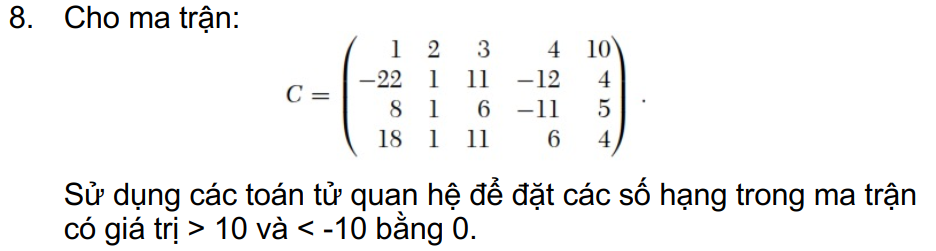
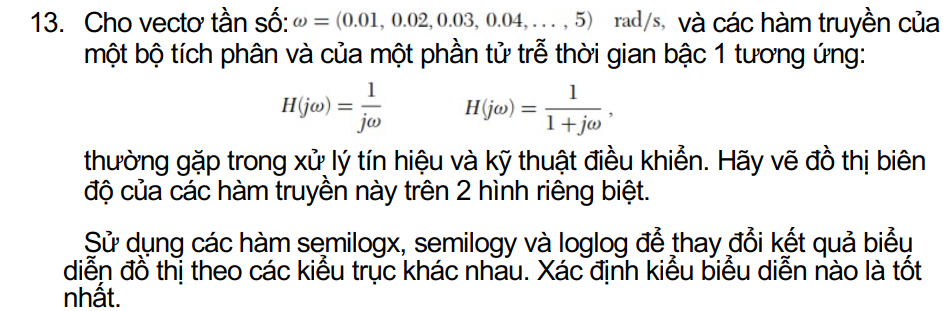
**BÀI TẬP SLIDE:**

****

|  |
| --- |
| C = [1 2 3 4 10; -22 1 11 -12 4; 8 1 6 -11 5; 18 1 11 6 4];  for i=1:4  for j=1:5  if ((C(i,j) > 10) || (C(i,j) < -10)) % Neu phan tu trong ma tran lon hon 10 hoac nho hon -10, gan = 0  C(i, j) = 0;  end  end  end  %% Cach 2  clc;  test1 = C < -10;  test2 = C > 10;  test3 = test1 | test2;  test4 = ~(test3);  C = C.\*test4; |

****

|  |
| --- |
| w = (0:0.01:5);  H1 = 1./w;  H2 = 1./(1+w.^2).^(1/2);  plot(w, H1);  title('Do thi 1');  xlabel('w');  ylabel('H(jw)');  figure;  plot(w, H2);  title('Do thi 2');  xlabel('w');  ylabel('H(jw)'); |

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

|  |
| --- |
| w = (0:0.01:5);  H1 = 1./w;  H2 = 1./(1+w.^2).^(1/2);  subplot(211); % do thi bieu dien theo ma tran (2,1) voi vi tri la 1  plot(w, H1);  title('Do thi 1');  xlabel('w');  ylabel('H(jw)');  subplot(212);  plot(w, H2);  title('Do thi 2');  xlabel('w');  ylabel('H(jw)'); |

**15. Tính và vẽ hàm x2 + y2 trong dải [-2,2]x[-1,1] sử dụng lưới có cỡ bước 0.2 theo chiều x và 0.1 theo chiều y.**

|  |
| --- |
| x = (-2:0.2:2);  y = (-1:0.1:1);  *% z = ones(length(x),1);*  *% X = z\*x;*  *% t = ones(length(y),1);*  *% Y = y\*t;*  *% Ham tao 2 ma tran X, Y thanh ma tran 2D de ve hinh 3D*  [X, Y] = meshgrid(x,y);  f = X.^2 + Y.^2;  surf(x, y, f); |

**16. Vẽ hình cầu có bán kính R = 3.**

|  |
| --- |
| r=3;  syms u v  x= r\* sin(u) \* cos (v);  y=r\* sin(u) \* sin(v);  z= r\* cos(u);  fsurf (x,y,z)  axis equal;  title ('Hinh cau co ban kinh R=3');  figure;  [X, Y, Z] = sphere(100);  mesh(X\*r, Y\*r, Z\*r);  axis equal;  title('Hinh cau co ban kinh R=3'); |

**17. Tạo vectơ y = (1, 1.5, 2, ..., 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, ..., 1.5, 1). Tạo vectơ z chỉ chứa các số nguyên từ vectơ y.**

|  |
| --- |
| clc;  A = (1:0.5:5);  B = flip(A);  [rows, colums] = size(A);  v = mod(A, 1) == 0; % Check so nguyen trong vecto A  z = A(v); % In ra cac so co gia tri logic la 1 trong v tuong ung voi vi tri trong MT A |

**18. Định nghĩa cấu trúc color với các trường dữ liệu red, blue và green. Sau đó định nghĩa một trường 1x20 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]**

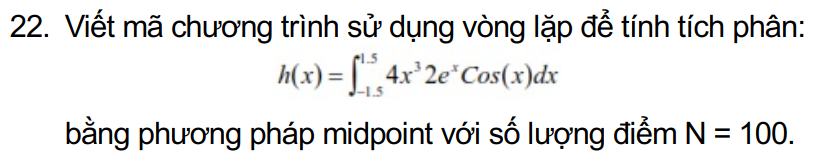
|  |
| --- |
| clc;  color = struct('red', [], 'blue', [], 'green', []);  myStruct = struct('field', repmat(color,1,20));  for i=1:20  myStruct.field(i).red = 'yes';  myStruct.field(i).blue = 'no';  myStruct.field(i).green = [0, 256, 0];  end |

**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)**

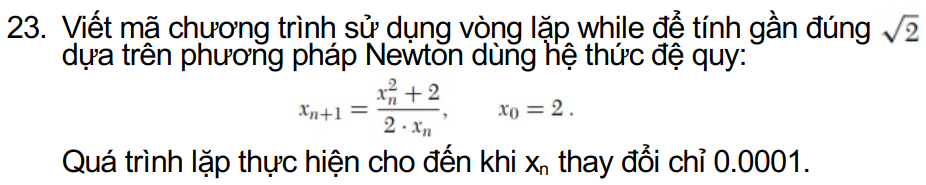
|  |
| --- |
| function [s, p] = circle\_prog(R)  xo = 1;  yo = 1;  Phi = linspace(0, 2\*pi);  x = xo + R \* cos(Phi);  y = yo + R \* sin(Phi);  plot(x, y, 'rp-');  axis equal;  s = pi\*R.^2;  s = sprintf('%.5f',s);  p = 2\*pi\*R;  p = sprintf('%.5f',p); |

**21**. **Cho một hàm f(x) = x3/3+4x2+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)**

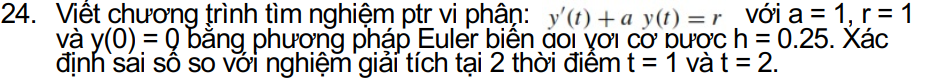
|  |
| --- |
| a = input('Nhap gia tri dau a = ');  b = input('Nhap gia tri cuoi b = ');  % f: hàm so can giai  f = @(x) (x.^3)/3 + 4\*x.^2 + x - 6;  % a, b: ?oan chia nghiem  % epsilon: ?o chinh xác mong muon  epsilon = 0.000001;    fa = f(a);  fb = f(b);  % Vi nghiem o khoang giua, lam cho f(mid) = 0 => 2 khoang a, b o bien trai va phai se  % lam cho f(a) \* f(b) luon nho hon 0  if fa\*fb > 0  disp('Phuong trinh vo nghiem');  else  while b - a > epsilon\*b  % Tim gia tri trung binh cua khoang [a, b]  c = (a + b)/2;  fc = f(c);  % Neu fc = 0, c chinh la nghiem  if fc == 0  x = c;  return;  end  % Neu f(a)\*f(c) > 0, nghiem nam trong khoang [c,b]  if sign(fc) == sign(fa)  a = c;  fa = fc;  % Ng?oc lai, nam trong khoang nghiem trong khoang [a,c]  else  b = c;  fb = fc;  end  end  x = c;  end  disp(x); |

