATLAB
 - Câu lệnh *switch-case*:

```
switch switch_expr
    case case_expr
        statement, ..., statement
    case {case_expr1, case_expr2, case_expr3, ...}
        statement, ..., statement
    otherwise
        statement, ..., statement
end
```

```
method = 'Bilinear';

switch lower(method)
    case {'linear','bilinear'}
        disp('Method is linear')
    case 'cubic'
        disp('Method is cubic')
    case 'nearest'
        disp('Method is nearest')
    otherwise
        disp('Unknown method.')
end
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

68

68



Lập trình trong MATLAB

- Các cấu trúc ngôn ngữ MATLAB
 - Câu lệnh switch-case:

```
function [t, sinfct, cosfct] = FSwitchIn(f1, f2, damp)
% function FSwitchIn
%
% call: [t, sinfct, cosfct] = FSwitchIn(f1, f2)
% or [t, sinfct, cosfct] = FSwitchIn(f1, f2, damp)
%
% An example of an MATLAB function with a variable
% number of input parameters

t=(0:0.01:2);

switch nargin
  case 2
    sinfct = sin(2*pi*f1*t);
    cosfct = 2*cos(2*pi*f2*t);
    plot(t,[sinfct; cosfct])
    xlabel('time / s')
    ylabel('Amplitude')
    title('sine and cosine oscillations')
```

Ví dụ sử dụng `nargin` và `nargout`

```
case 3
  sinfct = sin(2*pi*f1*t);
  cosfct = 2*cos(2*pi*f2*t);
  expfct = exp(-damp*t);
  plot(t,[sinfct; cosfct; expfct])
  xlabel('time / s')
  ylabel('Amplitude')
  title('three gorgeous signals')
otherwise
  msg = 'The function FSwitchIn must have 2';
  msg = strcat(msg, ' or 3 input parameters!');
  error(msg);
end

if nargout < 3
  msg = 'The function FSwitchIn should return a time';
  msg = strcat(msg, 'vector and two sine signals!');
  error(msg);
end
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

69

69



Lập trình trong MATLAB

- Bài tập

21. Cho một hàm $f(x) = x^3/3 + 4x^2 + x - 6$ trong dải $-1 < x < 3$. Viết chương trình tìm nghiệm phương trình trên bằng phương pháp bisection với sử dụng 2 dự đoán ban đầu tại $x = 0$ và $x = 3$.
(Sử dụng lệnh `input` để cho phép nhập giá trị các tham số đầu vào từ bàn phím khi chạy chương trình)
22. Viết mã chương trình sử dụng vòng lặp để tính tích phân:

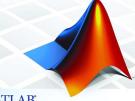
$$h(x) = \int_{-1.5}^{1.5} 4x^3 2e^x \cos(x) dx$$
 bằng phương pháp midpoint với số lượng điểm $N = 100$.
23. Viết mã chương trình sử dụng vòng lặp `while` để tính gần đúng $\sqrt{2}$ dựa trên phương pháp Newton dùng hệ thức đệ quy:

$$x_{n+1} = \frac{x_n^2 + 2}{2 \cdot x_n}, \quad x_0 = 2.$$
 Quá trình lặp thực hiện cho đến khi x_n thay đổi chỉ 0.0001.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

70

70



Lập trình trong MATLAB

- Hàm eval
 - Sử dụng để đánh giá các xâu ký tự (string):


```
>> theCommands = ['x = 2.0; ', 'y = 3.0; ', 'z = x*y; ', 'whos']

theCommands =
x = 2.0; y = 3.0; z = x*y; whos

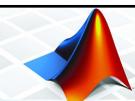
>> eval(theCommands)
Name           Size            Bytes  Class
theCommands    1x31             62  char array
x              1x1              8  double array
y              1x1              8  double array
z              1x1              8  double array

Grand total is 34 elements using 86 bytes
>> z

z =
6
```



71



Lập trình trong MATLAB

- Function handles
 - Một *handle function* hoạt động như con trỏ đến hàm bằng việc sử dụng @ trước hàm đó


```
>> FH_Sin = @sin;
>> whos
Name           Size            Bytes  Class
FH_Sin        1x1             16  function_handle array

Grand total is 1 element using 16 bytes

>> value = sin(2)      >> value = FH_Sin(2)      >> value = feval(FH_Sin, 2)
value =
0.9093
value =
0.9093
value =
0.9093
```



72

Lập trình trong MATLAB

- **Function handles**
 - Ví dụ: Tính tích phân số bằng phương pháp điểm giữa

```
function [integral] = midpoint(a, b, F, N)
% Function midpoint
%
% sample call: integ = midpoint(0, 2, @myfun, 10)
%
% The present example calculates the integral of the function F,
% whose name is passed on as a function handle to midpoint, over
% the limits [a,b]. Midpoint rule is used to calculate the integral;
%
h=(b-a)/N; % subinterval length
% intval=(a+h/2:h:a+(N-1/2)*h); % points marking subintervals
integral = F(a+h/2); % F at the lower limit of the interval
%
For k=2:N
    xi = a + (k-1/2)*h;
    integral = integral+F(xi);
    % integral = integral+F(intval(i));
end;
%
integral = integral*h; % normalizing with h
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

73

73

Giải phương trình vi phân

- **Phương trình vi phân thường (ODE)**
 - Trong mô hình của các hệ thống động: các tham số là hàm của thời gian $y(t)$, vận tốc $\frac{dy}{dt}$ và gia tốc $\frac{d^2y}{dt^2}$
 - Thường hầu hết mô hình các hệ thống động, ta có thể rút gọn từ các phương trình vi phân bậc 2 về các phương trình vi phân bậc 1 có dạng:
$$\frac{d}{dt}y_i(t) = f(y_i(t))$$
 - Hệ thống được mô tả đầy đủ:
$$\begin{aligned} y(x_0) &= y_0 \\ \frac{d}{dx}y(x) &= f(x, y(x)) \end{aligned}$$

Hàm f có thể là hàm tuyến tính hoặc phi tuyến của biến độc lập x và tham số phụ thuộc y .

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

74

74

MATLAB

Giải phương trình vi phân

- Phương pháp Euler
 - Dựa trên gần đúng sai phân hữu hạn đối với đạo hàm

$$y(x+h) \simeq y(x) + hy^{(1)}(x)$$

- Tóm tắt: $y_{n+1} = y_n + hf(x_n, y_n) \quad n = 0, 1, 2, \dots$

$$x_{n+1} = x_0 + (n+1)h = x_n + h$$

```
x(1) = x0;
y(1) = y0;

for k = 2:N % N – Number of steps
    x(k) = x(k-1) + h;
    y(k) = y(k-1)+Fdot(x(k-1),y(k-1))*h;
end;
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

75

75

MATLAB

Giải phương trình vi phân

- Phương pháp Euler biến đổi
 - Sử dụng chuỗi Taylor cho việc phân tích ODE xác định được:

$$y_{i+1} = y_i + \frac{dy_i}{dt}h + \frac{d^2y_i}{dt^2}\frac{h^2}{2!}$$

$$= y_i + \frac{dy_i}{dt}h + \frac{\frac{dy_{i+1}}{dt} - \frac{dy_i}{dt}}{h}\frac{h^2}{2!}$$

$$= y_i + \frac{dy_i}{dt}h + \left(\frac{dy_{i+1}}{dt} - \frac{dy_i}{dt}\right)\frac{h}{2!}$$

$$= y_i + \left(\frac{dy_{i+1}}{dt} + \frac{dy_i}{dt}\right)\frac{h}{2!}$$

Stepping formulas:
 $y_{i+1}^p = y_i + (dy_i / dt)h$
 $y_{i+1}^c = y_i + \frac{(dy_i / dt) + (dy_{i+1}^p / dt)}{2}h$
 $\epsilon_{i+1}^p, \epsilon_{i+1}^c$: Truncation errors

```
t(1) = t0;
y(1) = y0;
for k=2:N
    y1=y(k-1)+h*fdot(t(k-1),y(k-1));
    t1=t(k-1)+h;
    loopcount=0; diff=1;
    while abs(diff)>.05
        loopcount=loopcount+1;
        y2=y(k-1)+h*(fdot(t((k-1),y(k-1))+fdot(t1,y1))/2);
        diff=y1-y2; y1=y2;
    end;
    %collect values together for output
    t(k) = t1; y(k) = y1;
end;
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

76

76

MATLAB

Giải phương trình vi phân

- Phương pháp Runge-Kutta

- Phương pháp RK bậc 2:

Lựa chọn $c_1, c_2, a_2 \rightarrow$ các pp RK khác nhau:

Khi $c_1 = 1/2, c_2 = 1/2, a_2 = 0 \rightarrow$ PP Euler biến đổi

Khi $c_1 = 0, c_2 = 1, a_2 = 1/2$: PP Midpoint

$$y_{i+1} = y_i + c_1 k_1 + c_2 k_2 = y_i + k_2$$

$$k_1 = f(y_i, t_i)h$$

$$k_2 = f(y_i + a_2 k_1, t_i + a_2 h)h$$

$$= f(y_i + (1/2)f(y_i, t_i)h, t_i + (1/2)h)h$$

$$= f(y_i + (1/2)k_1, t_i + (1/2)h)h$$

Khi $c_1 = 1/4, c_2 = 3/4, a_2 = 2/3$:

$$y_{i+1} = y_i + (1/4)k_1 + (3/4)k_2$$

$$k_1 = f(y_i, t_i)h$$

$$k_2 = f(y_i + (2/3)k_1, t_i + (2/3)h)h$$

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

77

77

MATLAB

Giải phương trình vi phân

- Phương pháp Runge-Kutta

- Phương pháp RK bậc 3:

$$y_{i+1} = y_i + c_1 k_1 + c_2 k_2 + c_3 k_3$$

$$k_1 = f(y_i, t_i)h$$

$$k_2 = f(y_i + a_2 k_1, t_i + a_2 h)h$$

$$k_3 = f(y_i + b_3 k_1 + (a_3 - b_3)k_2, t_i + a_3 h)h$$

$c_1 + c_2 + c_3 = 1$	$y_{i+1} = y_i + (2/8)k_1 + (3/8)k_2 + (3/8)k_3$
$c_2 a_2 + c_3 a_3 = 1/2$	$k_1 = f(y_i, t_i)h$
$c_2 a_2^2 + c_3 a_3^2 = 1/3$	$k_2 = f(y_i + (2/3)k_1, t_i + (2/3)h)h$
$c_3(a_3 - b_3)a_2 = 1/6$	$k_3 = f(y_i + (2/3)k_2, t_i + (2/3)h)h$

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

78

78



Giải phương trình vi phân

- Phương pháp Runge-Kutta
 - Phương pháp RK bậc 4:

$K_1 = hF(x, y)$

$$K_2 = hF\left(x + \frac{h}{2}, y + \frac{K_1}{2}\right)$$

$$K_3 = hF\left(x + \frac{h}{2}, y + \frac{K_2}{2}\right)$$

$$K_4 = hF(x + h, y + K_3)$$

$$y(x + h) = y(x) + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

```
xSol(1) = x; ySol(1,:) = y;
k = 1;
while x < xStop
    k = k+1;
    K1 = h*feval(Fdot,x,y);
    K2 = h*feval(Fdot,x + h/2,y + K1/2);
    K3 = h*feval(Fdot,x + h/2,y + K2/2);
    K4 = h*feval(Fdot,x+h,y + K3);
    y = y + (K1+2*K2+2*K3+K4)/6;
    x = x+h;
    xSol(k) = x; ySol(k,:) = y; % Store current soln.
end
```

79

79



Giải phương trình vi phân

- Phương pháp Runge-Kutta
 - Tập các hàm giải phương trình vi phân trong MATLAB:

Solver	Solves These Kinds of Problems	Method
ode45	Nonstiff differential equations	Runge-Kutta
ode23	Nonstiff differential equations	Runge-Kutta
ode113	Nonstiff differential equations	Adams
ode15s	Stiff differential equations and DAEs	NDFs (BDFs)
ode23s	Stiff differential equations	Rosenbrock
ode23t	Moderately stiff differential equations and DAEs	Trapezoidal rule
ode23tb	Stiff differential equations	TR-BDF2
ode15i	Fully implicit differential equations	BDFs

`[T,Y] = solver(odefun,tspan,y0)`
`[T,Y] = solver(odefun,tspan,y0,options)`

Ví dụ: `[t,y] = ode45(myfun,[t0 tf],y0);`



80

80

40

MATLAB

Giải phương trình vi phân

- Ví dụ:**
 - Cho ODE bậc 1: $y'(t) + a y(t) = r$ with $y(0) = y_0$

Nghiệm giải tích: $y(t) = \left(y_0 - \frac{r}{a}\right)e^{-at} + \frac{r}{a}$

```
% Euler method to solve a 1st-order differential equation
clear, clf
a=1;r=1;y0=0; tf=2;
t = [0:0.01:tf]; yt=1-exp(-a*t); % true analytical solution
plot(t,yt,'k'); hold on
klasts = [8 4 2]; hs = tf./klasts;
y(1) = y0;
for itr = 1:3 %with various step size h = 1/8,1/4,1/2
    klast = klasts(itr); h = hs(itr); y(1)=y0;
    for k = 1:klast
        y(k + 1) = (1 - a*h)*y(k) + h*r; % Euler's formula
        plot([k - 1:k]*h,[y(k) y(k+1)],'b', k*h,y(k+1), 'ro')
        if k<4,pause;end
    end
end
```

Nghiệm thu được tại các cỡ bước khác nhau

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

81

81

MATLAB

Giải phương trình vi phân

- Ví dụ:**
 - Đao động con lắc:

$$\ddot{\alpha}(t) = -\frac{g}{l} \cdot \sin(\alpha(t)), \quad g = 9.81 \frac{\text{m}}{\text{s}^2}.$$

$$\begin{aligned} \alpha_1(t) &:= \alpha(t), \quad \dot{\alpha}_1(t) = \alpha_2(t), \\ \alpha_2(t) &:= \dot{\alpha}(t). \end{aligned} \quad \begin{aligned} \dot{\alpha}_1(t) &= \alpha_2(t), \\ \dot{\alpha}_2(t) &= -\frac{g}{l} \cdot \sin(\alpha_1(t)) \end{aligned}$$

Điều kiện ban đầu:

$$\vec{\alpha}(0) = \begin{pmatrix} \alpha(0) \\ \dot{\alpha}(0) \end{pmatrix} = \begin{pmatrix} \alpha_1(0) \\ \alpha_2(0) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Hàm pendde.m

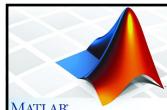
```
[alphadot] = pendde(t,alpha)
%% Setting constants
l=10; % pendulum length
g=9.81; % acceleration of gravity m/s
%% Preliminary initialization
alphadot = [0;0];
%% Representation of the differential equation
% the first first order equation
alphadot(1) = alpha(2);
% the second first order equation
alphadot(2) = -(g/l)*sin(alpha(1));
```

plot(t, solution(:,1),'r-',t, solution(:,2),'g--')

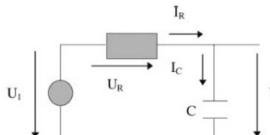
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

82

82

 **Giải phương trình vi phân**

- **Bài tập:**
 24. Viết chương trình tìm nghiệm ptr vi phân: $y'(t) + a y(t) = r$ với $a = 1$, $r = 1$ và $y(0) = 0$ bằng phương pháp Euler biến đổi với cỡ bước $h = 0.25$. Xác định sai số so với nghiệm giải tích tại 2 thời điểm $t = 1$ và $t = 2$.
 25. Tương tự bài tập 24 nhưng sử dụng phương pháp RK bậc 3.
 26. Tương tự bài tập 24 nhưng sử dụng phương pháp RK bậc 4.
 27. Cho sơ đồ mạch RC hình bên:



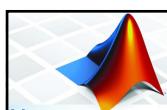
Điện áp đầu ra của hệ thống tuân theo ptr vi phân tuyến tính:

$$\frac{d}{dt}u(t) = -\frac{1}{RC}u(t) + \frac{1}{RC}u_1(t).$$

Hãy viết chương trình tìm nghiệm của ptr này trong khoảng $[0, 3]$ s bằng phương pháp RK bậc 4, biết $C = 4.7\mu F$ và $R = 10 k\Omega$. Hộm $u_1(t)$ là hàm bậc đơn vị. Sau đó so sánh kết quả với nghiệm thu được bằng việc sử dụng lệnh `ode45`.

 83

83

 **Tính toán dạng biểu tượng**

- **Symbolics toolbox:** `help symbolic.`
- Ví dụ: $f(x, y) = \sin(xy^2)\cos(vxy)$

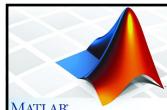
```