****

|  |
| --- |
| function I = bai22(f, a, b, N)  *%Dung midpoint*  *%f = @(x) 4\*x.^3\*2\*exp(x)\*cos(x); %Ham can tinh tich phan*  *% a = -1.5; %Gioi han duoi*  *% b = 1.5; %Gioi han tren*  *% N = 100; %So doan con chia doan tich phan*  dx = (b-a)/N; % Do dai cua moi doan con  I = 0; %Khoi tao gia tri tich phan ban dau  for i=1:N  x\_mid = a + (i-0.5) \* dx; % diem giua cua doan con  I = I + f(x\_mid); % tinh gia tri tich phan  end  I = I\*dx;  end |

****

|  |
| --- |
| function y = bai23()  x = 2;  epsilon = 0.0001;  dx = inf;    while abs(dx) > epsilon  x\_new = (x^2+2)/(2\*x);  dx = x\_new - x;  x= x\_new;  end  y = x; |

****

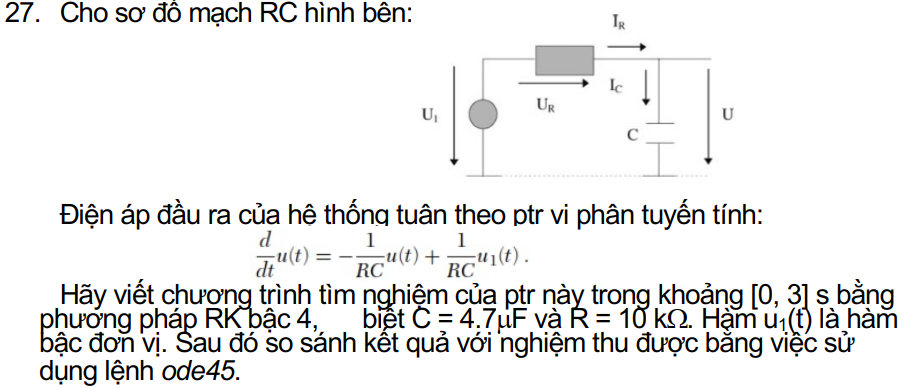
|  |
| --- |
| a = 1;  r = 1;  h = 0.25;  y0 = 0;  f = @(t,y)r-a\*y;  y\_ex = @(t) (y0 - r/a)\*exp(-a\*t) + r/a;  k1 = 0;  k2 = 3;  t0 = k1;  n = (k2-k1)/h;  t(1) = t0;  y(1) = y0;  for i=1:n  t(i+1) = t(i)+h;  y(i+1) = y(i)+h\*f(t(i),y(i));  end  plot(t,y,'ro-','linewidth', 2);  hold on  plot(t,y\_ex(t),'b\*-','linewidth',2);  legend('Nghiem theo Euler bien doi','Nghiem giai tich');  grid on |

**25. Tương tự bài tập 24 nhưng sử dụng phương pháp RK bậc 3.**

|  |
| --- |
| function [y] = bai25( t1,t2,h )  a=1;  r=1;  N=(t2-t1)/h;  y(1)=0;  x(1)=0;  f= @(y) r-a\*y;  for i=1:N  x(i+1)=x(i)+(2/3)\*h;  k1=f(y(i))\*h;  k2=f(y(i)+(2/3)\*k1)\*h;  k3=f(y(i)+(2/3)\*k2)\*h;  y(i+1)=y(i)+(2/8)\*k1+(3/8)\*k2+(3/8)\*k3;  end  plot(x,y,'r', 'linewidth',2)  title (' Do thi nghiem cua bai toan y(t)')  legend('y(t)')    end    *%% Cach 2*  *% a=1;r=1;*  *% t0=0;*  *% y0=0; % Ði?u ki?n ban ð?u*  *% tf=2;*  *%*  *% t = [0:0.01:tf];*  *% yt=1-exp(-a\*t); % nghi?m c?a PTVT*  *% plot(t,yt,'k'), hold on*  *%*  *% h = 1/4;*  *% klast = (tf-t0)/h;*  *%*  *% y(1) = y0;*  *% tvec(1) = t0;*  *% for k = 1:klast*  *% k1 = fun1(y(k),tvec(k));*  *% k2 = fun1(y(k)+2\*h\*k1/3,tvec(k)+2\*h/3);*  *% k3 = fun1(y(k)+2\*h\*k2/3,tvec(k)+2\*h/3);*  *% y(k + 1) = y(k) + h\*((2/8)\*k1 + (3/8)\*k2 + (3/8)\*k3);*  *% %Công th?c RK-3*  *% tvec(k+1) = tvec(k) + h;*  *% plot([k - 1 k]\*h,[y(k) y(k+1)],'b', k\*h,y(k+1),'ro')*  *% pause;*  *% end*  *% =========================================================*  *% %”fun1” function*  *% function yf = fun1(y,t)*  *% a = 1; r = 1;*  *% yf = r-a\*y;* |

**26. Tương tự bài tập 24 nhưng sử dụng phương pháp RK bậc 4.**

|  |
| --- |
| function [y] = bai26( t1,t2,h )  a=1;  r=1;  N=(t2-t1)/h;  y(1)=0;  x(1)=0;  f= @(y) r-a\*y  for i=1:N  x(i+1)= x(i)+h/2;  k1 = h\* f(y(i));  k2=h \* f(y(i)+k1/2);  k3=h\*f(y(i)+k2/2);  k4=h\*f(y(i)+k3);  y(i+1)=y(i)+(1/6) \*(k1+2\*k2+2\*k3+k4);  end  plot (x,y,'k','linewidth',2);  legend('y(t)');  title('Do thi nghiem y(t)')  end  *%% Cach 2*  *% clear, clf*  *% a=1;r=1;*  *% t0=0;*  *% y0=0; % Ði?u ki?n ban ð?u*  *% tf=2;*  *%*  *% t = [0:0.01:tf];*  *% yt=1-exp(-a\*t); % nghi?m c?a PTVT*  *% plot(t,yt,'k'), hold on*  *%*  *% h = 1/4;*  *% klast = (tf-t0)/h;*  *%*  *% y(1) = y0;*  *% tvec(1) = t0;*  *% for k = 1:klast*  *% k1 = feval(@fun1,y(k),tvec(k));*  *% k2 = feval(@fun1,y(k)+h\*k1/2,tvec(k)+h/2);*  *% k3 = feval(@fun1,y(k)+h\*k2/2,tvec(k)+h/2);*  *% k4 = feval(@fun1,y(k)+h\*k3,tvec(k)+h);*  *% y(k + 1) = y(k) + h\*(k1 + 2\*k2 + 2\*k3 + k4)/6; %Công th?c RK-4*  *% tvec(k+1) = tvec(k) + h;*  *% plot([k - 1 k]\*h,[y(k) y(k+1)],'b', k\*h,y(k+1),'ro')*  *% pause;*  *% end*  *%*  *% %”fun1” function*  *% function yf = fun1(y,t)*  *% a = 1; r = 1;*  *% yf = r-a\*y;* |

****

|  |
| --- |
| C = 4.7e-6;  R = 10e3;  %Khoi tao dieu kien ban dau cua RK4  t1=0;  t2=3;  u(1)= 0;  x(1)=0;  h=3e-3;  N=(t2-t1)/h;  f1= @(u) (-1/(R\*C))\*u + 1/(R\*C); %Ham su dung trong RK4  %Tim nghiem = cong thuc RK4  for i=1:N  x(i+1)= x(i)+h/2;  k1 = h\* f1(u(i));  k2=h \* f1(u(i)+k1/2);  k3=h\*f1(u(i)+k2/2);  k4=h\*f1(u(i)+k3);  u(i+1)=u(i)+(1/6) \*(k1+2\*k2+2\*k3+k4);  end  %Khoi tao dieu kien ban dau su dung pp ode45  u0 = 0;% V  t0 = 0; % s  tspan = [0 3];  n = 1000;  f = @(t, u) (-1/(R\*C)) \* u + (1/(R\*C)); %Ham su dung trong ode45  % Tính nghi?m b?ng ode45  [ t, u\_ode45] = ode45(@(t,u)f(t,u), tspan, u0, n);  % In ket qua ra = pp ode45  subplot(2,1,1);  plot (t,u\_ode45,'r', 'linewidth',2)  legend('ode45');  title('Ket qua su dung ode45');  %In ket qua ra = pp RK4 => 2 ket qua giong nhau  subplot(2,1,2)  plot(x,u,'linewidth',2)  axis([0 1 0 1.5])  legend('RK4');  title('Ket qua su dung RK4'); |

**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].**

**– Gợi ý: Chuyển đổi vector hàng thành ma trận mx4, sau đó dùng hàm bi2de để chuyển đổi sang dạng thập phân.**

|  |
| --- |
| *% Bài 31*  **x = randint(1,128);** *% Tạo véc tơ x gom 128 bít ngẫu nhiên phân bốđều*  **id=1;**  *%Chuyển véc tơ x thành ma trận y (32x4)*  **for i=1:32**  **for j=1:4**  **y(i,j) = x(id);**  **id = id+1;**  **end;**  **end;**  **disp('Ma tran y la: ')**  **y**  *%Chuyển véc tơ y từ nhị phân thành thập phân*  **z = bi2de(y,'left-msb') %Lấy trọng số từ trái sang** |

**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)**

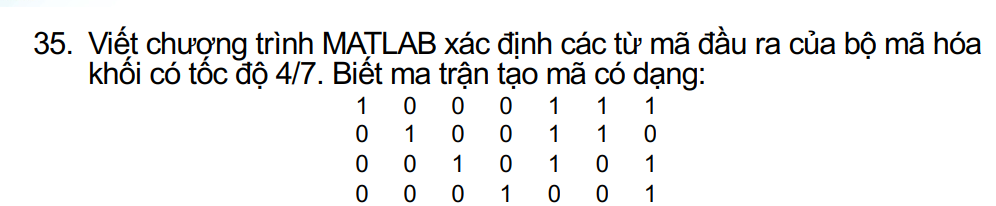
|  |
| --- |
| %Ham nen theo luat A  function [y,amax]= Alaw(x,A)  % voi x la vector dau vao, A la hang so  xmax= max(abs(x));% Tim gia tri tuyet doi lon nhat cua tin hieu  if xmax <= 1/A  amax = A\*xmax/(1+log(1+A));  else  amax = (1+log(A\*xmax))/log(1+A);  y = sign(x) \* log(1 + A \* abs(x)/amax)/log(1+A);  end  end  function [y] = invAlaw(x, A)  % Ham giai nen tien doan theo luat A-Law  % x: vector dau vao da bi nen theo luat A-Law  % A: he so nen theo luat A-Law  % amax: bien do cuc dai cua tin hieu dau vao  % y: vector sau khi giai nen  amax=max(abs(x));  xn = sign(x) .\* ((1+A).^abs(x) - 1) ./ A; % Ap dung cong thuc giai nen Mu-Law  y = xn .\* amax; % Nhan lai bien do cuc dai  end |

**33. Viết chương trình mã hóa tín hiệu x = 2cos(4pt) tại tần số lấy mẫu fs = 20 Hz sử dụng quá trình PCM theo luật A với 8 mức lượng tử. Hãy xác đinh 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)**

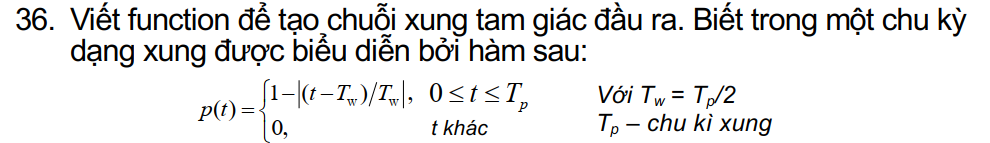
|  |
| --- |
| clc;  clear all;  fs=20;  A=8; % He so nen  %M=32; % So luong muc luong tu  N=2^7;  f=440;  T0=0;  Tf=5e-3;  Ts=(Tf-T0)/(N-1);  t=T0:Ts:Tf;  x=2\*cos(4\*pi\*f\*t);  plot(t,x,'linewidth',1);  hold on;  fc=f-fs;  xs= 2\*cos(4\*pi\*fc\*t);  stairs (t,xs,'g');  hold on;  [y,amax]=Alaw(x,A); % Nen tin hieu  [code,yq,sqnr]=uniform\_PCM(y,M); %Ma hoa  xq=invAlaw(yq,A); %Giai nen tin hieu  xq= xq\*amax;  amax=amax\*10;  xq=xq/amax;  sqnr= 20 \*log10(norm(x)/norm(x-xq)); % in dB  stairs(t,xq,'r','linewidth',1);  legend('Tin hieu ban dau','Tin hieu lay mau','Tin hieu luong tu hoa');  axis([0 1.5e-3 -2 2]); |

**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.**

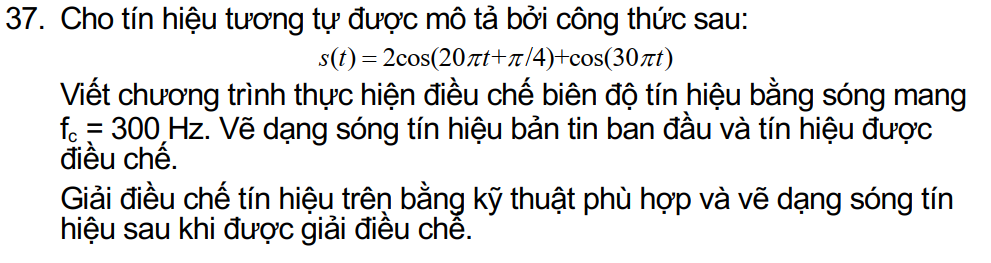
|  |
| --- |
| function [t,y,code] = rzcode(d,R,Ns)  % Chuong trinh ví d? ve ma duong truyen R  % d chuoi du lieu dau vao  % R toc do du lieu  % Ns so luong mau  % t vecto thoi gian  % y vecto mau tín hieu    Tb = 1/R; % chu ki bit  Nb = length(d); % do dai chuoi bit  Timewindow = Nb\*Tb; % thoi gian cua so  ts = Timewindow/(Ns-1); % thoi gian lay mau  t = 0:ts:Timewindow; % vecto thoi gian    y = zeros(size(t)); % tao vecto y toan so 0  code = [];% khoi tao vecto ma    for k = 1:Ns  n = fix(t(k)/Tb)+1; % lay phan nguyen cua t(k)/Tb cong them 1  if n >= Nb  n = Nb;  end;  if mod(t(k),Tb)<=Tb/2 % lay phan du cua t(k)/Tb so sanh voi Tb/2  y(k) = d(n);  code(n) = d(n);  else  y(k) = 0;  end;  end; |

****

|  |
| --- |
| % Bài 35  a= 0:15; % Tao gia tri ban tin co dang thap phan  i=de2bi(a','left-msb'); % Chuyen ban tin thanh dang nhi phan co trong so tu cao den thap  g=[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];  % Nhap ma tran sinh G  c=mod((i\*g),2); % Tinh ra ma tran ma |