>> syms x y v
>> whos
  Name      Size            Bytes  Class
  v          1x1              126  sym object
  x          1x1              126  sym object
  y          1x1              126  sym object
Grand total is 6 elements using 378 bytes
>> f = sin(x*y^2)*cos(v*x*y)      % defining the function
f =
sin(x*y^2)*cos(v*x*y)
>> % differentiate with respect to symbol y
>> dfy = diff(f, 'y')               2
                                         sin(x y ) cos(v x y)
dfy =
2*cos(x*y^2)*x*y*cos(v*x*y)-sin(x*y^2)*sin(v*x*y)*v*x
>> % differentiate with respect to symbol v
>> dfv = diff(f, 'v')
dfv = -sin(x*y^2)*sin(v*x*y)*x*y

```

 84

84



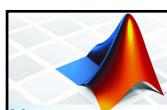
Giới thiệu về Simulink®

- Giới thiệu chung
- Nguyên lý hoạt động và quản lý
- Giải phương trình vi phân
- Đơn giản hóa hệ thống
- Tương tác với MATLAB

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

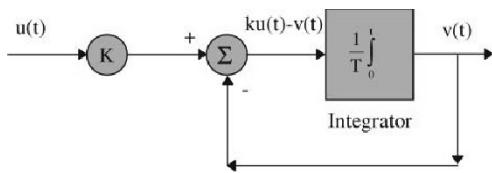
85

85



Giới thiệu chung

- *Simulink®:*
 - Các quá trình phụ thuộc thời gian tuyến tính hay phi tuyến có thể được mô tả bởi các phương trình vi phân.
 - Một cách khác mô tả hệ thống động: thông qua sơ đồ khối



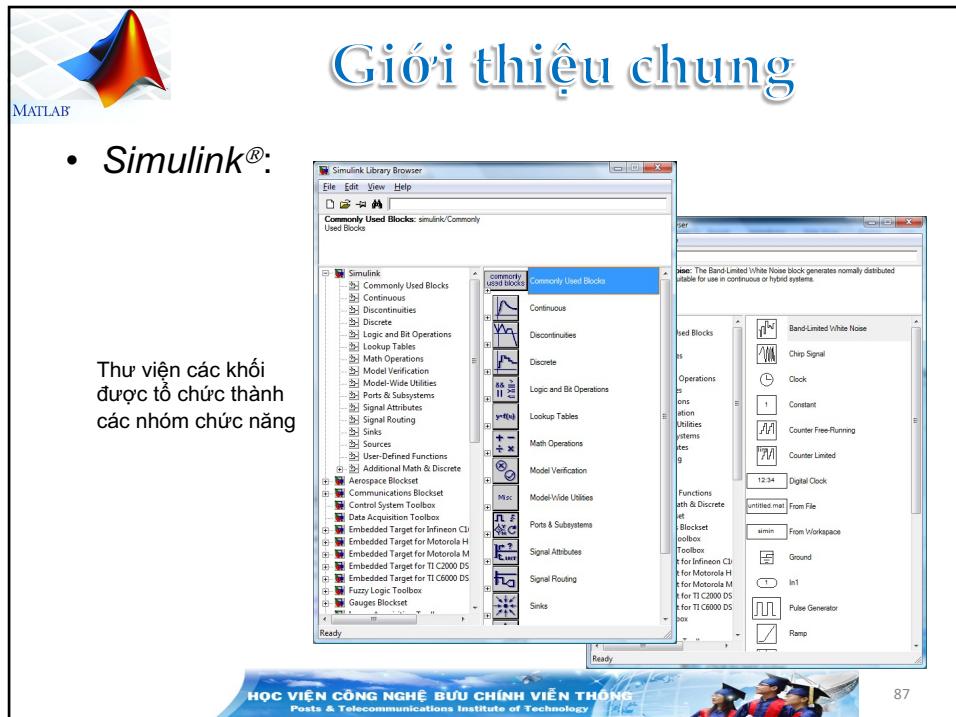
Mô tả một hệ thống động bằng sơ đồ khối

- *Simulink:* bộ giải phương trình vi phân số.

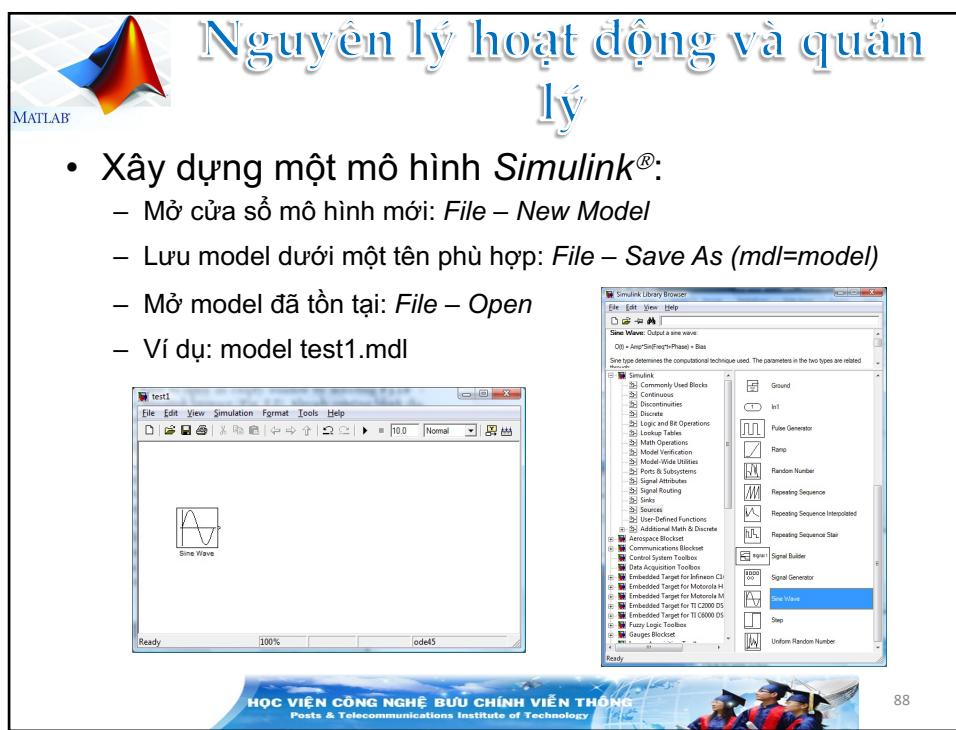
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

86

86



87



88

Nguyên lý hoạt động và quản lý

- Xây dựng một mô hình *Simulink*®:
 - Dùng khối *Sine Wave* trong thư viện *Sources* làm nguồn tín hiệu
 - Dùng khối *Integrator* trong thư viện *Continuous* để lấy tích phân
 - Sử dụng khối *Mux* từ thư viện *Signal Routing* để ghép 2 tín hiệu trước và sau khi lấy tích phân.
 - Dùng khối *Scope* từ thư viện *Sinks* để hiển thị kết quả.

89

89

Nguyên lý hoạt động và quản lý

- Thiết lập tham số cho các khối *Simulink*®:
 - Kích đúp vào mỗi khối chức năng để thiết lập tham số

90

90

Nguyên lý hoạt động và quản lý

- Thiết lập tham số cho các khối Simulink®:
 - Thiết lập tham số tại cửa sổ hiển thị của khối Scope

The screenshot shows the MATLAB interface with a Scope block open. The Scope window displays a plot of a signal over time, with a red arrow pointing to the title bar. To the right, a 'Scope' parameters dialog box is displayed, containing settings for data history, workspace saving, and variable name.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

91

91

Nguyên lý hoạt động và quản lý

- Thiết lập tham số cho các khối Simulink®:

The screenshot shows the Simulink model 'test1'. The block diagram consists of a Signal source (triangle), an Integrator block with gain $\frac{1}{s}$, another Integrator block, a Source block, and a Scope block. Arrows indicate the signal flow from left to right. Below the diagram, configuration parameters are shown, including a command for outputting signals to MATLAB: `plot(S_test1_signals(:,1),S_test1_signals(:,2), S_test1_signals(:,3));`. Other details include the date (04/14/2007), file name (S_test1.mdl), and author (Prof. Dr. Ottmar Beucher).

Hệ thống Simulink test1 hoàn thiện

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

92

92

Nguyên lý hoạt động và quản lý

- Chạy mô phỏng *Simulink®*:

Thiết lập tham số cấu hình mô phỏng

- Sau khi thiết lập xong các tham số → Chạy mô phỏng:

Simulation - start

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

93

93

Nguyên lý hoạt động và quản lý

- Chạy mô phỏng *Simulink®*:
- Các kết quả sau khi chạy mô phỏng có thể xử lý bằng MATLAB

```
>> plot(S_test1.signals(:,1), ...
           [S_test1.signals(:,2),S_test1.signals(:,3)])
>> title('Result of s_test1 with ode3')
>> xlabel('Time /s')
>> ylabel('Function value')
>> grid
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

94

94

MATLAB

Giải phương trình vi phân

- Ví dụ đơn giản:
 - Giải các phương trình vi phân phi tuyến phức tạp đơn giản hơn
 - Chuyển đổi ptr. vi phân thành một hệ thống động → mô tả bằng sơ đồ khối trong Simulink®.
 - Bài toán ptr. vi phân bậc 2: $\ddot{y}(t) = -y(t)$, $y(0) = 1, \dot{y}(0) = 0$.

Nghiệm của ptr. : $y(t) = \cos(t)$.

Kỹ thuật lấy tích phân để tìm $y(t)$

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

95

95

MATLAB

Giải phương trình vi phân

- Ví dụ đơn giản:

Mô hình giải ptr. vi phân bằng Simulink®

Configuration Parameters: a_dg2or/Configuration

Solver	Start time: 0.0	Stop time: 10
Diagnostics	Type: Fixed step	Solver: ode4 (Runge-Kutta)
Diagnostics	Periodic sample time constraint: Unconstrained	
Diagnostics	Step size (fundamental sample time): [0.0]	
Diagnostics	Tasking mode for periodic sample times: Auc	
Diagnostics	<input type="checkbox"/> Higher priority value indicates higher task priority	
Diagnostics	<input type="checkbox"/> Automatically handle data transfers between tasks	

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

96

96

MATLAB

Giải phương trình vi phân

- Một số ví dụ khác:
 - Ptr. vi phân: $y' = y$ $y(0) = 1$

- Ptr. vi phân: $y'' = -2y$ $y(0) = 1, \quad y'(0) = 0$

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

97

97

MATLAB

Giải phương trình vi phân

- Một số ví dụ khác:
 - Ptr. vi phân: $y'' = e^x - 2y$ $y(0) = 1, \quad y'(0) = 0$

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

98

98

**Đơn giản hóa hệ thống
Simulink**

- Khối Fcn:

– Hệ thống Simulink có thể được đơn giản hóa đáng kể bằng việc sử dụng khối Fcn trong thư viện *User-Defined Functions* → các khối phần tử mức thấp nhất (VD: khối Sum hoặc Gain) được loại bỏ.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

99

99

**Đơn giản hóa hệ thống
Simulink**

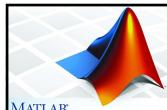
- Xây dựng các phân hệ (Subsystem):

– Trong trường hợp bài toán cỡ lớn → module hóa hệ thống
 – Một số các khối có thể được kết hợp lại thành một phân hệ
 – Cho phép tổ chức hệ thống Simulink® dạng phân cấp
 – Tạo một subsystem: sử dụng menu *Edit/Create Subsystem*

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

100

100

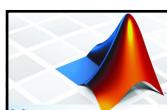


Tương tác với MATLAB

- Truyền các biến giữa Simulink® và MATLAB:
 - Có nhiều cách thực hiện trong Simulink®:
 - Sử dụng khối To Workspace trong Sinks
 - Trong thiết lập tham số cấu hình Simulink®: vào các biến phù hợp trong khối tham số ở mục Data Import/Export – Save to Workspace.
 - Sử dụng phần thiết lập tham số trong khối Scope/ Data History.
 - Giá trị các biến sử dụng trong chương trình Simulink® có thể được định nghĩa trong MATLAB bằng việc viết một chương trình m-file → vào tên biến thay cho giá trị cụ thể trong phần thiết lập tham số cho các khối.

101

101



Tương tác với MATLAB

- Lắp lại các mô phỏng Simulink® trong MATLAB:
 - Khi mô phỏng Simulink® phụ thuộc vào số lượng lớn các tham số → thiết lập sự phụ thuộc hệ thống mô phỏng vào các tham số.
 - Gọi chương trình Simulink® thông qua MATLAB:
 - Sử dụng hàm `sim [T,X,Y] = SIM('model',TIMESPAN,OPTIONS,UT)`
 - Ví dụ: `[t,x,y] = sim ('system', [starttime, endtime]);`
 - Thiết lập các tham số Simulink®:
 - Sử dụng hàm `SET_PARAM('OBJ','PARAMETER1',VALUE1,'PARAMETER2',VALUE2,...)`
 - Ví dụ: `set_param('vdp','Solver','ode15s','StopTime','3000')`
 - Xem các tham số Simulink®:
 - Sử dụng hàm `GET_PARAM('OBJ','PARAMETER')`
 - Ví dụ: `currentGain = get_param('s_denon3/Factorc','Gain')`

102

102

 **Tương tác với MATLAB**

- Lặp lại các mô phỏng *Simulink®* trong *MATLAB*
 - Gọi chương trình *Simulink®* nhiều lần:

```
for i=1:iterations
    % Set the block parameters with set_param
    % for each new iteration.
    reciprocalmass = num2str(1/M(i));
    set_param('s_denon3/invm','Gain',reciprocalmass);
    b = num2str(B(i));
    set_param('s_denon3/Factorb','Gain',b);
    c = num2str(Fc);
    set_param('s_denon3/Factorc','Gain',c);
    [t,x,y] = sim('s_denon3', [0,tm]);
    Y = [Y,y];
end;
```
- Truyền các biến thông qua biến toàn cục:
 - Khai báo tham số như biến toàn cục:

```
global x y z p1 a2 ...
```

 103

 **Giới thiệu về Simulink®**

- Bài tập:
 - Thiết kế một hệ thống *Simulink* để giải bài toán giá trị ban đầu:
 $\ddot{y}(t) + y(t) = 0, \quad y(0) = 1, \dot{y}(0) = 0.$
 Chạy và so sánh kết quả với nghiệm chính xác.
 Biến đổi hệ thống để mô phỏng một hàm nhiễu loạn (về bên phải $\neq 0$)
 được xác định là e^{-t}
 - Giải bài toán giá trị ban đầu: $t\ddot{y}(t) + 2\dot{y}(t) + 4y(t) = 4, \quad y(1) = 1, \dot{y}(1) = 1$
 bằng một hệ thống *Simulink* phù hợp.
 - Giải hệ phương trình vi phân sau:

$$\begin{aligned} \dot{y}_1(t) &= -3y_1(t) - 2y_2(t), & y_1(0) &= 1, \\ \dot{y}_2(t) &= 4y_1(t) + 2y_2(t), & y_2(0) &= 1 \end{aligned}$$

 bằng một hệ thống *Simulink* phù hợp.

 104

 **Mô phỏng tín hiệu và quá trình thu phát**

- Giới thiệu
- Mô phỏng nguồn tín hiệu
- Mã hóa
- Điều chế và giải điều chế
- Quá trình lọc
- Quá trình đồng bộ

 105

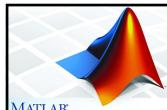
105

 **Giới thiệu**

- *Mô phỏng tín hiệu băng gốc và thông dài:*
 - Tín hiệu băng gốc (baseband): có phổ tần tập trung quanh tần số 0.
 - Tín hiệu thông dài (passband): có phổ tần tập trung quanh một tần số sóng mang f_c .
 - Tín hiệu băng gốc có thể được chuyển đổi thành tín hiệu thông dài qua quá trình đổi tần lên (up-conversion)
 - Tín hiệu thông dài có thể được chuyển đổi thành tín hiệu băng gốc qua quá trình đổi tần xuống (down-conversion)
 - Tín hiệu thông dài $s_P(t)$ được xây dựng từ hai tín hiệu băng gốc $s_I(t)$ và $s_Q(t)$ (trong điều chế số)

 106

106



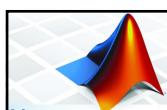
Giới thiệu

- Mô phỏng tín hiệu băng gốc và thông dải:**
 - Tín hiệu thông dải có thể được viết:
$$s_P(t) = \sqrt{2} \cdot A s_I(t) \cdot \cos(2\pi f_c t) + \sqrt{2} \cdot A s_Q(t) \cdot \sin(2\pi f_c t).$$
 - Định nghĩa tín hiệu $s(t)$: $s(t) = s_I(t) - j \cdot s_Q(t)$, \rightarrow tín hiệu $s_P(t)$ có thể viết lại
$$s_P(t) = \sqrt{2} \cdot A \cdot \Re\{s(t) \cdot \exp(j2\pi f_c t)\}.$$
 - Tín hiệu $s(t)$:
 - Được gọi là tín hiệu tương đương băng gốc hoặc lớp vỏ phức của tín hiệu thông dải $s_P(t)$
 - Chứa cùng thông tin như $s_P(t)$
 - $s(t)$ là tín hiệu phức



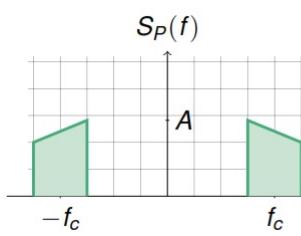
107

107



Giới thiệu

- Mô phỏng tín hiệu băng gốc và thông dải:**
 - Trong miền tần số:
$$S(f) = \begin{cases} \sqrt{2} \cdot S_P(f + f_c) & \text{for } f + f_c > 0 \\ 0 & \text{else.} \end{cases}$$
 - Hệ số $\sqrt{2}$ đảm bảo cả hai loại tín hiệu có cùng mức công suất.

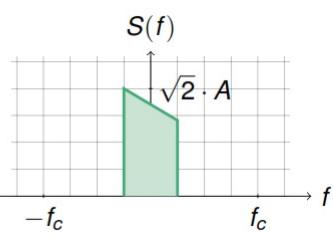


$S_P(f)$

f

$-f_c$ f_c

A



$S(f)$

f

$-f_c$ f_c

$\sqrt{2} \cdot A$



108

108

Giới thiệu

MATLAB

- Mô phỏng tín hiệu băng gốc và thông dài:
- Mô hình thông dài:

Passband model

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

109

109

109

Giới thiệu

MATLAB

- Mô phỏng tín hiệu băng gốc và thông dài:
- Mô hình tương đương thông thấp:

Baseband model

$$r_{RF}(t) = \text{Re}\{r(t) e^{j2\pi f_c t}\}$$

$$c_{RF}(t; t) = 2 \text{Re}\{c(t; t) e^{j2\pi f_c t}\}$$

$$s_{RF}(t) = \text{Re}\{s(t) e^{j2\pi f_c t}\}$$

$$n_{RF}(t) = \text{Re}\{n(t) e^{j2\pi f_c t}\}$$

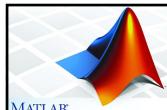
- Thu được hệ thống tương đương băng gốc:
sử dụng các tín hiệu băng gốc
- Tín hiệu phát tương đương băng gốc:
Kênh tương đương băng gốc với đáp ứng xung kim
giá trị phức $h(t)$ với $h_P(t) = \Re\{h(t) \cdot \exp(j2\pi f_c t)\}$
- Tín hiệu thu tương đương băng gốc: $R(t)$
- Nhiều Gaussian cộng giá trị phức: $N(t)$

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

110

110

110

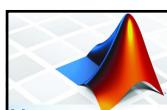


Giới thiệu

- *Mô phỏng tín hiệu bằng gốc và thông dải:*
 - Mô hình thông dải:
 - Các tín hiệu là thực
 - Sát với hệ thống thực
 - Tần số lấy mẫu cao hơn
 - Mô hình tương đương thông thấp:
 - Các tín hiệu là phức
 - Mô hình gọn và đơn giản hơn
 - Tần số lấy mẫu thấp hơn
 - Trong các trường hợp thực tế, xử lý tín hiệu số được thực hiện trên tín hiệu được chuyển đổi bằng gốc.
 - Mô hình tương đương bằng gốc là thuận tiện hơn trong mô phỏng hệ thống.
 - Hệ thống tuyến tính: $R(t) = s(t) * h(t) + n(t)$ and $R(f) = S(f) \cdot H(f) + N(f)$.

 111

111



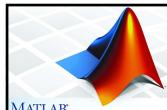
Giới thiệu

- *Quá trình lấy mẫu và nội suy:*
 - Trong mô phỏng hệ thống truyền tin trên hệ thống máy tính số đòi hỏi sự chuyển đổi mô hình thời gian liên tục thành mô hình rời rạc về thời gian.
 - Theo định lý lấy mẫu Nyquist (hoặc Shannon): nếu B_s là độ rộng băng tần của tín hiệu bằng gốc $s(t) \rightarrow$ tần số lấy mẫu $f_s \geq 2B_s$.
 - Quá trình lấy mẫu: $s(t) \rightarrow s_s(t) = s(nT_s)$

$$s_s(t) = s(t)p(t) \text{ với } p(t) = \sum_{n=-\infty}^{\infty} \delta(t-nT_s) \rightarrow s_s(t) = \sum_{n=-\infty}^{\infty} s(nT_s)\delta(t-nT_s)$$
 trong đó: T_s – chu kỳ lấy mẫu, $f_s = 1/T_s$ – tần số lấy mẫu
 - Tần số lấy mẫu được lựa chọn phù hợp để giảm thiểu lõi chồng phổ mà tránh tăng thời gian mô phỏng.

 112

112

 **Giới thiệu**

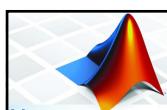
- **Quá trình lấy mẫu và nội suy:**
 - Trong một số trường hợp mô phỏng hệ thống trên các độ rộng băng tần khác nhau → chuyển đổi tốc độ mẫu
 - Tăng mẫu (upsampling): tại biên giữa phần tín hiệu băng hẹp và băng rộng $s(kT_s) \rightarrow s(kT_u) = s(kT_s/M)$
 - Giảm mẫu (downsampling): tại biên giữa phần tín hiệu băng rộng và băng hẹp $s(kT_s) \rightarrow s(kT_d) = s(kMT_s)$
 - Quá trình nội suy: quan trọng trong kỹ thuật đa tốc độ
 - Bộ nội suy hàm sinc
 - Bộ nội suy tuyến tính

Trong MATLAB sử dụng hàm *interp*:

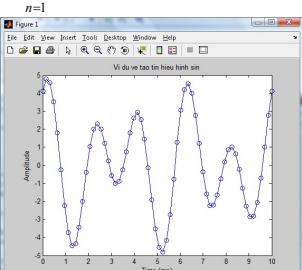
$y = interp(x,r)$ thực hiện lấy lại mẫu giá trị trong vectơ x tại r lần tốc độ lấy mẫu ban đầu.

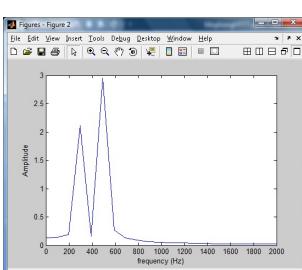
113

113

 **Mô phỏng nguồn tín hiệu**

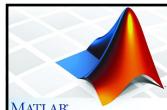
- **Nguồn tín hiệu tương tự:**
 - Tín hiệu đơn tần:
$$x(t_k) = A \cos(2\pi f_0 t_k + \varphi) \quad \text{hoặc} \quad \tilde{x}(k) = A \exp(2\pi j k f_0 / f_s) \exp(j\varphi)$$
- Tín hiệu đa tần: $x(t_k) = \sum_{n=1}^M x_n(t_k)$ với $x_n(t_k) = A_n \cos(2\pi f_n t_k + \varphi_n)$





114

114

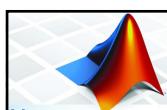
 **Mô phỏng nguồn tín hiệu**

- **Nguồn tín hiệu số:**
 - Nguồn thông tin rời rạc thường có giá trị trong bảng alphabet.
 - Tín hiệu số: là dạng sóng mang thông tin số.
 - Có 3 tham số chính:
 - Kiểu nguồn (alphabet): danh sách các ký hiệu thông tin có thể mà nguồn tạo ra.
 - VD: $A = \{0, 1\}$; các ký hiệu được gọi là bit
 - Với M ký hiệu ($M = 2^n$): $A = \{0, 1, \dots, M-1\}$ hoặc $A = \{\pm 1, \pm 3, \dots, \pm(M-1)\}$
 - Các symbol có thể có giá trị phức: $A = \{\pm 1, \pm j\}$
 - Xác suất ưu tiên phát: tần số xuất hiện tương đối của mỗi ký hiệu mà nguồn tạo ra.
 - VD: nguồn sinh ra các bit 0 và 1 có xác suất bằng nhau $\pi_0 = \pi_1 = \frac{1}{2}$.
 - Tốc độ ký hiệu (symbol rate): số lượng ký hiệu thông tin mà nguồn sinh ra trong một đơn vị thời gian (baud rate)

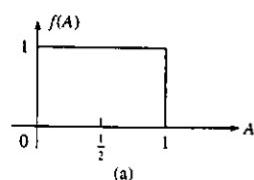
$$R_b = R \cdot \log_2(M).$$

 115

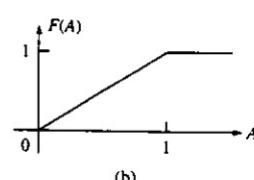
115

 **Mô phỏng nguồn tín hiệu**

- **Nguồn tín hiệu ngẫu nhiên:**
 - Các nguồn tin trong thực tế là ngẫu nhiên → tạo các tín hiệu ngẫu nhiên trong mô phỏng.
 - Tạo biến ngẫu nhiên phân bố đều: .



(a)



(b)

- Sử dụng hàm *rand* trong MATLAB

```
>> x = rand(5,10) - Tạo ma trận 5x10 các số ngẫu nhiên phân bố đều trong khoảng [0,1]
Tạo các số ngẫu nhiên phân bố đều trong khoảng [a, b] :
>> x = a + (b-a) * rand(m,n)
Tạo các số nguyên ngẫu nhiên phân bố đều trên tập 1:n :
>> x = ceil(n.*rand(100,1));
```

 116

116

Mô phỏng nguồn tín hiệu

- Nguồn tín hiệu ngẫu nhiên:**
 - Tạo biến ngẫu nhiên phân bố đều: .

Ví dụ: Tạo vector hàng 1000 số ngẫu nhiên phân bố đều trong khoảng [0,1], hiển thị 10 số đầu tiên

```
>> x = rand(1,1000);
>> x(1:10)
```

ans =

0.4330	0.8424	0.1845	0.5082	0.4522	0.3256	0.3801	0.8865	0.7613	0.8838
--------	--------	--------	--------	--------	--------	--------	--------	--------	--------

```
>> hist(x,10)
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

117

117

Mô phỏng nguồn tín hiệu

- Nguồn tín hiệu ngẫu nhiên:**
 - Tạo biến ngẫu nhiên phân bố chuẩn:

• Sử dụng hàm `randn` trong MATLAB .

Tạo các số ngẫu nhiên phân bố chuẩn có trung bình bằng 0 và độ lệch chuẩn bằng 1 :

```
>> x = randn(m,n)
```

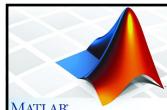
Tạo các số ngẫu nhiên phân bố chuẩn có trung bình bằng m và phương sai v :

```
>> x = m + sqrt(v) * randn(m,n)
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

118

118

 **Mô phỏng nguồn tín hiệu**

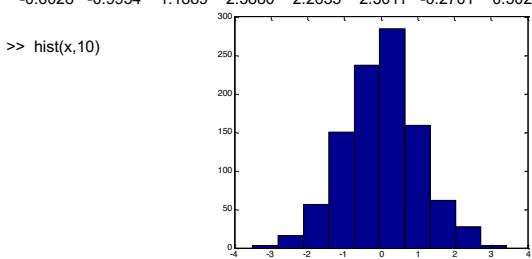
- *Nguồn tín hiệu ngẫu nhiên:*
 - Tạo biến ngẫu nhiên phân bố chuẩn:

Ví dụ: Tạo vector hàng 1000 số ngẫu nhiên phân bố chuẩn có trung bình 0 và độ lệch chuẩn bằng 1, hiển thị 10 số đầu tiên

```
>> x = randn(1,1000);
>> x(1:10)
```

ans =

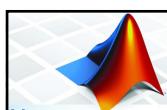
```
-0.6028 -0.9934 1.1889 2.3880 2.2655 2.3011 -0.2701 0.5028 -0.1192 -0.0019
```



HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

119

119

 **Mô phỏng nguồn tín hiệu**

- *Nguồn tín hiệu ngẫu nhiên:*
 - Tạo số nguyên ngẫu nhiên phân bố đều:
 - Sử dụng hàm *randint* trong MATLAB:

Tạo ma trận mxn các số 0 và 1 có xác suất bằng nhau

```
>> x = randint(m,n);
```