****

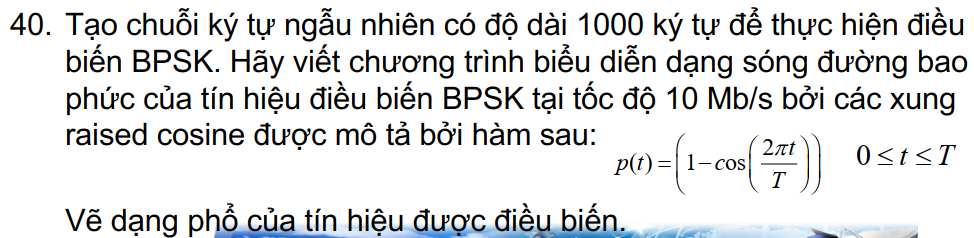
|  |
| --- |
| function tg = tamgiac(Tp,n)  % n la so chu ki muon ve  % Tp là chu ki xung  Tw=Tp/2;  for i=0:n-1  x=[i\*Tp:0.0001:(i+1)\*Tp]; % vector thoi gian  fx=1-abs(((x-i\*Tp)-Tw)/Tw); % tinh gia tri ham tai tung thoi diem  plot(x,fx)  hold on  end |

****

|  |
| --- |
| %bai 37  Fc=300;  T0=0;  Tf=0.3;  Fs=1000;  Ts=1/Fs;  t=T0:Ts:Tf;  subplot(3,1,1);  s=2\*cos(20\*pi\*t+pi/4)+cos(30\*pi\*t);  plot(t,s,'b','linewidth',1.5);  title('Tin hieu ban dau');  subplot(3,1,2);  phic=pi/4;  y=ammod(s,Fc,Fs,phic);  plot(t,y,'g','linewidth',1.5);  title('Tien hieu dieu che bien do');  subplot(3,1,3);  sr=amdemod(y,Fc,Fs,phic);  plot(t,sr,'r','linewidth',1.5);  title('Tin hieu giai dieu che'); |

**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).**

|  |
| --- |
| %bai 38  h = randi([0, 1], 1, 5000);% Chuoi 5000 bit ngau nhien [0 1];    Tb = 1 /100e6;  t = 0 : Tb : Tb\*(length(h) -1);  NRZ = 2 \* h -1;    rolloff = 0.5;  span = 10;  sps = 4;  rrc\_filter = rcosdesign(rolloff, span, sps);  filtered\_signal = filter(rrc\_filter, 1, upsample(h, sps));    subplot(2, 1, 1);  plot(t(1 : 100), h(1 :100));  xlabel('Time (s)');  ylabel('Amplitude');  title('Signal Before Filtering');    subplot(2, 1, 2);  plot(t(1 : length(filtered\_signal) / sps), filtered\_signal(1 : length(filtered\_signal) / sps));  xlabel('Time (s)');  ylabel('Amplitude');  title('Signal After Filtering');    eyediagram(filtered\_signal, 2 \*sps);  title('Eye Diagram of Filtered Signal'); |

****

|  |
| --- |
| nSamplePoints = 1024;  rN = rand(1, nSamplePoints);  rBinary = round(rN);    Fc = 2;  Fs = 4;  nCycles = 1;  Tb = nCycles/Fc;  t = 0:1/Fs:(nCycles-1/Fs);  xC = cos(2\*pi\*t);      A = 1;  Eb = (A^2\*Tb)/2;  Eb\_N0dB = 0 : 2 : 14;  Eb\_N0 = 10.^(Eb\_N0dB/10);  nVar = (Eb)./ Eb\_N0;    bitStream = [];  carrierSignal = [];  i = 1;    while(i<=nSamplePoints)  if(rBinary(i))  bitStream = [bitStream ones(1,length(xC))];  else  bitStream = [bitStream zeros(1,length(xC))];  end  carrierSignal = [carrierSignal A\*xC];  i = i+1;  end    bits = 2\*(bitStream-0.5);  bpskSignal = carrierSignal.\*bits;  plot(bits);  xlim([0 300]); ylim([-1.2 1.2]);    figure(1);  subplot(3,1,1); plot(bitStream); title('Digital Signal');  xlim([0 300]); ylim([-0.2 1.2]);  subplot(3,1,2); plot(carrierSignal); title('Carrier Signal');  xlim([0 300]); ylim([-1.2 1.2]);  subplot(3,1,3); plot(bpskSignal); title('BPSK modulated signal');  xlim([0 300]); ylim([-1.2 1.2]); |

DẠNG BÀI THỰC HÀNHPhần 1: Matrix manipulation(Ưu tiên sử dụng các phép toán ma trận)  
Nội dung 1: Các câu lệnh liên quan đến tạo ma trận hoặc vecto và xử lýma trận hoăc vecto đơn giảnNote: a, b là hai số cuối của mã SV. Nếu a = 0 thì lấy a =1, nếu b=0 thì lấy b=5, c=b+a-2  
và N = a \* b.

**[1] Tạo ma trận A bất kì sau đó trả về kích thước của ma trận A.**

|  |
| --- |
| a = randi(10);  b = randi(10);  A = randi([-100 100], a, b);  [hang, cot] = size(A); |

**[2] Tạo ma trận A có kích thước là (axb) sau đó chuyển thành B là vecto hang**

|  |
| --- |
| A = randi([-100 100], 6, 8);  B = A(:);  B = B'; |

**[3] Tạo ma trận A có kích thước là (axb) sau đó tạo ma trận B là đường chéo của ma trận A.**

|  |
| --- |
| A = randi([-10 10], 6, 8);  a=6;  b=8;  B = diag(A);  B = B .\* eye(length(B)); |

**[4] Tạo ma trận A có kích thước là (axb) sau đó tìm giá trị lớn nhất và nhỏ nhất của ma trận.**

|  |
| --- |
| A = randi([-100 100], 6, 8);  Max = max(max(A)); % max dau tra ve hang co gia tri max, max 2 tra ve phan tu co gia tri max  Min = min(min(A)); % min dau tra ve hang co gia tri min, min 2 tra ve phan tu co gia tri min |

**[5] Tạo ma trận A có kích thước (axb) có giá trị tăng từ 1 đến axb sau đó tính tổng và trung bình của các phần tử trong ma trận.**

|  |
| --- |
| % A = [];  % k=1;  % sum = 0;  % for i=1:6  % for j=1:8  % A(i,j)=k;  % sum=sum+k;  % k=k+1;  % end  % end  % tb = sum/(6\*8);    % Cach 2  % a = 6;  % b = 8;  % k = a\*b;  % A(1:9) = 1:9;  % B = A';  % SUM1 = sum(B);  % TB1 = mean(B);  % SUM2 = sum(SUM1');  % TB = SUM2/(a\*b);  %Cach 3  A(1:8\*6) = 1:8\*6;  A = reshape(A, 6, 8)';  SUM = sum(sum(A)'); % sum lan 1 tra ve tong cac cot, tiep tuc chuyen vi roi tinh sum lan 2  TB = SUM/(6\*8); |

**[6] Tạo ma trận A là ma trận đơn vị có kích thước là a và ma trận B có kích thước phù hợp và có giá trị toàn b, sau đó ghép 2 ma trận đó với nhau.**

|  |
| --- |
| a = 6;  b = 8;  A = eye(a);  cot = randi(100);  Brow = b\*ones(a, randi(25));  RESULTsameRows = cat(2, A, Brow);  Bcol = b\*ones(randi(25), a);  RESULTsamCol = cat(1, A, Bcol); |

**[7] Tạo ma trận A có kích thước 10x11 sau đó thực hiện:**

**i) Tạo vector B từ hàng a của ma trận A**

**ii) Tạo vector C từ cột b của ma trận A**

**iii) Lấy giá trị phần tử tại vị trí (a,b) gán vào biến z**

|  |
| --- |
| a = 6;  b = 8;  A = randi([-10 10], 11, 10);  B = A(a,:);  C = A(:,b);  z = A(a,b); |