Ví dụ:

```
>> x = randint(1,10)
```

x =

```
0 0 1 1 1 0 1 1 0 0
```

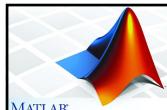
Tạo ma trận mxn có các giá trị phân bố đều trong dải từ 0 đến 7

```
>> x = randint(m, n, [0, 7]); hoặc
>> x = randint(m,n, 8);
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

120

120

 **Mô phỏng nguồn tín hiệu**

- **Nguồn tín hiệu ngẫu nhiên:**
 - Tạo symbol ngẫu nhiên theo danh sách alphabet định trước:
 - Sử dụng hàm `randsrc` trong MATLAB:

Tạo ma trận mxn các số -1 và 1 có xác suất bằng nhau
`>> x = randsrc(m,n);`
 Ví dụ:
`>> x = randsrc(1,10)`

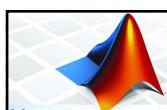
```
x =
-1  1  1  -1  -1  -1  -1  1  1  1
```

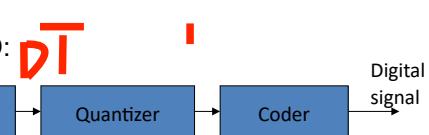
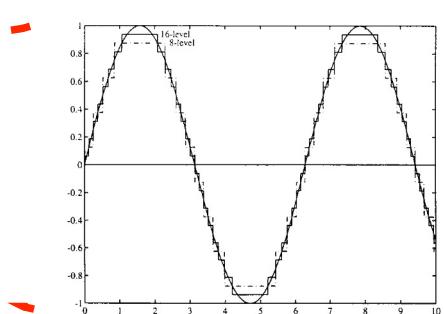
Tạo ma trận mxn có các giá trị phân bố đều trong tập {-3,-1,1,3}
`>> x = randsrc(10,10,[-3 -1 1 3]);` hoặc
`>> x = randsrc(10,10,[-3 -1 1 3; .25 .25 .25 .25]);`
- Tạo nguồn lõi ngẫu nhiên:
 - Sử dụng hàm `randerr` trong MATLAB



121

121

 **Mã hóa**

- **Mã hóa nguồn:**
 - Quá trình chuyển đổi A/D: 
 - Quá trình PCM:
 - Lượng tử hóa đều
 - Lượng tử hóa không đều

122

122

Mã hóa

• **Mã hóa nguồn:**

- MATLAB code cho quá trình PCM lượng tử hóa đều:

```

function [code,xq,sqnr] = uniform_pcm(x,M)
% Uniform PCM encoding of a sequence
% x = input sequence
% M = number of quantization levels
% code = the encoded output
% xq = quantized sequence before encoding
% sqnr = signal to quantization noise ratio in dB
% Written by Nguyen Duc Nhan - 2012

Nb = log2(M);
Amax = max(abs(x));
delta = 2*Amax/(M-1);
Mq = -Amax:delta:Amax;
Ml = 0:M-1;

xq = zeros(size(x));
xcode = xq;
for k = 1:M
    ind = find(x > Mq(k)-delta/2 & x <= Mq(k)+delta/2);
    xq(ind) = Mq(k);
    xcode(ind) = Ml(k);
end
sqnr = 20*log10(norm(x)/norm(x-xq)); % in dB
code = de2bi(xcode,Nb,'left-msb');

```

code =	1	1	0	1
1	1	1	0	
1	1	1	1	
1	1	1	1	
1	1	1	1	
1	1	1	1	

sqnr = 25.4136 dB

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

123

123

Mã hóa

• **Mã hóa nguồn:**

- Quá trình PCM lượng tử hóa không đều: Ví dụ theo luật μ

```

%% nonuniform PCM process
mu = 255;
M = 32; % number of quantization levels
[y,amax] = mulaw(x,mu); % compress the signal
[code,yq,sqnr] = uniform_pcm(y,M); % coding
xq = invmulaw(yq,mu); % expand the signal
xq = xq*amax;
sqnr = 20*log10(norm(x)/norm(x-xq)); % in dB

```

Tín hiệu lượng tử hóa sau khi nén

Ví dụ về nonuniform PCM

Ví dụ về nonuniform PCM

124

124

Mã hóa

- Mã đường:**
 - Kiểu mã hóa để tạo dạng phô và một số đặc tính xác định của xung tín hiệu hỗ trợ cho quá trình đồng bộ.
 - Gồm 2 bước:
 - Sắp xếp logic
 - Chuyển đổi thành dạng sóng
 - Ví dụ:

125

Mã hóa

- Mã đường:**
 - Tạo chuỗi xung vuông:
 - Trong mỗi chu kỳ xung, hàm xung vuông được định nghĩa như sau:

$$u(t) = \begin{cases} 1, & t \leq T_p \\ 0, & t > T_p \end{cases}$$

```

>> [t,y]=rectpulse(0.5e-6,1e6,256,8);
>> plot(t,y);

```

```

function [t,y] = rectpulse(Tw,Rp,Ns,Np)
% Chuong trinh vi du tao chuoi xung vuong
% Tw - the pulselwidth
% Rp - the repetition rate of pulse Tp < 1/Rp
% Ns - the number of samples
% Np - the number of pulses (the length of pulse train)
% t - the time vector output
% y - the vector output of the pulse samples
% written by Nguyen Duc Nhan

Tp = 1/Rp; % pulse period
Timewindow = Np*Tp; % time window
ts = Timewindow/(Ns-1); % sampling time
t = 0:ts:Timewindow; % time vector
Nsp = round(Tp/ts); % number of samples within Tp

y = zeros(size(t));
for k = 1:Ns
  if mod(t(k),Nsp*ts) <= Tw
    y(k) = 1;
  else
    y(k) = 0;
  end
end

```

126

Mã hóa

Mã đường:

- Mã NRZ:

```

function [t,y,code] = nrzcode(d,R,Ns,type)
% Chương trình vi du ve ma duong truyen NRZ
% d - the data sequence
% R - the data rate
% Ns - the number of samples
% t - the time vector output
% y - the vector output of the pulse samples
% type - the type of code (unipolar - 'unipol' or polar - 'pol')
% written by Nguyen Duc Nhan

Tb = 1/R; % bit period
Nb = length(d); % number of bits
Timewindow = Nb*Tb; % time window
ts = Timewindow/(Ns-1); % sampling time
t = 0:ts:Timewindow; % time vector
y = zeros(size(t));
code = [];

if nargin <=3
    type = 'unipol';
end
for k = 1:Ns
    n = fix(t(k)/Tb)+1;
    if n >= Nb
        n = Nb;
    end
    switch (type)
        case 'unipol'
            y(k) = d(n);
            code(n) = d(n);
        case 'pol'
            y(k) = 2*d(n)-1;
            code(n) = 2*d(n)-1;
    end
end

```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

127

127

Mã hóa

Mã đường:

- Mã AMI:

```

function [t,y,code] = amicode(d,R,Ns,type)
% Chương trình vi du ve ma AMI
% d - the data sequence
% R - the data rate
% Ns - the number of samples
% t - the time vector output
% y - the vector output of the pulse samples
% type - the type of code (NRZ - 'NRZ' or RZ - 'RZ')
% written by Nguyen Duc Nhan
...
y = zeros(size(t));
code = [];
...

```

```

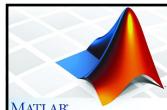
s = 1;
for k = 1:Nb
    if d(k) == 0
        code(k) = 0;
    else
        s = s+1;
        if mod(s,2)==0
            code(k) = 1;
        else
            code(k) = -1;
        end
    end
end

```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

128

128



Mã hóa

- **Mã hóa kênh:**
 - Tăng hiệu năng của kênh truyền:
 - Phát hiện lỗi
 - Sửa lỗi
 - Gồm 2 loại chính:
 - Mã khồi
 - Mã xoắn
 - Ví dụ mã khồi:
 - Mã hóa: Từ mã $c = uG$

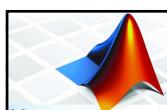
Trong đó: u – chuỗi dữ liệu có k bit, G – ma trận tạo mã có $k \times n$, c – từ mã được mã hóa có n bit ($n > k$)

- Khoảng cách Hamming tối thiểu: $d_{\min} = \min_{i \neq j} d_H(c_i, c_j)$
- Đối với mã khồi tuyến tính: khoảng cách tối thiểu bằng với trọng số nhỏ nhất của mā

$$w_{\min} = \min_{c_i \neq 0} w(c_i)$$

129

129



Mã hóa

- **Mã hóa kênh:**
 - Ví dụ mã khồi:

```
% Chương trình vi du ve ma hoa kenh
% Block coding
k = 4;
for i=1:2^4
    for j=k:-1:1
        if rem(i-1,2^(j+k+1))>=2^(j+k)
            u(i,j)=1;
        else
            u(i,j)=0;
        end
    end
end
% Define G, the generator matrix
g =[1 0 0 1 1 0 1 1 1;
     1 1 0 0 0 1 1 0;
     0 1 1 0 1 1 0 1 0;
     1 1 0 1 1 0 1 0 1;
     1 1 0 1 1 1 0 0 1];
% generate codewords
c = rem(u*g,2);
% find the minimum distance
w_min = min(sum((c(2:2^k,:))'));

```

u =	c =
0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1	1 1 0 1 1 1 1 0 0 0 1
0 0 1 0	0 1 1 0 1 1 0 1 0 1 0 1 0 1
0 0 1 1	1 0 1 1 0 0 1 1 1 0 1 0 0 0 0
0 1 0 0	0 1 1 0 0 0 1 1 1 0 1 1 0 1 0
0 1 0 1	0 0 1 1 1 1 0 1 0 1 1 1 1 1 1
0 1 1 0	1 0 0 0 0 1 1 1 1 0 1 0 1 1 1
0 1 1 1	0 1 0 1 0 0 0 0 0 1 0 1 0 1 0
1 0 0 0	1 0 0 0 1 1 1 0 1 1 1 1 1 1 1
1 0 0 1	0 1 0 0 0 0 0 1 1 1 1 1 1 0 0
1 0 1 0	1 1 1 1 0 0 0 0 0 1 1 1 0 1 0
1 0 1 1	0 0 1 0 1 1 1 1 1 0 1 1 0 1 1
1 1 0 0	0 1 1 1 1 1 1 0 0 1 1 1 0 0 1
1 1 0 1	1 0 1 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 0	0 0 0 1 0 0 0 1 1 0 1 0 0 0 0
1 1 1 1	1 1 0 1 0 1 1 0 1 0 1 0 1 0 1

130

130

MATLAB

Điều chế và giải điều chế

- **Điều chế tín hiệu tương tự:**
 - Điều chế biên độ AM:
 - Tín hiệu bản tin: $m(t) = M \cdot \cos(\omega_m t + \phi)$.
 - Tín hiệu sóng mang: $c(t) = A \cdot \sin(\omega_c t + \phi_c)$,
 - Quá trình điều chế: là một quá trình nhân
$$y(t) = [1 + m(t)] \cdot c(t),$$

$$= A \cdot [1 + M \cdot \cos(\omega_m t + \phi)] \cdot \sin(\omega_c t).$$
 - Độ sâu điều chế (modulation index):

$$h = \frac{\text{peak value of } m(t)}{A} = \frac{M}{A},$$

$$m(t) \xrightarrow{\mathcal{F}} M(\omega)$$

$$\sin(\omega_c t) \xrightarrow{\mathcal{F}} i\pi \cdot [\delta(\omega + \omega_c) - \delta(\omega - \omega_c)]$$

$$A \cdot \sin(\omega_c t) \xrightarrow{\mathcal{F}} i\pi A \cdot [\delta(\omega + \omega_c) - \delta(\omega - \omega_c)]$$

$$m(t) \cdot A \sin(\omega_c t) \xrightarrow{\mathcal{F}} \frac{A}{2\pi} \cdot \{M(\omega)\} * \{i\pi \cdot [\delta(\omega + \omega_c) - \delta(\omega - \omega_c)]\}$$

$$= \frac{iA}{2} \cdot [M(\omega + \omega_c) - M(\omega - \omega_c)]$$

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

131

131

MATLAB

Điều chế và giải điều chế

- **Điều chế tín hiệu tương tự:**
 - Điều chế biên độ AM:
 - Vi du ve dieu che AM
 - M = 50%
 - M = 100%

```
% Chuong trinh vi du ve dieu che AM
% Set parameters
% Message
A = 1; % amplitude
f = 440; % frequency [Hz]
phi = -pi/4; % Phase [rad]

% Carrier
m = 0.5; % modulation index
Ac = A/m; % amplitude
fc = 5e3; % frequency [Hz]
phi_c = 0; % Phase [rad]

N = 2^9; % number of samples
T0 = 0; % start time [s]
Tf = 5e-3; % end time [s]
Ts = (Tf-T0)/(N-1); % sampling period
fs = 1/Ts; % sampling frequency [Hz]

%% Amplitude Modulation
% Generate sinusoid
t = T0:Ts:Tf; % time vector
x = A*cos(2*pi*f*t+phi); % message signal
xc = Ac*cos(2*pi*fc*t+phi_c); % carrier signal

% Modulation
y = (1+x/Ac).*xc;
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

132

132

MATLAB

Điều chế và giải điều chế

- **Điều chế tín hiệu tương tự:**
 - Điều chế biên độ AM:
 - Điều chế SSB (Single side band): Sử dụng bộ lọc hoặc dùng khai triển Hilbert
$$\begin{aligned} s_{ssb}(t) &= s(t) \cdot \cos(2\pi f_0 t) - \hat{s}(t) \cdot \sin(2\pi f_0 t), \\ s_{ssb}(t) &= Re\{s_a(t) \cdot e^{j2\pi f_0 t}\} \\ &= Re\{[s(t) + j \cdot \hat{s}(t)] \cdot [\cos(2\pi f_0 t) + j \cdot \sin(2\pi f_0 t)]\} \\ &= s(t) \cdot \cos(2\pi f_0 t) - \hat{s}(t) \cdot \sin(2\pi f_0 t). \\ s_{lsb}(t) &= Re\{s_a^*(t) \cdot e^{j2\pi f_0 t}\} \\ &= s(t) \cdot \cos(2\pi f_0 t) + \hat{s}(t) \cdot \sin(2\pi f_0 t). \end{aligned}$$

```
%% SSB Modulation
% Generate sinusoid
t = T0:Ts:Tf;
x = A*cos(2*pi*f*t+phi);
% Modulation
y = ssbmod(x,fc,fs,phic);

% Demodulation
xr = ssbdemod(y,fc,fs,phic);
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

133

MATLAB

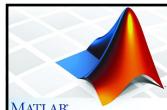
Điều chế và giải điều chế

- **Điều chế tín hiệu tương tự:**
 - Điều chế tần số FM:
 - Trong MATLAB: sử dụng hàm `fmod` và `fmdemod` cho điều chế FM

```
% Modulation
y = pmmod(x,fc,fs,phic);
% Demodulation
xr = pmdemod(y,fc,fs,phic);
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

134

 **Điều chế và giải điều chế**

- **Điều chế tín hiệu tương tự:**
 - Điều chế pha PM:
 - Sóng mang: $c(t) = A_c \sin(\omega_c t + \phi_c)$.
 - Tín hiệu điều chế: $y(t) = A_c \sin(\omega_c t + m(t) + \phi_c)$.

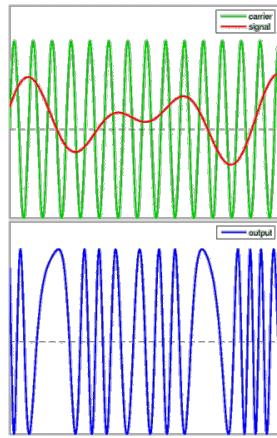
Chỉ số điều chế: $2(h+1)f_M$

$$f_M = \omega_m / 2\pi$$

$$h = \Delta\theta$$

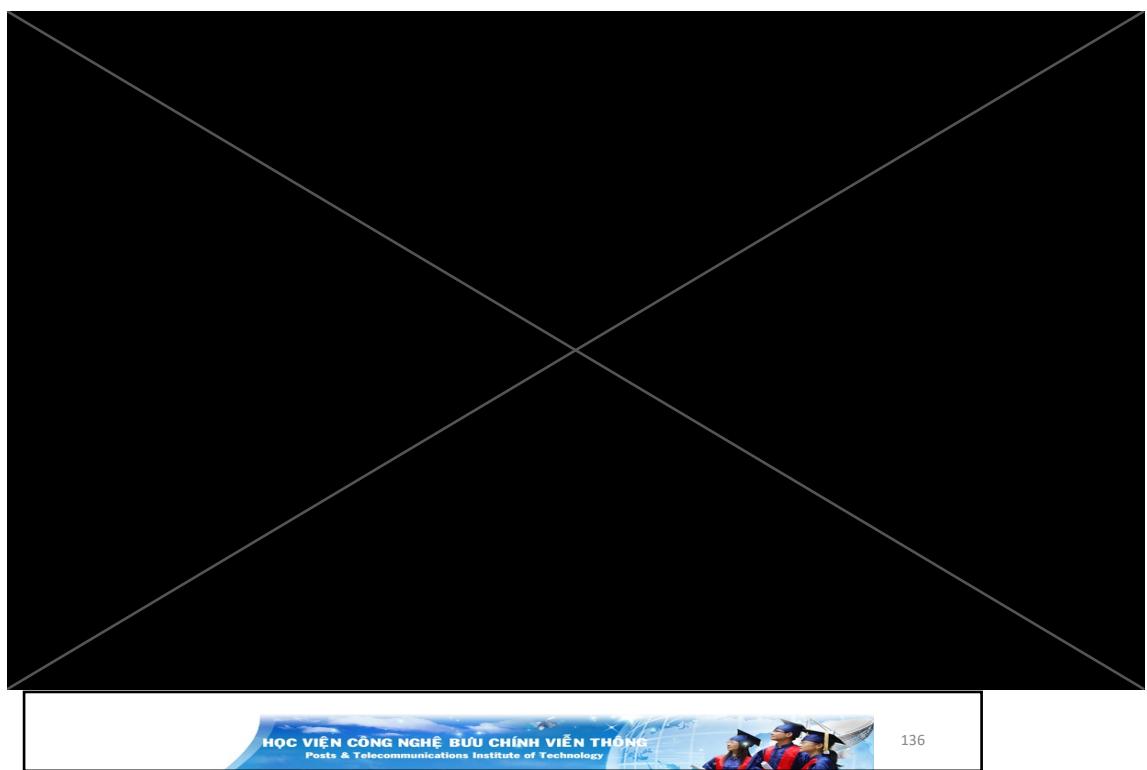
• Trong MATLAB sử dụng các hàm *pmmod* và *pmdemod* cho điều chế PM.

```
%% Phase Modulation
t = T0:Ts:Tf;
x = A*cos(2*pi*f*t+phi);
% Modulation
y = pmmod(x,fc,fs,phic);
% Demodulation
xr = pmdemod(y,fc,fs,phic);
```



HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

135



Điều chế và giải điều chế

- Điều chế tín hiệu số:
 - Điều chế biên độ ASK:
 - Sử dụng hàm `pammod` và `pamdemod` trong MATLAB → tạo ra ký hiệu phức tương đương bằng gốc.

b_n p(t)

m(t)

```
% Signal generator
dm = randint(1,1000,4);
% PAM modulation
s = pammod(dm,4);
% PAM demodulation
r = pamdemod(s,4);
```

Để biểu diễn dạng sóng điều chế có sóng mang có thể sử dụng hàm `ammod` trong điều chế AM với các mức được lấy mẫu:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

137

137

Điều chế và giải điều chế

- Điều chế tín hiệu số:
 - Điều chế pha PSK:
 - Điều chế M-PSK:

Bộ điều chế

Bộ giải điều chế

b_n = 1/2

Tín hiệu BPSK:
 $s_n(t) = \sqrt{\frac{2E_b}{T_b}} \cos(2\pi f_c t + \pi(1-n)), n=1, 0.$

Tín hiệu QPSK:
 $s_n(t) = \sqrt{\frac{2E_s}{T_s}} \cos\left(2\pi f_c t + (2n-1)\frac{\pi}{4}\right), n=1, 2, 3, 4.$

Dạng sóng QPSK:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

138

Điều chế và giải điều chế

• Điều chế tín hiệu số:

- Điều chế pha PSK:
 - Trong MATLAB sử dụng hàm `pskmod` và `pskdemod` cho điều chế PSK kết hợp với `modem` để xây dựng object của bộ điều chế và giải điều chế:

```
% Create a random digital message
M = 4; % Alphabet size
x = randint(5000,1,M); % Message generator
% Use QPSK modulation to produce y.
h = modem.pskmod(M,pi/4);
h.symbolorder = 'gray';
y = modulate(h,x);

% Transmit signal through an AWGN channel.
ynoisy = awgn(y,15,'measured');

% Create scatter plot from noisy data.
h = scatterplot(ynoisy,1,0,'xb');
hold on;
scatterplot(y,1,0,'or',h);
% Demodulate ynoisy to recover the message.
h = modem.pskdemod(M,pi/4);
h.symbolorder = 'gray';
z= demodulate(h,ynoisy);
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

139

Điều chế và giải điều chế

• Điều chế tín hiệu số:

- Điều chế pha PSK:
 - Điều chế PSK mã hóa vi sai: sử dụng hàm `dpskmod` và `dpskdemod`

```
% Create a random digital message
M = 2; % Alphabet size
x = randint(5000,1,M); % Message
% Use DPSK modulation to produce y.
y = dpskmod(x,M);

% Demodulate to recover the message.
z = dpskdemod(y,M);
```

Bieu dien bien do cua tin hieu dieu che DPSK

Bieu dien bien do cua tin hieu dieu che DPSK

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

140

MATLAB

Điều chế và giải điều chế

- **Điều chế tín hiệu số:**
 - Điều chế QAM: kết hợp giữa ASK và PSK

Bộ điều chế

Bộ giải điều chế

Rectangular QAM

Circular QAM

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

141

141

MATLAB

Điều chế và giải điều chế

- **Điều chế tín hiệu số:**
 - Điều chế QAM:
 - Sử dụng hàm `qammod` và `qamdemod` trong MATLAB

```

close all;
% Create a random digital message
M = 16; % Alphabet size
x = randint(5000,1,M);

% Use 16-QAM modulation to produce y.
y=modulate(modem.qammod(M),x);

% Transmit signal through an AWGN channel.
ynoisy = awgn(y,15,'measured');

% Create scatter plot from noisy data.
h = scatterplot(ynoisy,1,0,'xb');
hold on;
scatterplot(y,1,0,'or',h); hold off;

```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

142

142

Điều chế và giải điều chế

- **Điều chế tín hiệu số:**
 - Điều chế FSK:
 - Điều chế FSK
 - Điều chế FSK pha liên tục (CPFSK)
 - Điều chế MSK: một kiểu CPFSK

• Sử dụng hàm `fskmod` và `fskdemod` cho điều chế FSK
 • Sử dụng hàm `mskmod` và `fskdemod` cho điều chế MSK

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

143

143

Điều chế và giải điều chế

- **Điều chế tín hiệu số:**
 - Điều chế FSK:

```
% Chuong trinh vi du ve MSK
close all;
% Parameters
Ns = 8; % number of samples per symbol
x = randint(1000,1); % Random signal
% Use MSK modulation to produce y.
y = mskmod(x,Ns,1,pi/2);
h = scatterplot(y,1,0,'xb');
hold on;
scatterplot(y,Ns,0,'or',h);
hold off;
% Transmit signal through an AWGN channel.
yn = awgn(y,25,'measured');
% Plot eyediagram
eyediagram(yn,16);
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

144

144

 **Quá trình lọc**

- **Tạo dạng phổ:**
 - Mật độ phổ công suất $S_Y(f)$ của tín hiệu ra $y(t)$:

$$S_Y(f) = S_X(f)|H(f)|^2$$

Trong đó $S_X(f)$ là PSD của tín hiệu vào $x(t)$ và $H(f)$ là hàm truyền của hệ thống (hay bộ lọc)
 - Bằng việc lựa chọn cản thận $H(f) \rightarrow$ tăng cường hoặc khử các thành phần phổ chọn lọc của tín hiệu vào.
 - Khi $S_X(f)$ và $H(f)$ xác định \rightarrow xác định được $S_Y(f)$
 - Ngược lại: biết $S_Y(f)$ là PSD đầu ra mong muốn của PSD đầu vào $S_X(f) \rightarrow$ xác định được $H(f)$
 - Ví dụ bộ lọc butterworth:

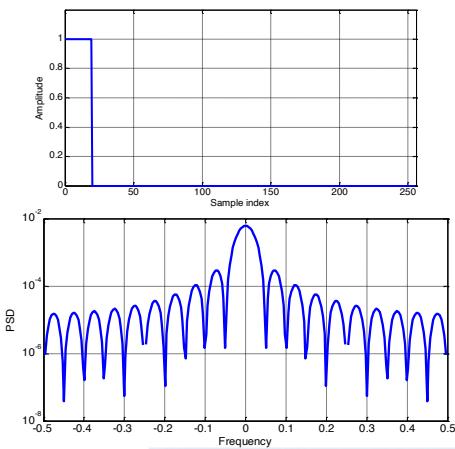
$$|H(f)|^2 = \frac{1}{1 + (\omega / \omega_b)^{2n}} \quad n - \text{bậc của bộ lọc}$$

$$\omega_b - \text{độ rộng băng tần 3 dB}$$

 145

145

 **Quá trình lọc**

- **Tạo dạng phổ:**
 - Tính toán PSD của một tín hiệu:


$S_X(f) \propto U$

```
function [f,Pf] = spectrocal(t,x)
% Ví dụ chương trình tính toán spectrum
% t - time vector
% x - input samples
% f - frequency vector
% Pf - estimated PSD of x
% written by Nguyen Duc Nhan

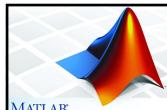
Ns = length(x);
Ts = t(2)-t(1);

f = (-Ns/2:Ns/2-1)/(Ns*Ts); % freq. vector
Pf = fft(x,Ns);
Pf = fftshift(Pf)/Ns;
Pf = abs(Pf).^2;
```

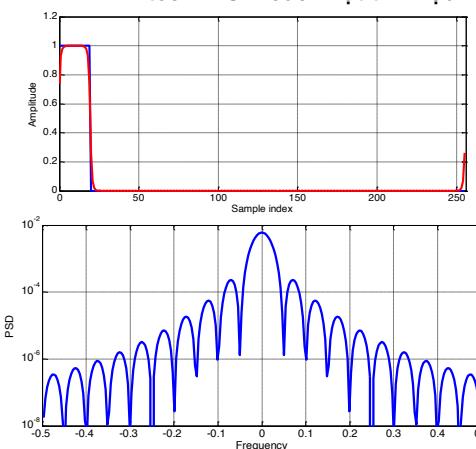
```
>> t = 0:255;
>> x = zeros(1,length(t));
>> x(1:20) = 1;
>> plot(t,x);
>> [f,Xf] = spectrocal(t,x);
>> semilogy(f,Xf);
```

 146

146

 **Quá trình lọc**

- **Tạo dạng phô:**
 - Tính toán PSD của một tín hiệu:



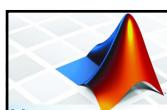
```
function y = butterwfil(x,n,B,Ts)
% Function bo loc butterworth
% B - filter bandwidth
% Ts - sampling time
% n - filter order
% y - filtered output
Ns = length(x);
% Frequency domain
f = [0:Ns/2-1 -Ns/2:-1]/(Ns*Ts);
Xf = fft(x);
Hf = 1./(1+(f./B).^(2*n)); % transfer func.
Yf = Xf.*Hf;
% Convert into time domain
y = ifft(Yf);
```

```
>> y = butterwfil(x,1,0.2,1);
>> [f,Xf]=spectrocal(t,y);
>> semilogy(f,Xf); grid;
>> figure(2);
>> plot(t,x,t,y); grid;
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

147

147

 **Quá trình lọc**

- **Tạo dạng xung:**
 - Mỗi xung đơn dạng sóng trong các hệ thống truyền dẫn số thường được yêu cầu thỏa mãn 2 điều kiện quan trọng về:
 - Băng thông
 - Chuyển tiếp qua 0 (zero crossings)
 - Nyquist đã chứng minh rằng: một xung $p(t)$ có zero crossing mỗi chu kỳ $T_b = 1/R_b$ nếu khai triển $P(f)$ đáp ứng các ràng buộc sau:

$$\sum_{k=-\infty}^{\infty} P(f+kR_b) = T_b \quad |f| < R_b/2$$

- Một họ $P(f)$ đáp ứng tiêu chuẩn Nyquist là họ *raised cosine*:
$$P(f) = \begin{cases} T_b, & |f| \leq R_b/2 - \beta \\ T_b \cos^2 \frac{\pi}{4\beta} \left(|f| - \frac{R_b}{2} + \beta \right), & R_b/2 - \beta < |f| \leq R_b/2 + \beta \\ 0, & |f| > R_b/2 + \beta \end{cases}$$

Trong đó β là tham số băng tần trội (hệ số rolloff) và $P(f)$ bị giới hạn bằng tới $\beta + (R_b/2)$.

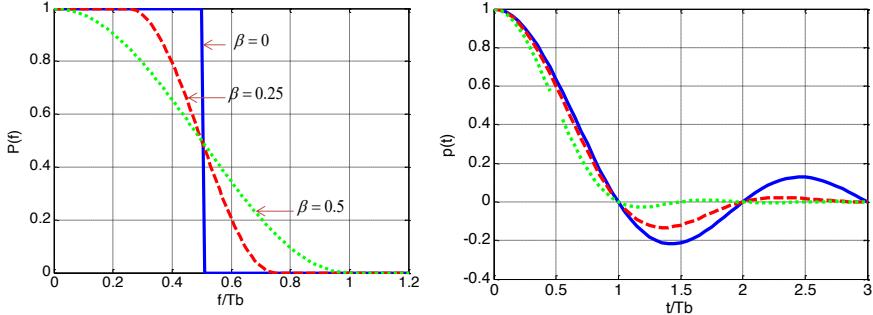
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

148

148

 **Quá trình lọc**

- **Tạo dạng xung:**
 - Đáp ứng xung kim của họ *raised cosine*:

$$p(t) = \frac{\cos 2\pi \beta t}{1 - (4\beta t)^2} \left(\frac{\sin \pi R_b t}{\pi R_b t} \right)$$


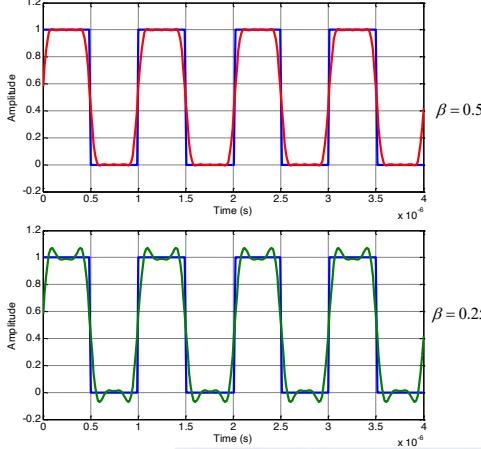
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

149

149

 **Quá trình lọc**

- **Tạo dạng xung:**
 - Ví dụ bộ lọc *raised cosine*:

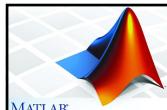