**[8] Tạo ma trận A (3x5) có giá trị nằm trong khoảng từ -2 đến 12 trích xuất ra:**

**i) các phần tử lớn hơn a và gán vào B**

**ii) các phần tử nhỏ hơn hoặc bằng b và gán vào C**

|  |
| --- |
| a = 6;  b = 8;  A = randi([-2 12], 3, 5);  TMP = (A(:))';  GTa = a\*ones(1,length(TMP));  CMPGTa = TMP > GTa;  B = TMP(CMPGTa);  %%%%%%%%%%%%%%  LEb = b\*ones(1, length(TMP));  CMPLEb = TMP <= LEb;  C = TMP(CMPLEb); |

**[9] Tạo ma trận A có kích thước 9x9, sau đó gán phần tử ở hàng a cột b bằng 4. Gán**

**tất cả các phần tử trong ma trận có giá trị là 5 bằng giá trị -5**

|  |
| --- |
| a = 6;  b = 8;  A = randi([0 10], 9, 9);  A(6, 8) = 4;  A(A==5) = -5; |

**[10] Tạo ma trận A có kích thước 9x9, vecto B có 9 phần tử, sau đó ghép với ma trận A, sau đó xóa hàng thứ a, xóa cột thứ b và phần tử axb**

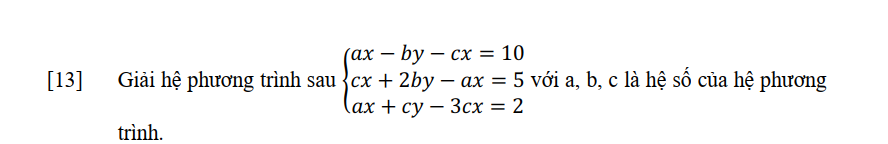
|  |
| --- |
| a = 6;  b = 8;  A = randi([0 10], 9, 9);  B = randi([0 10], 1, 9);  A = cat(1, A, B);  A(a,:) = [];  A(:,b) = [];  A(a,b) = 0; |

**[11] Tạo ma trận A=[1 2; 3 4], mở rộng ma trận A thành ma trận (b+4) hàng và (b+4) cột với phần tử ở hàng thứ (b+4) cột (b+4) có giá trị bằng 10 còn lại các giá trị còn lại bằng a.**

|  |
| --- |
| a = 6;  b = 8;  A = [1 2; 3 4];  B = a\*ones(2, 2);  A = [A B B B B B; B B B B B B; B B B B B B; B B B B B B; B B B B B B; B B B B B B];  A(b+4, b+4) = 10; |

**[12] Tạo ma trận A có kích thước là 9x10, hãy tạo ma trận B có kích thước 5x18 từ ma trận A**

|  |
| --- |
| A = randi([0 10], 9 ,10); % Tao MT A 9x10 =90  B = reshape(A, 5, 18); % Tao MT B 5x18 = 90 |

****

|  |
| --- |
| a = 6; b = 8; c = 12;  A = [a -b -c; c 2\*b -a;a c -3\*c];  B = [10; 5; 2];  Ai = [];  result = [];  if det(A) == 0  disp('PT vo nghiem hoac vo so nghiem');  else  for i=1:size(A,2)  Ai = A;  Ai(:,i) = B;  result(i) = det(Ai)/det(A);  end  end  disp(result'); |

**.**

**[14] Vẽ đồ thị hàm sin(x) và cos(x) trên cùng 1 đồ thị và phân biệt hai hàm số bằng màu sắc. Đánh dấu các điểm tròn tại các vị trí là nguyên lần nửa chu kì của tín hiệu. Ghi chú trục tung, trục hoành, tên đồ thị và tên đường cong tương ứng.**

|  |
| --- |
| x = -2\*pi:pi/50:2\*pi;  y1 = sin(x);  y2 = cos(x);  h1 = plot(x, y1, 'r--');  grid on;  hold on;  h2 = plot(x, y2, 'g--');  hold off;  whitebg('black');  title('Do thi y = sin(x) va y = cos(x)');  ylabel('y = f(x)');  xlabel('x');  ax = gca;  ax.FontSize = 15;  hold on;  xhalf = -2\*pi:pi/2:2\*pi;  yhalf = sin(xhalf);  stem(xhalf, yhalf, 'rp','MarkerFaceColor','red','MarkerEdgeColor','yellow');  yhalf = cos(xhalf);  stem(xhalf, yhalf, 'gp', 'MarkerFaceColor','green', 'MarkerEdgeColor','white');  legend('sin(x)','cos(x)', 'Location', 'southwest'); |

**[15] Vẽ 1 xung hình chữ nhật có chu kỳ T**

|  |
| --- |
| a = 6;  b = 8;  x = -10:0.01:10;  f = @(x0, a0, b0) a0\*rectpuls(x0, b0);    plot(x, f(x, a, b), 'r-');  title('Xung hinh chu nhat T = 8');  axis([-10 10 -1 8]);  xlabel('t');  ylabel('Bien do'); |

**[16] Vẽ 1 xung tam giác có chu kỳ T**

|  |
| --- |
| a = 6;  b = 8;  c = 12;  x = -10:0.01:10;  f = @(x0, a0, b0) a0\*tripuls(x0, b0);    plot(x, f(x, a, b), 'r-');  title('Xung hinh tam giac T = 8');  axis([-10 10 -1 8]);  xlabel('t');  ylabel('Bien do'); |

**Nội dung 2: Viết chương trình để thực hiện các nhiệm vụ cho dưới đây. Yêu cầu: SV**

**viết dưới dạng thủ tục sau đó chuyển sang hàm có thực hiện truyền biến, tuỳ chọn biến đầu vào và đầu ra phù hợp. Các bài toán liên quan đến vòng lặp và tính toán đơn giản.**

**[17] Nhập vào tháng và năm, tính số ngày của tháng đó**

|  |
| --- |
| function day = bai17(month, year)  switch month  case {1, 3, 5, 7, 8, 10, 12}  day = 31;  case {4, 6, 9, 11}  day = 30;  otherwise  if mod(year,4) == 0  day = 29;  else  day = 28;  end  end |

**[18] Tính N!**

|  |
| --- |
| function gt = bai18(n)  gt = 1;  while n>=2  gt = gt\*n;  n = n-1;  end |

**[19] Tìm giá trị lớn nhất của n để tổng 1! + 2! + . . . + 𝑛! bé hơn N\*100.**

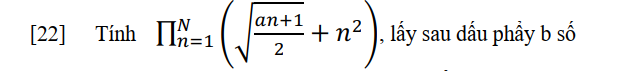
|  |
| --- |
| function nmax = bai19(N)  % 1^2 + 2^2 + ... + n^2 < N\*100  % => VT = n(n+1)(2n+1)/6  % => n(n+1)(2n+1) < N\*600  % nmin = 0, nmax = N  f = @(x) x\*(x+1)\*(2\*x+1);  nmin = 0;  nmax = N\*100;  % Tim kiem nhi phan  while nmin <= nmax  mid = nmin + fix((nmax-nmin)/2);  tmp = f(mid);  if tmp<N\*600  nmin = mid+1;  else  nmax = mid-1;  end  end |

**[20] Tính giá trị thứ N của dãy Fibonnaci**

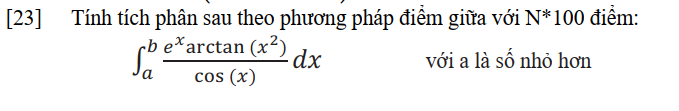
|  |
| --- |
| function fibo = bai20(n)  if n<0  error('n phai la so nguyen duong');  elseif n==1 || n==2  fibo = 1;  else  fibo = bai20(n-1) + bai20(n-2);  end |

****

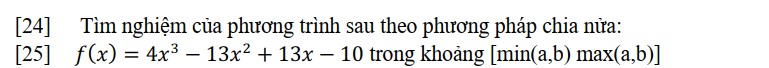
|  |
| --- |
| function sum = bai21(N)  a = 6;  b = 8;  sum = 0;  f = @(n) (-2^a)/exp(-n);  for i=1:N  sum = sum + f(i);  end  sum = sprintf('%.8f', sum); |