```
function y = raisedcosf(x,Rb,Ts,beta)
% Function for raised cosine filter
% x - input samples
% Rb - filter bandwidth
% Ts - sampling time
% beta - rolloff factor
% y - filtered output
Ns = length(x);
Tb = 1/Rb;
beta = beta*Tb;
% Frequency domain
f = [0:Ns/2-1 -Ns/2:-1]/(Ns*Ts);
Xf = fft(x);
Yf = zeros(size(Xf));
ind = (abs(f)<=(Rb/2-beta));
Yf(ind) = Xf(ind).*Tb;
ind = (abs(f)<=(Rb/2+beta)&abs(f)>(Rb/2-beta));
Yf(ind) = Xf(ind).*(Tb*cos(pi/(4*beta)*...
(abs(f(ind)-Rb/2+beta)).^2);
ind = (abs(f)>(Rb/2+beta));
Yf(ind) = Xf(ind).*0;
% Convert into time domain
y = ifft(Yf)/Tb;
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

150

150



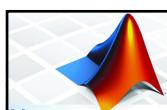
Quá trình lọc

- Tạo dạng xung:**
 - Trong MATLAB có thể sử dụng các hàm cho quá trình lọc trong signal processing toolbox:
 - Ví dụ sử dụng hàm filter:
$$y = \text{filter}(b,a,x) \quad \text{Trong đó } a, b \text{ là các vector chứa các hệ số của bộ lọc}$$

$$y(n) = b(1)*x(n) + b(2)*x(n-1) + \dots + b(nb+1)*x(n-nb) - a(2)*y(n-1) - \dots - a(na+1)*y(n-na)$$
- Một số kiểu bộ lọc sẵn có:
 - Bộ lọc butterworth: $[b,a] = \text{butter}(n,Wn)$
 - Bộ lọc bessel: $[b,a] = \text{besself}(n,Wo)$
 - Bộ lọc chebyshev: $[b,a] = \text{cheby1}(n,R,Wp)$ và $[b,a] = \text{cheby2}(n,R,Wst)$
 - Bộ lọc raised cosine: $y = \text{rcosfit}(x,Fd,Fs)$

151

151



Quá trình lọc

- Bộ lọc phối hợp (matched filters):**
 - Xét bài toán thu được hoặc không thu được một xung có dạng $p(t)$ từ tín hiệu quan sát $y(t)$:
$$y(t) = \begin{cases} p(t) + N(t) \\ \text{or} \\ N(t) \end{cases} \quad N(t) - \text{nhiều cộng có trung bình 0.}$$

Xử lý $y(t)$ để quyết định có hay không có $p(t)$ trong khoảng chu kỳ T .

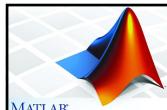
 - Xử lý $y(t)$ thu được: $z = \int_0^T y(\tau)h(T-\tau)d\tau$
 - Xác suất lỗi trung bình nhỏ nhất khi bộ lọc có hàm truyền:
$$H(f) = KP^*(f) \exp(j2\pi fT)$$

→ đáp ứng xung: $h(t) = p(T-t) \quad \rightarrow z = \int_0^T y(\tau)h(T-\tau)d\tau = \int_0^T y(\tau)p(\tau)d\tau$

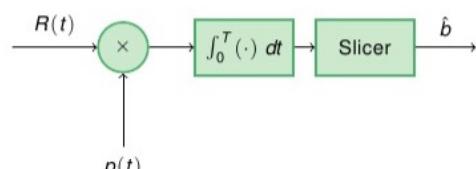
Đáp ứng xung phối hợp với $p(t)$.

152

152

 **Quá trình lọc**

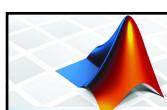
- **Bộ lọc phối hợp (matched filters):**
 - Bộ lọc phối hợp cho một tín hiệu điều chế tuyến tính sử dụng dạng xung $p(t)$:



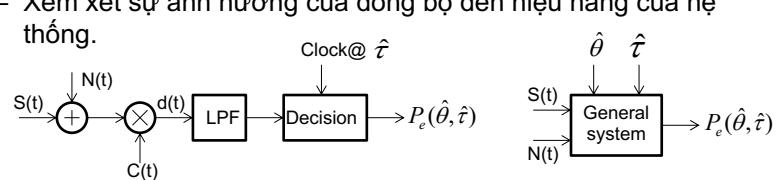
- Khi $p(t)$ có dạng xung vuông đơn vị \rightarrow z là tích phân của tín hiệu đầu vào.
- Quá trình lấy tích phân được bắt đầu thực hiện tại đầu mỗi khoảng chu kỳ \rightarrow bộ lọc integrate-and-dump (I&D).

153

153

 **Quá trình đồng bộ**

- **Quá trình đồng bộ trong mô phỏng:**
 - Xem xét sự ảnh hưởng của đồng bộ đến hiệu năng của hệ thống.
 - Có các cách tiếp cận khác nhau:
 - Dạng cấu trúc
 - Dạng mô tả thống kê
 - Tại bộ thu, quá trình đồng bộ bao gồm:
 - Khôi phục sóng mang
 - Khôi phục đồng hồ (tín hiệu định thời)



154

154

Quá trình đồng bộ

MATLAB

- Mô phỏng mạch vòng khóa pha PLL:**
 - PLL được mô tả bởi ptr. vi phân phi tuyến
 - Bộ lọc vòng bậc 2:

The diagram illustrates a PLL system. It starts with a reference signal $\sqrt{2}A \cos[2\pi f_c t + \theta(t)] + N(t)$ entering a multiplier. The output of the multiplier goes to a 'Loop filter' block labeled $K_1 F(s)$. The output of the filter is fed into a 'VCO' block labeled $K_2 (\text{rad/s})/\text{volt}$. The VCO's output is a sine wave $\sqrt{2K_2} \sin[2\pi f_c t + \hat{\theta}(t)]$. This signal is also fed back through a low-pass filter (represented by a $\frac{1}{s}$ block) to the 'Loop filter'. Below the main loop, there is a 'Mô tả tương đương trong miền pha' (Phase-domain equivalent model). It shows the error signal $e(t) = A \sin(\phi(t))$ being fed into a $\frac{v(t)}{K_1 K_3 F(s)}$ block. The output of this block is $v(t)$, which is fed back to the VCO. The error signal $e(t)$ is also used to calculate the derivative $\dot{e}(t)$, which is then used to update the estimated phase $\hat{\theta}(t)$ via the equation $\dot{\hat{\theta}}(t) = \frac{\alpha_1}{\tau_1} \dot{e}(t)$.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

155

155

Quá trình đồng bộ

MATLAB

- Mô phỏng mạch vòng khóa pha PLL:**
 - Hàm truyền bộ lọc vòng:

$$F(s) = \frac{s\tau_2 + 1}{s\tau_1 + \alpha_1} \quad \text{Với} \quad \alpha_1 = \begin{cases} 1 & \text{Bộ lọc thụ động} \\ 0 & \text{Bộ lọc tích cực} \end{cases}$$

- PLL mở rộng:**

$$\begin{aligned} v(t) &= (K_1 K_3 / \tau_1) [\tau_2 \ddot{y}(t) + \dot{y}(t)] \\ \hat{\theta}(t) &= (K_1 K_2 K_3 / \tau_1) [\tau_2 \dot{y}(t) + y(t)] \\ \phi(t) &= \theta(t) - \hat{\theta}(t) = \theta(t) - (K_1 K_2 K_3 / \tau_1) [\tau_2 \dot{y}(t) + y(t)] \\ e(t) &= A \sin \{ \theta(t) - (K_1 K_2 K_3 / \tau_1) [\tau_2 \dot{y}(t) + y(t)] \} \\ \ddot{y}(t) &= e(t) - (\alpha_1 / \tau_1) \dot{y}(t) \end{aligned}$$

Đặt các hằng số:

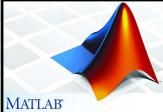
$$c_1 = \frac{-K_1 K_2 K_3 \tau_2}{\tau_1} \quad c_2 = \frac{-K_1 K_2 K_3}{\tau_1} \quad c_3 = -\frac{\alpha_1}{\tau_1}$$

$$\rightarrow \ddot{y}(t) = A \sin [\theta(t) + c_1 \dot{y}(t) + c_2 y(t) + c_3 \dot{y}(t)]$$

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

156

156

 **Bài tập chương 4**

31. Viết chương trình MATLAB tạo chuỗi bit ngẫu nhiên phân bố đều có độ dài 128 bit. Sau đó thực hiện chuyển đổi chuỗi bit này thành các giá trị thập phân trong dải từ [0,15].
 - Ghi ý: Chuyển đổi vector hàng thành ma trận mx4, sau đó dùng hàm *bi2de* để chuyển đổi sang dạng thập phân.

32. Xây dựng các *function* để nén và giải nén tín hiệu theo luật A có dạng cấu trúc sau:

```
function [y,amax] = alaw(x,A) và function x = invalaw(y,A)
```

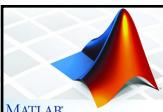
33. Viết chương trình mã hóa tín hiệu $x = 2\cos(4\pi t)$ tại tần số lấy mẫu $f_s = 20$ Hz sử dụng quá trình PCM theo luật A với 8 mức lượng tử. Hãy xác định từ mã đầu ra của 5 mẫu đầu tiên. Vẽ biểu diễn tín hiệu gốc ban đầu, tín hiệu được lấy mẫu và tín hiệu được lượng tử hóa trên cùng một hình. (Sử dụng các *function* đã xây dựng của bài tập trên)

34. Xây dựng *function* để chuyển đổi chuỗi bit dữ liệu đầu vào thành mã đường RZ đơn cực.
 Sử dụng *function* này để mã hóa RZ và biểu diễn dạng sóng của chuỗi bit [0 1 1 0 1 0 1 0] tại tốc độ 1 Mbit/s.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

157

157

 **Bài tập chương 4**

35. Viết chương trình MATLAB xác định các từ mã đầu ra của bộ mã hóa khồi có tốc độ 4/7. Biết ma trận tạo mã có dạng:

$$\begin{matrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{matrix}$$

36. Viết *function* để tạo chuỗi xung tam giác đầu ra. Biết trong một chu kỳ dạng xung được biểu diễn bởi hàm sau:

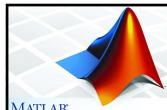
$$p(t) = \begin{cases} 1 - |(t - T_w)/T_w|, & 0 \leq t \leq T_p \\ 0, & t \text{ khác} \end{cases} \quad \text{Với } T_w = T_p/2 \quad T_p - \text{chu kỳ xung}$$

37. Cho tín hiệu tương tự được mô tả bởi công thức sau:
 $s(t) = 2\cos(20\pi t + \pi/4) + \cos(30\pi t)$
 Viết chương trình thực hiện điều chế biến độ tín hiệu bằng sóng mang $f_c = 300$ Hz. Vẽ dạng sóng tín hiệu bản tin ban đầu và tín hiệu được điều chế.
 Giải điều chế tín hiệu trên bằng kỹ thuật phù hợp và vẽ dạng sóng tín hiệu sau khi được giải điều chế.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

158

158



Bài tập chương 4

38. Tạo chuỗi bit ngẫu nhiên có độ dài 5000 bits. Chuyển đổi chuỗi bit này thành dạng sóng mã đường NRZ lưỡng cực tại tốc độ 100 Mbit/s. Sử dụng bộ lọc *raised cosine* có độ rộng băng tần 300 MHz và hệ số rolloff bằng 0.5 để lọc chuỗi tín hiệu NRZ này. Vẽ biểu diễn dạng sóng tín hiệu trên 10 chu kỳ bit trước và sau khi lọc tín hiệu. Vẽ biểu đồ mắt của tín hiệu sau khi lọc trên cửa sổ 2 chu kỳ bit. (Sử dụng hàm *eyediagram* để vẽ mẫu mắt).

39. Tương tự bài tập 38, sử dụng bộ lọc *butterworth* có tần số cắt khoảng 250 MHz để lọc tín hiệu. (Sử dụng các hàm *butter* và *filter* trong signal processing toolbox).

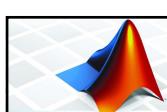
40. Tạo chuỗi ký tự ngẫu nhiên có độ dài 1000 ký tự để thực hiện điều biến BPSK. Hãy viết chương trình biểu diễn dạng sóng đường bao phức của tín hiệu điều biến BPSK tại tốc độ 10 Mb/s bởi các xung raised cosine được mô tả bởi hàm sau:

$$p(t) = \left(1 - \cos\left(\frac{2\pi t}{T}\right)\right) \quad 0 \leq t \leq T$$

Vẽ dạng phô của tín hiệu được điều biến.



159

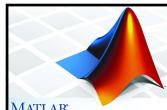


Mô phỏng kênh thông tin

- **Giới thiệu**
- **Kênh AWGN**
- **Một số mô hình kênh thông tin**



160

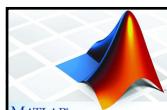


Giới thiệu

- **Mô phỏng kênh thông tin:**
 - Mô hình kênh: mô tả sự suy giảm tín hiệu phát trải qua trên đường truyền tới bộ thu.
 - Mô hình kênh được xác định từ:
 - Đo đặc thực nghiệm
 - Lý thuyết truyền sóng trong môi trường vật lý
 - Các kênh thông tin:
 - Hữu tuyến
 - Vô tuyến
 - Các mô hình mô phỏng:
 - Mô hình hàm truyền đạt cho kênh bất biến theo thời gian
 - Mô hình đường trễ rẽ nhánh cho kênh biến đổi theo thời gian

 161

161

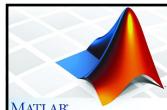


Kênh AWGN

- **Nhiễu AWGN:**
 - AWGN: Additive White Gaussian Noise – Nhiễu Gauss trắng cộng.
 - Trong hầu hết các hệ thống thông tin: nhiễu được mô tả như là AWGN
 - **Tính cộng:** kênh cộng thêm nhiễu vào tín hiệu được phát đi
 - **Trắng:** mô tả tương quan thời gian của nhiễu
 - **Gaussian:** phân bố xác suất là phân bố chuẩn hoặc Gauss
 - Nhiễu tương đương băng gốc: có giá trị phức
 - Các thành phần nhiễu đồng pha $N_r(t)$ và vuông pha $N_q(t)$ được mô phỏng một cách độc lập (nhiễu đối xứng tròn)

 162

162



Kênh AWGN

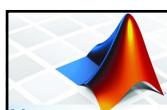
- Nhiễu trắng:**
 - Thuật ngữ “trắng” mô tả cụ thể rằng:
 - Trung bình của nhiễu bằng 0 (zero)
 - $E[N(t)] = 0$,
 - Hàm tự tương quan của nhiễu thời gian của nhiễu
$$\rho_N(\tau) = E[N(t) \cdot N^*(t + \tau)] = N_0 \delta(\tau).$$