****

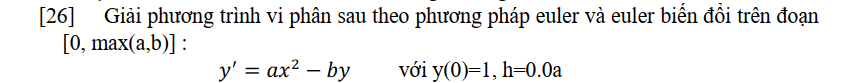
|  |
| --- |
| function mul = bai22(N)  a = 6;  b = 8;  f = @(n) sqrt((a\*n+1)/2) + n^2;  mul = 1;  for i=1:N  mul = mul \* f(i);  end  mul = sprintf('%.8f', mul); |

****

|  |
| --- |
| function tp = bai23(N)  a = 6;  b = 8;  f = @(x) exp(x)\*atan(x^2)/cos(x);  dx = (b-a)/(N\*100);  tp = 0;  for i=1:N  xmid = a + (i - 0.5) \* dx;  tp = tp + f(xmid);  end  tp = tp\*dx;  disp('Gia tri tich phan: '); |

****

|  |
| --- |
| function x = bai25(a, b)  % f: hàm so can giai  f = @(x) 4\*x^3 - 13\*x^2 + 13\*x - 10;  % a, b: ðoan chia nghiem  % epsilon: ðo chinh xác mong muon  epsilon = 0.000001;    fa = f(a);  fb = f(b);  % Vi nghiem o khoang giua, lam cho f(mid) = 0 => 2 khoang a, b o bien trai va phai se  % lam cho f(a) \* f(b) luon nho hon 0  if fa\*fb > 0  disp('Phuong trinh vo nghiem');  else  while b - a > epsilon\*b  % Tim gia tri trung binh cua khoang [a, b]  c = (a + b)/2;  fc = f(c);  % Neu fc = 0, c chinh la nghiem  if fc == 0  x = c;  return;  end  % Neu f(a)\*f(c) > 0, nghiem nam trong khoang [c,b]  if sign(fc) == sign(fa)  a = c;  fa = fc;  % Ngýoc lai, nam trong khoang nghiem trong khoang [a,c]  else  b = c;  fb = fc;  end  end  x = c;  end |

****

|  |
| --- |
| %Giai phuong trinh vi phan sau theo pp euler va euler bien ðoi  %trên ðoan [0, max(a,b)]  function [x\_euler, y\_euler, x\_euler\_bien\_doi, y\_euler\_bien\_doi] = bai26\_euler(a, b, dy)  a = 8;  b = 6;  dy = @(x, y) a\*x^2 - b\*y;  left = 0;  right = max(a, b);  % h = 0.0a  h = 0.08;  N = (right-left)/h;    x\_euler(1) = left;  x\_euler\_bien\_doi(1) = left;    y\_euler(1) = 1; %y[0] = 1;  y\_euler\_bien\_doi(1) = 1;    %% Phuong phap Euler  for i=1:N  y\_euler(i+1) = y\_euler(i) + h\*dy(x\_euler(i), y\_euler(i));  x\_euler(i+1) = x\_euler(i) + h;  end    %% Phuong phap Euler bien doi  for i=1:N  y1 = y\_euler\_bien\_doi(i) + h\*dy(x\_euler\_bien\_doi(i), y\_euler\_bien\_doi(i));  x1 = x\_euler\_bien\_doi(i) + h;  loopcount=0;  diff = 1;  while abs(diff) > 0.05  loopcount = loopcount + 1;  y2 = y\_euler\_bien\_doi(i) + h\*(dy(x\_euler\_bien\_doi(i),y\_euler\_bien\_doi(i)) + dy(x1,y1)/2);  diff = y1 - y2;  y1 = y2;  end  % thu duoc cac cap gia tri cho dau ra  y\_euler\_bien\_doi(i+1) = y1;  x\_euler\_bien\_doi(i+1) = x1;  end    %% Ve do thi  plot(x\_euler, y\_euler, 'g', x\_euler\_bien\_doi, y\_euler\_bien\_doi, 'r');  title('Giai PTVP bac 1 bang pp Euler va Euler bien doi');  xlabel('x');  ylabel('y');  legend('Euler', 'Euler bien doi');  grid on; |

**[27] Vẽ chuỗi (b+2) xung chữ nhật có chu kỳ T**

|  |
| --- |
| function [t, y] = bai27(b, T)  N=b+2; % so xung can ve  fs=1000;  % T do rong xung  t=[]; y=[];  for i=0:N-1  t1= i\*T+1/fs :1/fs : (i+1/2)\*T  y1(1:length(t1)) = 1;  t2= (i+1/2)\*T+1/fs :1/fs : (i+1)\*T;  y2(1:length(t2)) = 0;  t = [t,t1,t2];  y = [y,y1,y2];  end  plot(t,y,'lineWidth', 2);  title(sprintf('Chuoi %d xung chu nhat voi chu ky %.2fs', N, T));  xlabel('time');  ylabel('Amplitude');  grid on; |

**[28] Vẽ chuỗi (b+2) xung tam giác có chu kỳ T**

|  |
| --- |
| function [t, y] = bai28(b, T)  N = b+2; % So xung  %T chu ki xung  %fs tan so xung  fs = 1/T \* 100;  Tw=T/2;  tN=N\*T;  Ts=1/fs;  y=[];  for t=0:Ts:tN  y1=1-abs((mod(t,T)-Tw)/Tw);  y=[y y1];  end  t=0:Ts:tN;  plot(t, y,'LineWidth', 2);  grid on;  title(sprintf('Chuoi tam giac co %d xung voi chu ky %.2fs', N, T));  xlabel('time');  ylabel('Amplitude'); |

**[29] Vẽ chuỗi (b+2) xung hàm exp(x2) có chu kỳ T**

|  |
| --- |
| function [t, y] = bai29(b, T)  N = b+2; %so xung can ve  fs=1e2;  t=0 : 1/fs: N;  dem =1;  fs=100;  f=@(t) exp((t.^2));  t=[];  y=[];  for i=0:N-1  t1=i\*T : 1/fs : (i+1)\*T-1/fs;  st1=f(t1-i\*T);  t=[t,t1];  y=[y,st1];  end  plot(t,y);  grid on;  title(sprintf('%d xung hàm e^{x^2} co chu ki la %.2f', N, T));  xlabel('Thoi gian (s)');  ylabel('Bien do'); |

**[30] Viết hàm xét số đầu vào là số âm hay dương nếu dương thì số đó có phải là số nguyên tố hay không?**

|  |
| --- |
| function bai30(N)  if N<0  disp('So vua nhap la so am');  return;  else  if N<2  disp('So vua nhap khong phai so nguyen to');  return;  end  for i=2:sqrt(N)  if mod(N,i) == 0  disp('So vua nhap khong phai so nguyen to');  return;  end  end  disp('So vua nhap la so nguyen to');  end |

**[31] Tính tích chập của hai hàm số sau: 𝑥(𝑡) = sin (𝑡) và 𝑦(𝑡) = cos (𝑡) trong khoảng**

**[0, N]**

|  |
| --- |
| function tc = bai31(N)  t = 0:0.01:N;  x = sin(t);  y = cos(t);  % Tinh tich chap, 'full' tinh toan ket qua tich chap cho toan bo kich thuoc  % dau vao, ko cat bot kq  tc = conv(x, y, 'full');  plot(tc); |

**Phần 2: Mô phỏng tín hiệu và quá trình phát  
[1] Cho tín hiệu hình sin có chu kỳ T=a, biên độ là 1.5xa. Viết chương trình thực hiện lấy mẫu với tần số lấy mẫu gấp 32 lần tấn số Nyquist của tín hiệu. Vẽ tín hiệu trước và sau khi lấy mẫu**

|  |
| --- |
| % Thiet lap thong so cua tin hieu  a = 8; % Chu ky T  A = 1.5\*a; % Bien do  Fs = 32/(2\*a); % Tan so lay mau  t = 0:1/Fs:2\*a-1/Fs; % Tao vector thoi gian    % Tao tin hieu sin  x = A\*sin(2\*pi\*t/a);    % Ve tin hieu truoc khi lay mau  subplot(2,1,1);  plot(t,x);  title('Tin hieu truoc khi lay mau');  xlabel('Thoi gian (s)');  ylabel('Bien do');    % Lay mau tin hieu  Fs\_new = 32\*2/a; % Tan so lay mau moi  x\_new = resample(x, Fs\_new/Fs, 1);    % Tao vector thoi gian moi  t\_new = 0:1/Fs\_new:2\*a-1/Fs\_new;    % Ve tin hieu sau khi lay mau  subplot(2,1,2);  stem(t\_new, x\_new);  title('Tin hieu sau khi lay mau');  xlabel('Thoi gian (s)');  ylabel('Bien do'); |