→ các mẫu nhiễu phân biệt là độc lập nhau
 - Hàm tự tương quan cũng chỉ ra rằng các mẫu nhiễu có phương sai vô hạn
 - Nhiễu cần được lọc trước khi được lấy mẫu
 - Mỗi thành phần nhiễu đồng pha và vuông pha có tự tương quan $\frac{N_0}{2} \delta(\tau)$.
 - Thuật ngữ “trắng” liên quan đến tính chất phổ của nhiễu
 - PSD của nhiễu trắng là không đổi ở tất cả các thành phần tần số
$$S_N(f) = N_0 \text{ for all } f.$$

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

163

163



Kênh AWGN

- Tạo nhiễu Gaussian:**
 - Để mô phỏng nhiễu cộng, cần tạo các mẫu nhiễu Gaussian
 - Trong MATLAB có thể sử dụng hàm *randn* cho mục đích này
 - Tạo ra N mẫu nhiễu Gaussian phức độc lập nhau có phương sai là VarN
$$\text{Noise} = \text{sqrt}(\text{VarN}/2) * (\text{randn}(1,N) + j * \text{randn}(1,N))$$
 - Phản thực và ảo, mỗi phần có phương sai $\text{VarN}/2 \rightarrow$ Phương sai nhiễu tổng cộng là VarN
 - Phương sai nhiễu có thể được xác định từ công suất tín hiệu và tỉ số SNR yêu cầu.
 - Có thể dùng các hàm sẵn có trong MATLAB: *awgn*, *wgn*

```

function yNoise = addnoise(yClean,VarN)
% This function adds Gaussian noise into
% the input signal.
% yClean - the input signal
% VarN - the variance of noise
% yNoise - the noisy signal output

if (isreal(yClean))
    yNoise = yClean + sqrt(VarN)*randn(size(yClean));
else
    yNoise = yClean + sqrt(VarN/2) ...
        *(randn(size(yClean))+j*randn(size(yClean)));
end

```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

164

164

Kênh AWGN

• *Tạo nhiễu Gaussian:*




Figure showing three plots: a time-domain plot of amplitude vs. time (0 to 1 second) with a noisy blue line; an eye diagram showing a clear eye opening; and another eye diagram showing a noisy eye opening.

```
% Ví dụ về kênh AWGN
d = randint(1,100); % Message data
% NRZ coding
[t,x] = nrzcode(d,1e6,2000,'pol');
Ts = t(2)-t(1); % sampling time
% pulse shaping
xp = raisedcosfit(x,1.5e6,Ts,0.5);
% Transmit through AWGN channel
SNRdB = 10; % dB
SNR = 10^(SNRdB/10);
varn = var(xp)/SNR; % computing variance of noise
xnoise = addnoise(xp,varn);
% Display
plot(t,xp,'r',t,xnoise);grid;
xlabel('Time (s)');
ylabel('Amplitude');
eyediagram(xp,40);
eyediagram(xnoise,40);
```

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

165

165

Một số mô hình kênh thông tin

• *Mô hình kênh hữu tuyến:*

- Kênh hữu tuyến bao gồm:
 - Cáp xoắn đôi, cáp đồng trực
 - Sợi quang
 - Ống dẫn sóng
- Đối với các kênh hữu tuyến: được mô hình hóa là các hệ thống tuyến tính không thay đổi theo thời gian và được đặc trưng bởi hàm truyền đạt.

→ Xác định hàm truyền đạt $H(f)$ khi lập mô hình mô phỏng

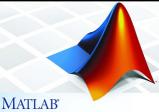
$$R(f) = S(f)H(f) + N(f)$$

- Hàm truyền đạt xác định quan hệ vào/ra của kênh: là hàm của khoảng cách truyền dẫn và tần số điều chế.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

166

166

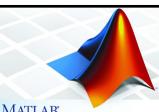


Một số mô hình kênh thông tin

- **Mô hình kênh vô tuyến:**
 - Kênh vô tuyến: một số yếu tố ảnh hưởng
 - Nhiễu: tín hiệu ngẫu nhiên cộng thêm vào tín hiệu thu được
 - Chú ý do nhiễu nhiệt từ các thành phần điện tử trong bộ thu
 - Cũng có thể mô hình hóa sự giao thoa từ các nguồn phát khác trong lân cận bộ thu
 - Mô hình thống kê được sử dụng để mô tả nhiễu
 - Méo dạng: quá trình lọc không mong muốn trong quá trình truyền sóng
 - Chú ý do truyền đa đường (multipath)
 - Cả hai mô hình xác định và thống kê đều có thể sử dụng phụ thuộc vào định rõ thời gian quan tâm
 - Bản chất và tính động của méo dạng là sự khác biệt quan trọng đối với hệ thống hỗn tuyến
 - Kênh pha định đa đường (multipath fading):
 - Tín hiệu từ bộ phát truyền tới bộ thu trên nhiều đường, mỗi đường có suy hao và độ trễ khác nhau
 - Tín hiệu thu là tổng của các tín hiệu này
 - Hiệu ứng: quá trình lọc tín hiệu phát không mong muốn

 167

167



Một số mô hình kênh thông tin

- **Mô hình kênh vô tuyến:**
 - Trong thông tin di động, các đặc tính kênh đa đường biến đổi nhanh → pha định
 - Fading cơ bản vì sự thay đổi pha do thay đổi độ trễ đối với các đường truyền khác nhau
 - Fading băng hẹp
 - Fading băng rộng
 - Đặc tính thống kê của kênh vô tuyến:
 - Dịch phổ Doppler (Doppler spectrum)
 - Trễ trễ (delay spread)
 - Thời gian kết hợp (coherence time)
 - Băng tần kết hợp (coherence bandwidth)

 168

168



Một số mô hình kênh thông tin

- Mô hình kênh vô tuyến:**
 - Tổn hao tuyến: L_p liên hệ giữa công suất thu P_r và công suất phát P_t
$$P_r = P_t \cdot \frac{G_r \cdot G_t}{L_p},$$
 - G_t, G_r là độ lợi của antenna
 - d - distance between transmitter and receiver,
 - f_c - carrier frequency,
 - h_b, h_m - antenna heights,
 - Terrain type, building density, ...
- Truyền trong không gian tự do: L_p được xác định bởi công thức Friis

$$L_p = \left(\frac{4\pi d}{\lambda_c} \right)^2 = \left(\frac{4\pi f_c d}{c} \right)^2$$

- Theo dB: $L_{P(dB)} = -20 \log_{10}(\frac{c}{4\pi}) + 20 \log_{10}(f_c) + 20 \log_{10}(d)$.
- Ví dụ: $f_c = 1$ GHz, $d = 1$ km

$$L_{P(dB)} = -146 \text{ dB} + 180 \text{ dB} + 60 \text{ dB} = 94 \text{ dB.}$$


 Học viện Công nghệ Bưu chính Viễn thông
 Posts & Telecommunications Institute of Technology

169

169



Một số mô hình kênh thông tin

- Mô hình kênh vô tuyến:**
 - Kênh 2 tia:
 - Gồm 2 tia truyền:
 - Tia truyền trực tiếp
 - Tia phản xạ
 - Phụ thuộc khoảng cách d , các tín hiệu thu được trên 2 tia sẽ giao thoa cộng hưởng hoặc triệt tiêu
 - Tổn hao tuyến: $L_p = \frac{1}{4} \cdot \left(\frac{4\pi f_c d}{c} \right)^2 \cdot \left(\frac{1}{\sin(\frac{2\pi c h_b h_m}{f_c d})} \right)^2$
 - Đối với $d \gg h_b h_m \rightarrow$ gần đúng
$$L_p \approx \left(\frac{d^2}{h_b h_m} \right)^2$$


 Học viện Công nghệ Bưu chính Viễn thông
 Posts & Telecommunications Institute of Technology

170

170



Một số mô hình kênh thông tin

- Mô hình kênh vô tuyến:**
 - Mô hình Okumura-Hata: mô hình tổn hao tuyến thực nghiệm
 - Mô hình khác nhau giữa các khu vực:
 - Đô thị
 - Nông thôn
 - Các khu vực rộng lớn
 - Các thành phố lớn, trung bình và nhỏ
 - Mô hình khu vực đô thị

$$L_{P(dB)} = A + B \log_{10}(d),$$

Trong đó:

$$\begin{aligned} A &= 69.55 + 26.16 \log_{10}(f_c) - 13.82 \log_{10}(h_b) - a(h_m) \\ B &= 44.9 - 6.55 \log_{10}(h_b) \\ a(h_m) &= (1.1 \log_{10}(f_c) - 0.7) \cdot h_m - (1.56 \log_{10}(f_c) - 0.8) \end{aligned}$$


 Posts & Telecommunications Institute of Technology

171

171



Một số mô hình kênh thông tin

- Mô hình kênh vô tuyến:**
 - Công suất tín hiệu và nhiễu:
 - Công suất tín hiệu thu được:

$$P_r = P_t \cdot \frac{G_r \cdot G_t}{L_P \cdot L_R},$$
 Trong đó L_R là tổn hao bỗ sung $\sim 2-3$ dB
 - Công suất nhiễu:

$$P_N = kT_0 \cdot B_W \cdot F,$$
 Trong đó:
 - k – hằng số Boltzmann ($1.38 \cdot 10^{-23}$ Ws/K)
 - T_0 – nhiệt độ K (tại nhiệt độ phòng $T_0 = 290$ K) $\Rightarrow kT_0 = 4 \cdot 10^{-21}$ W/Hz = -174 dBm/Hz
 - B_W – độ rộng băng tần của tín hiệu
 - F – hình ảnh (hệ số) nhiễu của bộ thu (điển hình 5 dB)


 Posts & Telecommunications Institute of Technology

172

172

MATLAB

Một số mô hình kênh thông tin

- Mô hình kênh vô tuyến:**
 - Tỉ số tín hiệu trên nhiễu SNR:
 - Tỉ lệ giữa công suất tín hiệu thu và công suất nhiễu:

$$\text{SNR} = \frac{P_t G_r \cdot G_t}{kT_0 \cdot B_W \cdot F \cdot L_P \cdot L_R}.$$
 - Tỉ số Es/N0:
 - Đối với hiệu năng của hệ thống viễn thông được đo qua tốc độ lỗi ký hiệu, tỉ lệ năng lượng tín hiệu E_s và mật độ phô công suất nhiễu N_0 hay sử dụng hơn.
 - Vì $E_s = P_r T_s$ và $N_0 = kT_0 \cdot F = P_N / B_W$, $\Rightarrow \frac{E_s}{N_0} = \text{SNR} \cdot \frac{B_W}{R_s}$,
 - Es/N0 được xác định: $\frac{E_s}{N_0} = \frac{P_t G_r \cdot G_t}{kT_0 \cdot R_s \cdot F \cdot L_P \cdot L_R}$. T_s, R_s là chu kỳ và tốc độ ký hiệu tương ứng
 - Theo dB:

$$(\frac{E_s}{N_0})_{(dB)} = P_{t(dBm)} + G_{t(dB)} + Gr(dB) - (kT_0)_{(dBm/Hz)} - R_s(dBHz) - F_{(dB)} - L_{R(dB)}.$$

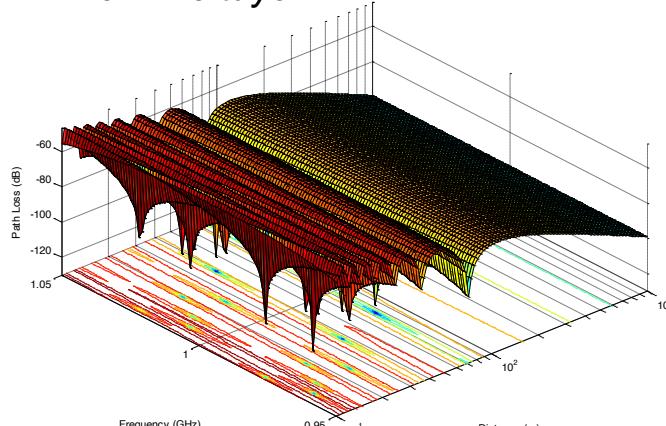
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

173

173

MATLAB

Một số mô hình kênh thông tin

- Mô hình kênh vô tuyến:**


Tổn hao đường truyền trên một bề mặt phản xạ phẳng

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

174

174

MATLAB

Một số mô hình kênh thông tin

- Mô hình kênh fading đa đường:**
 - Tín hiệu phát truyền từ bộ phát tới bộ thu trên nhiều đường khác nhau.
 - Các đường này có đặc tính khác nhau về:
 - Suy hao đường truyền a_k
 - Độ trễ đường τ_k
 - Độ dịch pha ϕ_k
 - Góc tới θ_k
 - Từ các tham số trên → xác định đáp ứng xung của kênh (tương đương bằng gốc):
$$h(t) = \sum_{k=1}^K a_k \cdot e^{j\phi_k} \cdot e^{-j2\pi f_c \tau_k} \cdot \delta(t - \tau_k)$$
 - Độ trễ τ_k cũng đóng góp vào độ dịch pha ϕ_k

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

175

175

MATLAB

Một số mô hình kênh thông tin

- Mô hình kênh fading đa đường:**
 - Tín hiệu thu được: tích chập giữa tín hiệu phát và đáp ứng xung
$$R(t) = s(t) * h(t) = \sum_{k=1}^K a_k \cdot e^{j\phi_k} \cdot e^{-j2\pi f_c \tau_k} \cdot s(t - \tau_k).$$
 - Tín hiệu thu bao gồm nhiều bản sao của tín hiệu phát đã được:
 - Định cờ bởi $a_k \cdot e^{j\phi_k} \cdot e^{-j2\pi f_c \tau_k}$
 - Trễ bởi τ_k
 - Đáp ứng tần của kênh:
$$H(f) = \sum_{k=1}^K a_k \cdot e^{j\phi_k} \cdot e^{-j2\pi f_c \tau_k} \cdot e^{-j2\pi f \tau_k}.$$
 - Đối với một tần số xác định f , đáp ứng tần là tổng của các số phức
 - Khi các số hạng này cộng triệt tiêu nhau, đáp ứng tần là rất nhỏ hoặc thậm chí bằng 0 tại tần số đó
 - Các điểm null trong đáp ứng tần của kênh là điểm hình trong thông tin vô tuyến và được xem như là pha định lựa chọn tần số.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

176

176

MATLAB

Một số mô hình kênh thông tin

- Mô hình kênh fading đa đường:**
 - Thực hiện trong MATLAB:

HH = PropData.Field.*exp(-j*2*pi*fc*tau) * exp(-j*2*pi*tau*ff);

- Chú ý: $\tau \cdot ff$ là phép nhân ma trận và tích của hai hàm mũ \exp cũng là nhân ma trận → tạo ra phép tính tổng trong biểu thức đáp ứng tần.