**[2] Cho tín hiệu 𝑦 = 𝑒x với x trong khoảng [-a,a]. Viết chương trình mã hóa PCM đều. Vẽ tín hiệu trước và sau khi lượng tử hoá trên cùng một hình.**

|  |
| --- |
| % Thiet lap thong so cua tin hieu  x = -8:0.001:8; % Tao vector x  y = exp(x); % Tao tin hieu y    % Ve tin hieu truoc khi luong tu hoa  plot(x,y);  title('Tin hieu');  xlabel('x');  ylabel('y');  hold on;  axis([4 6 150 500]);  % Thuc hien luong tu hoa  L = 8; % So bit luong tu  delta = (max(y)-min(y))/(2^L); % Tinh bac cua luong tu  y\_quantized = round(y/delta)\*delta; % Luong tu hoa tin hieu y    % Ve tin hieu sau khi luong tu hoa  plot(x,y\_quantized);  legend('Tin hieu truoc khi luong tu hoa', 'Tin hieu sau khi luong tu hoa'); |

**[3] Cho tín hiệu 𝑦 = 𝑒x với x trong khoảng [-a,a]. Viết chương trình thực hiện PCM không đều theo luật A. Vẽ tín hiệu trước và sau khi lượng tử hoá trên cùng một hình.**

|  |
| --- |
| % Thiet lap thong so cua tin hieu  x = -8:0.0001:8; % Tao vector x  y = exp(x); % Tao tin hieu y    % Ve tin hieu truoc khi luong tu hoa  plot(x,y,'b');  title('Tin hieu');  xlabel('x');  ylabel('y');  hold on;  axis([4 6 150 500]);    % Thuc hien luong tu hoa  L = 8; % So bit luong tu  delta = (max(y)-min(y))/(2^L); % Tinh bac cua luong tu  y\_quantized = round(y/delta)\*delta; % Luong tu hoa tin hieu y    % Ve tin hieu sau khi luong tu hoa  plot(x,y\_quantized, 'r');  legend('Tin hieu truoc khi luong tu hoa', 'Tin hieu sau khi luong tu hoa'); |

**[4] Cho tín hiệu 𝑦 = 𝑒x với x trong khoảng [-a,a]. Viết chương trình nén PCM không đều theo luật 𝜇. Vẽ tín hiệu trước và sau khi lượng tử hoá trên cùng một hình**

|  |
| --- |
| % Thiet lap thong so cua tin hieu  x = -8:0.001:8; % Tao vector x  y = exp(x); % Tao tin hieu y    % Ve tin hieu truoc khi luong tu hoa  plot(x,y);  title('Tin hieu');  xlabel('x');  ylabel('y');  hold on;    % Thuc hien luong tu hoa  L = 8; % So bit luong tu  mu = 255; % Luat mu  delta = zeros(size(y)); % Tao vector chua cac bac luong tu  y\_quantized = zeros(size(y)); % Tao vector chua tin hieu luong tu hoa  for i = 2:length(y)  delta(i) = (y(i)-y\_quantized(i-1))/mu; % Tinh bac cua luong tu  y\_quantized(i) = y\_quantized(i-1) + round(delta(i))\*mu; % Luong tu hoa tin hieu y  end    % Ve tin hieu sau khi luong tu hoa  plot(x,y\_quantized);  legend('Tin hieu truoc khi luong tu hoa', 'Tin hieu sau khi luong tu hoa'); |

**[5] Cho chuỗi bit sau: a =[1100101100101011111110010110]. Viết chương trình thực hiện mã hoá chuỗi bit a theo dạng sau**

**i) Mã NRZ**

|  |
| --- |
| d = [1 1 0 0 1 0 1 1 0 0 1 0 1 0 1 1 1 1 1 1 1 0 0 1 0 1 1 0];  R = 6e6;  Ns = 6e6;  [t,y,code] = nrzcode(d,R,Ns,'pol');  plot(t,y);  ylim([-2 2]);  function [t,y,code] = nrzcode(d,R,Ns,type)  % Chuong trinh 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' (khong doi cuc, ko co phan am) or polar - 'pol')  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  end |

**ii) RZ 50%**

|  |
| --- |
| d = [1 1 0 0 1 0 1 1 0 0 1 0 1 0 1 1 1 1 1 1 1 0 0 1 0 1 1 0];  R = 6e6;  Ns = 6e6;  [t,y,code] = rzcode(d,R,Ns,'pol');  plot(t,y);  ylim([-2 2]);    function [t,y,code] = rzcode(d,R,Ns,type)  % Chuong trinh vi du ve ma duong truyen RZ  % 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' (khong doi cuc, ko co phan am) or polar - 'pol')  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'  if d(n) == 1  y(k) = 1;  else  y(k) = -0.5;  end  code(n) = d(n);  case 'pol'  if d(n) == 1  y(k) = 1;  else  y(k) = -1;  end  code(n) = 2\*d(n)-1;  end  end  end |

**iii) AMI**

|  |
| --- |
| d = [1 1 0 0 1 0 1 1 0 0 1 0 1 0 1 1 1 1 1 1 1 0 0 1 0 1 1 0];  R = 6e3;  Ns =20;  [t,y,code] = amicode(d,R,Ns);  stairs(t,y);  ylim([-2 2]);    function [t,y,code] = amicode(d,R,Ns)  % Chuong trinh 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  % code - the AMI code  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 = zeros(size(d));  prev = 0; % Bien luu cac gia tri truoc do  for k = 1:Ns  n = fix(t(k)/Tb)+1;  if n >= Nb  n = Nb;  end  if d(n) == 0  y(k) = 0;  code(n) = 0;  else  if prev == 0  y(k) = 1;  code(n) = 1;  prev = 1;  else  y(k) = -1;  code(n) = -1;  prev = 0;  end  end  end  end |

**[6] Viết chương trình điều chế AM cho tín hiệu tương tự 𝑠(𝑡) = 𝑎 ∗ cos (2𝜋𝑓𝑡 + 𝑏), f=N\*100 với sóng mang 𝑐(𝑡) = 𝑏 ∗ sin (2𝜋𝑓c𝑡 + 𝑏), 𝑓c = 1000 ∗ 𝑓**

|  |
| --- |
| % Chuong trinh vi du ve dieu che AM  %% Set parameters  a = 8;  b = 6;  N = 48;  % Message  A = a;  f = N\*100; % frequency [Hz]  % phi = -pi/4; % Phase [rad]  % Carrier  m = 0.5; % modulation index  Ac = A/m; % amplitude  fc = 1000\*f; % frequency [Hz]  % phi\_c = 0; % Phase [rad]  Ns = 2^9; % number of samples  T0 = 0; % start time [s]  Tf = 5e-3; % end time [s]  Ts = (Tf-T0)/(Ns-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+b); % message signal  xc = b\*sin(2\*pi\*fc\*t+b); % carrier signal  % Modulation  y = (1+x/Ac).\*xc;  figure(1)  plot(t,x);  title('Tin hieu tuong tu');  xlabel('Thoi gian');  ylabel('Bien do');  figure(2)  plot(t,xc);  title('Tin hieu song mang');  xlabel('Thoi gian');  ylabel('Bien do');  figure(3)  plot(t,y);  title('Dieu che bien do AM');  xlabel('Thoi gian');  ylabel('Bien do'); |