Đáp ứng xung của kênh cho thấy suy hao, trễ và pha của mỗi đường giữa bộ thu và bộ phát

Đáp ứng tần của một kênh đa đường được đặc trưng bởi các điểm notch.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

177

177

MATLAB

Một số mô hình kênh thông tin

- Mô hình kênh fading đa đường:**
 - Hiệu ứng đa đường: được mô phỏng bởi bộ lọc FIR (đường dây trễ nhánh)
 - Số lượng nhánh của bộ lọc được xác định bởi tích trái trễ và tốc độ mẫu.
 - Các nhánh là ngẫu nhiên theo một phân bố xác định (Gaussian)
 - Độ lớn trọng số của nhánh phản ánh mặt cắt trễ công suất
 - Các nhánh biến đổi theo thời gian (dịch Doppler)

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

178

178

MATLAB

Một số mô hình kênh thông tin

- Mô hình kênh rời rạc:**
 - Mô phỏng trên cơ sở từng symbol.
 - Mô hình kênh rời rạc: bao hàm tất cả các phần tử của hệ thống truyền thông giữa 2 điểm A và B
 - Đầu vào tại A: vecto các ký hiệu rời rạc $X = [x_1, x_2, \dots]$
 - Đầu ra tại B: vecto các ký hiệu rời rạc $Y = [y_1, y_2, \dots]$
 - Các mô hình kênh rời rạc mô tả cơ chế tạo lỗi có thể xảy ra:
 - Kênh rời rạc không nhớ: chuyển tiếp các ký hiệu vào/ra không tương quan về thời gian.
 - Kênh rời rạc có nhớ: có tương quan về thời gian.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

179

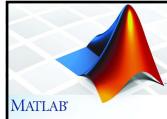
MATLAB

Một số mô hình kênh thông tin

- Mô hình kênh rời rạc:**
 - Mô hình kênh rời rạc có hiệu quả tính toán cao hơn:
 - Tốc độ mẫu thấp hơn: tốc độ ký hiệu
 - Mức độ trừu tượng hóa (rút gọn) cao hơn
 - Được sử dụng cho thiết kế và phân tích các bộ mã hóa sửa lỗi, interleavers,... và cho mô phỏng mạng.
 - Mô hình kênh rời rạc không nhớ:
 - Kênh đối xứng nhị phân (BSC)
 - Kênh bất đối xứng nhị phân
 - Kênh nhiều đầu vào và đầu ra
 - Mô hình Markov cho kênh rời rạc có nhớ: Mô hình Markov ẩn
 - Mô hình Markov 2 trạng thái (Mô hình Gilbert)
 - Mô hình Markov N trạng thái
 - Mô hình Markov bậc 1.

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

180

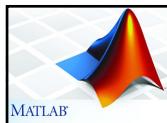


Ước tính tham số và đánh giá hiệu năng

- **Ước tính tham số**
- **Ước tính tần số SNR**
- **Đánh giá hiệu năng hệ thống**
- **Thực hiện mô phỏng một số hệ thống viễn thông**

 181

181



Ước tính tham số

- **Ước tính mức sóng trung bình:**
 - Đổi với dạng sóng $X(t) \rightarrow$ được lấy mẫu $X(kT_s) \equiv X_k$.
 - Mức trung bình: $\langle X \rangle = \frac{1}{N} \sum_{k=1}^N X_k$
 - Kỳ vọng của ước tính:

$$E(\langle X \rangle) = E\left(\frac{1}{N} \sum_{k=1}^N X_k\right) = \frac{1}{N} \sum_{k=1}^N E(X_k) = \frac{1}{N} \sum_{k=1}^N E(X) = E(X)$$
 - Phương sai của ước tính:

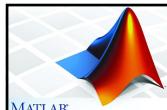
$$\text{Var}(\langle X \rangle) = E(\langle X \rangle - E(\langle X \rangle))^2 = \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N C_{xx}(i, j)$$

$$C_{xx}(i, j) = E[(X_i - E(X_i))(X_j - E(X_j))]$$

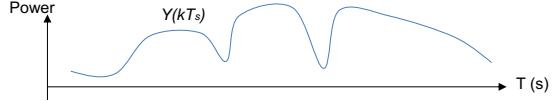
$\longrightarrow \text{Var}(\langle X \rangle) \leq \frac{\sigma_x^2}{N}$ σ_x^2 - phương sai của quá trình,
 N - số lượng mẫu độc lập

 182

182

 **Ước tính tham số**

- **Ước tính công suất trung bình:**
 - Bộ ước tính công suất trung bình:

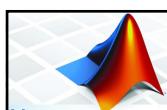
$$P_N(X) = \frac{1}{N} \sum_{k=1}^N X_k^2$$
 - Đặt $Y(t) = X^2(t)$ $\rightarrow \langle Y \rangle = \frac{1}{N} \sum_{k=1}^N Y_k$

- Kỳ vọng của ước tính:

$$E(\langle Y \rangle) = E(P_N(X)) = E(X^2)$$
- Phương sai của ước tính:

$$\text{Var}(\langle Y \rangle) = E((\langle Y \rangle - E(\langle Y \rangle))^2)$$

 183

183

 **Ước tính SNR**

- **Bộ ước tính:**
 - Tín hiệu đầu ra hệ thống: $x = s_0 + n_0$
 - Tín hiệu: $s_0 = As_\tau$
 - Trong đó: $s_\tau = \begin{cases} s(t-\tau), & 0 \leq t \leq T \quad \text{or} \quad s(iT_s - \tau), & i = 1, 2, \dots, N \\ 0, & \text{khác} \end{cases}$
 - Nhiều: sai số trung bình bình phương \rightarrow tối thiểu hóa

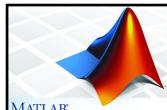
$$\varepsilon^2 = \langle (x - As_\tau)^2 \rangle$$
 - Công suất trung bình N điểm:

$$\langle s_\tau^2 \rangle = \frac{1}{N} \sum_{k=1}^N s^2(kT_s - \tau)$$
 - Ước tính SNR:

$$SNR = \frac{\langle s_0^2 \rangle}{\varepsilon^2}$$

 184

184



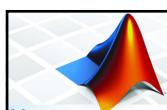
Đánh giá hiệu năng hệ thống

- Phương pháp Monte-Carlo:**
 - Mô phỏng Monte-Carlo: tính toán dựa vào việc lấy mẫu ngẫu nhiên lặp lại.
 - Đối tượng mô phỏng Monte-Carlo: để đánh giá tỉ số lỗi ký hiệu (symbol error rate) mà hệ thống có thể thực hiện được
 - Ý tưởng đằng sau mô phỏng Monte-Carlo: đơn giản
 - Mô phỏng hệ thống lặp lại
 - Mỗi lần chạy mô phỏng, đếm số lượng ký hiệu được phát đi và số lượng lỗi ký hiệu
 - Ước tính tốc độ lỗi ký hiệu như là tỉ lệ giữa tổng số lỗi quan sát được trên tổng số ký hiệu phát đi

→ Cấu trúc đơn giản để mô phỏng Monte-Carlo

185

185



Đánh giá hiệu năng hệ thống

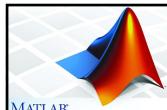
- Phương pháp Monte-Carlo:**
 - Bên trong vòng lặp:
 - Thực hiện mô phỏng hệ thống
 - Đếm các đại lượng tham số quan tâm

```
% inner loop iterates until enough errors have been found
while (~Done)
    NumErrors(kk) = NumErrors(kk) + MCSimple( Parameters );
    NumSymbols(kk) = NumSymbols(kk) + Parameters.NSymbols;

    % compute Stop condition
    Done = NumErrors(kk) > MinErrors || NumSymbols(kk) > MaxSymbols;
end
```
 - Vòng lặp thực hiện bao nhiêu lần là đủ? → Phụ thuộc vào:
 - Mức độ chính xác (độ tin cậy) mong muốn
 - Tốc độ lỗi ký hiệu yêu cầu

186

186



Đánh giá hiệu năng hệ thống

- Phương pháp Monte-Carlo:**
 - Khoảng tin cậy:
 - Mong muốn $|\hat{P}_e - P_e|$ là nhỏ, trong đó P_e là ước tính lỗi bằng mô phỏng, \hat{P}_e là tốc độ lỗi thực.
 - Cụ thể, đảm bảo xác suất p_c để $|\hat{P}_e - P_e| < s_c$ là cao (ví dụ: $p_c = 95\%$)

Trong đó s_c được gọi là khoảng tin cậy, nó phụ thuộc vào độ tin cậy p_c , xác suất lỗi P_e và số lượng ký hiệu phát.

 - Nó được chứng minh rằng:
$$s_c = z_c \cdot \sqrt{\frac{P_e(1 - P_e)}{N}},$$

Trong đó z_c phụ thuộc vào độ tin cậy p_c

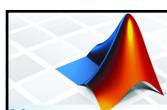
Cụ thể, $Q(z_c) = (1-p_c)/2$. Ví dụ: $p_c = 95\%$, $z_c = 1,96$.

 - Bao nhiêu mô phỏng cần phải chạy là đủ :



187

187



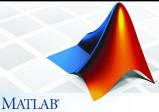
Đánh giá hiệu năng hệ thống

- Phương pháp Monte-Carlo:**
 - Đối với mô phỏng Monte-Carlo, tiêu chuẩn dừng có thể được xác định từ:
 - Độ tin cậy p_c mong muốn (hay z_c).
 - Khoảng tin cậy s_c có thể chấp nhận được
 - Tốc độ lỗi P_e
 - Giải phương trình tìm N ta có được:
$$N = P_e \cdot (1 - P_e) \cdot (z_c / s_c)^2.$$
 - Mô phỏng Monte-Carlo dừng sau khi thực hiện mô phỏng truyền dẫn được N ký hiệu.
 - Ví dụ: Với $p_c = 95\%$, $P_e = 10^{-3}$, và $s_c = 10^{-4} \rightarrow N \approx 400000$.



188

188

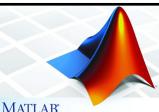


Đánh giá hiệu năng hệ thống

- *Phương pháp Monte-Carlo:*
 - Khi mô phỏng các hệ thống truyền thông, tốc độ lỗi thường rất nhỏ → đáng mong muốn để xác định khoảng tin cậy là một phần nhỏ của tốc độ lỗi.
 - Khoảng tin cậy có dạng $s_c = \alpha_c \cdot P_e$ (VD: $\alpha_c = 0,1$ cho 10% sai số ước tính có thể chấp nhận được)
 - Biến đổi biểu thức tìm N và sắp xếp lại số hạng ta có được:

$$P_e \cdot N = (1 - P_e) \cdot (z_c / \alpha_c)^2 \approx (z_c / \alpha_c)^2.$$
 - Ở đây $P_e \cdot N$ là số lượng lỗi được kỳ vọng.
 - Nghĩa là: mô phỏng sẽ dừng khi số lượng lỗi đạt đến $(z_c / \alpha_c)^2$.
 - Theo kinh nghiệm: Chạy mô phỏng cho đến khi khoảng 400 lỗi xảy ra ($p_c = 95\%$ và $\alpha_c = 10\%$)

189



Đánh giá hiệu năng hệ thống

- *Phương pháp Monte-Carlo:*

```
% Thiết lập tham số cho mô phỏng;
...
EsOverN0dB = 0:0.5:9; % Thay đổi SNR từ 0 đến 9dB
MaxSymbols = 1e6; % Số symbol cực đại
...
% Xác định tham số dừng vòng lặp
...
MinErrors = ( ZValue/ConfIntSize )^2; %
...
% Các biến vòng lặp
NumErrors = zeros( size( EsOverN0dB ) );
NumSymbols = zeros( size( EsOverN0dB ) );

% Vòng lặp ngoài
for kk = 1:length( EsOverN0dB )
    EsOverN0 = dB2lin( EsOverN0dB(kk) ); % Chuyển đổi SNR cho mỗi vòng lặp
    Done = false; % Khởi tạo lại điều kiện dừng cho vòng lặp trong
    % Vòng lặp trong
    while ( ~Done )
        ...
        Done = NumErrors(kk) > MinErrors || NumSymbols(kk) > MaxSymbols; % Tính điều kiện dừng
    end
    % Tính tốc độ lỗi
    ...
end
```

190

MATLAB

Đánh giá hiệu năng hệ thống

- Phương pháp bán giải tích:**
 - Giảm thời gian chạy mô phỏng
 - Được thực hiện:
 - Phát một chuỗi PN thông qua hệ thống mô phỏng
 - Tính xác suất lỗi trung bình dựa trên phương pháp giải tích thông qua đánh giá hàm mật độ xác suất của mẫu thu được
 - Đối với trường hợp đơn giản: hệ thống nhị phân

Xác suất lỗi: $p_k = \int_{-\infty}^{-v_k} f_n(\eta) d\eta = F_n(-v_k)$

$$v_k = (1 - \varepsilon_k) A$$

Đối với nguồn nhiễu tương đương Gauss

$$p_k = \frac{1}{2} \operatorname{erfc}\left(\frac{v_k}{\sqrt{2}\sigma}\right)$$

Tốc độ lỗi tổng cộng: $p = \frac{1}{N} \sum_{k=1}^N p_k$

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

191

191

MATLAB

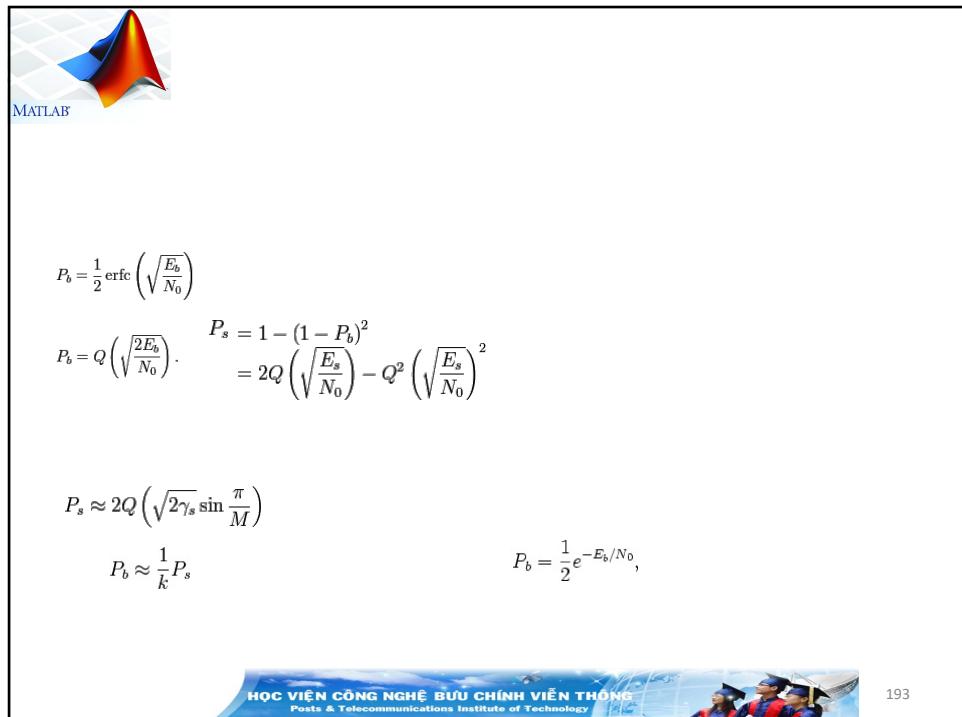
Đánh giá hiệu năng hệ thống

- Một số phương pháp khác:**
 - Phương pháp ngoại suy đuôi (Tail extrapolation):
 - Kỹ thuật cho phép mở rộng sự quan tâm
 - Thực hiện xác định tốc độ lỗi tại chế độ lỗi cao: các ước tính có độ tin cậy cao hơn
 - Mở rộng các ước tính để dự đoán tốc độ lỗi tại những điều kiện SNR mà ở đó tốc độ lỗi là nhỏ
 - Phương pháp lấy mẫu quan trọng (Important sampling):
 - Kỹ thuật để ước tính các tính chất của một phân bố xác định trong khi chỉ có các mẫu từ một dạng phân bố khác.
 - Thực hiện lấy mẫu các quá trình: các quá trình này đặc trưng cho các quá trình khác nhau ảnh hưởng lên tín hiệu

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts & Telecommunications Institute of Technology

192

192



193