**[7] Viết chương trình điều chế PM cho tín hiệu tương tự 𝑠(𝑡) = 𝑎 ∗ cos (2𝜋𝑓𝑡 + 𝑏), f=N\*100 với sóng mang 𝑐(𝑡) = 𝑏 ∗ sin (2𝜋𝑓c𝑡 + 𝑏), 𝑓c = 1000 ∗ 𝑓**

|  |
| --- |
| N = 48; % Gia tri N  f = N \* 100; % Tan so tin hieu  fc = 1000 \* f; % Tan so song mang  b = 6; % Pha ban dau  a= 8;    Ns = 2^7; % number of samples  T0 = 0; % start time [s]  Tf = 5e-3; % end time [s]  Ts = (Tf-T0)/(Ns-1); % sampling period  fs = 1/Ts; % sampling frequency [Hz]  t = T0:Ts:Tf; % time vector    s = a \* cos(2\*pi\*f\*t + b); % Tin hieu goc  c = b \* sin(2\*pi\*fc\*t + b); % Song mang  % Dieu che PM  kf = 2\*pi\*f; % He so do bien thien tan so  phi\_pm = cumsum(kf \* s); % Tich phan tin hieu de thu pha PM    PM = cos(2\*pi\*fc\*t + phi\_pm); % Tin hieu dieu che pha PM      subplot(3, 1, 1);  plot(t, s);  title('Tin hieu goc');  xlabel('Thoi gian');  ylabel('Bien do');    subplot(3, 1, 2);  plot(t, c);  title('Song mang');  xlabel('Thoi gian');  ylabel('Bien do');    subplot(3, 1, 3);  plot(t, PM);  title('Tin hieu dieu che PM');  xlabel('Thoi gian');  ylabel('Bien do'); |

**[8] Viết chương trình điều chế FM cho tín hiệu tương tự 𝑠(𝑡) = 𝑎 ∗ cos (2𝜋𝑓𝑡 + 𝑏), f=N\*100 với sóng mang 𝑐(𝑡) = 𝑏 ∗ sin (2𝜋𝑓c𝑡 + 𝑏), 𝑓c = 1000 ∗ 𝑓**

|  |
| --- |
| N = 48; % Gia tri N  f = N \* 100; % f tin hieu  fc = 1000 \* f; % f song mang  b = 6; % Pha ban dau  kf = 0.1; % He so bien thien tan so    Ns = 2^9; % number of samples  T0 = 0; % start time [s]  Tf = 5e-3; % end time [s]  Ts = (Tf-T0)/(Ns-1); % sampling period  fs = 1/Ts; % sampling frequency [Hz]  t = T0:Ts:Tf; % time vector    s = 8 \* cos(2\*pi\*f\*t + b); % Tin hieu goc  c = 6 \* sin(2\*pi\*fc\*t + b); % Song mang  plot(t,c);  % Ðieu che FM  fm = kf \* s; % Tan so bien thien FM    FM = cos(2\*pi\*fc\*t + cumsum(fm)); % Tin hieu ðieu che FM    subplot(3, 1, 1);  plot(t, s);  title('Tin hieu goc');  xlabel('Thoi gian');  ylabel('Bien do');    subplot(3, 1, 2);  plot(t, c);  title('Song mang');  xlabel('Thoi gian');  ylabel('Bien do');    subplot(3, 1, 3);  plot(t, FM);  title('Tin hieu dieu che FM');  xlabel('Thoi gian');  ylabel('Bien do'); |

**[9] Viết chương trình điều chế số ASK cho chuỗi bit đầu vào với số lượng bit là N\*10 bit nếu N chẵn và (N+1)\*10 nếu N lẻ.**

|  |
| --- |
| % Bit vao  N = 48;  bit\_sequence = randi([0 1], 1 ,N);    % Thong so dieu che ASK  bit\_rate = 1000; % Toc do bit  carrier\_freq = 10000; % Tan so song mang  amplitude\_0 = 0; % Bien do khi bit = 0  amplitude\_1 = 1; % Bien do khi bit = 1    % Thoi gian mau hoa  sampling\_freq = 10 \* bit\_rate; % Tan so lay mau  sampling\_period = 1 / sampling\_freq; % Chu ky lay mau  t = 0:sampling\_period:(length(bit\_sequence) \* sampling\_period); % Truc thoi gian    % Tao tin hieu ASK  ask\_signal = zeros(size(t)); % Khoi tao tin hieu ASK ban dau    for i = 1:length(bit\_sequence)  if bit\_sequence(i) == 0  ask\_signal((i-1)\*sampling\_period <= t & t < i\*sampling\_period) = amplitude\_0;  else  ask\_signal((i-1)\*sampling\_period <= t & t < i\*sampling\_period) = amplitude\_1;  end  end    % Hien thi do thi tin hieu ASK  subplot(2, 1, 1);  plot(t, ask\_signal);  title('Tin hieu ASK');  xlabel('Thoi gian (s)');  ylabel('Bien do');    % Giai dieu che ASK  demodulated\_bits = zeros(1, length(bit\_sequence)); % Khoi tao chuoi bit giai dieu che    for i = 1:length(bit\_sequence)  if ask\_signal((i-1)\*sampling\_period <= t & t < i\*sampling\_period) == amplitude\_0  demodulated\_bits(i) = 0;  else  demodulated\_bits(i) = 1;  end  end    % Hien thi chuoi bit giai dieu che  subplot(2, 1, 2);  stem(demodulated\_bits);  title(' chuoi bit giai dieu che');  xlabel('Thoi gian (s)');  ylabel('Bit');    % Tao chuoi bit dau vao va chuoi bit giai dieu che  input\_bits = mat2str(bit\_sequence);  output\_bits = mat2str(demodulated\_bits);    % Hien thi ket qua  disp(['Bit input : ' input\_bits(2:end-1)]);  disp(['Bit output: ' output\_bits(2:end-1)]); |

**[10] Viết chương trình điều chế số M-PSK (M là bậc điều chế) cho chuỗi kí tự gồm N\*50 phần tử. Tín hiệu sau khi điều chế đi qua kênh AWGN có SNR = (2xa) dB. Vẽ giản đồ chòm sao tín hiệu sau điều chế và sau khi đi qua kênh.**

|  |
| --- |
| % Thong so dau vao  M = 16; % Bac dieu che  k = log2(M); % So bit tren moi mau tin hieu  N = 48\*50; % So mau tin hieu  SNR = 16; % SNR (dB)  a = random\_string - '0';  symbols = randi([0 M-1], [1 N]); % Chuoi cac mau tin hieu ngau nhien    % Dieu che tin hieu M-PSK  modulated = pskmod(symbols,M);  % Them nhieu AWGN vao tin hieu  noisy = awgn(modulated, SNR, 'measured');  % Ve gian do chom sao cua tin hieu dieu che  figure;  subplot(2,1,1);  scatter(real(modulated), imag(modulated), 'filled');  xlabel('Phan thuc');  ylabel('Phan ao');  title('Gian do chom sao cua tin hieu sau khi dieu che');    % Ve gian do chom sao cua tin hieu sau khi AWGN  subplot(2,1,2);  scatter(real(noisy), imag(noisy), 'filled');  xlabel('Phan thuc');  ylabel('Phan ao');  title(['Gian do chom sao cua tin hieu sau khi AWGN (SNR = ', num2str(SNR), ' dB)']); |

**[11] Viết chương trình điều chế số M-QAM (M là bậc điều chế) cho chuỗi kí tự gồm N\*50 phần tử. Tín hiệu sau khi điều chế đi qua kênh AWGN có công suất nhiễu N0= b. Vẽ giản đồ chòm sao tín hiệu sau điều chế và sau khi đi qua kênh.**

|  |
| --- |
| % Thong so dau vao  M = 64; % Bac dieu che  k = log2(M); % So bit tren moi mau tin hieu  N = 48\*50; % So mau tin hieu  EbNo = 10; % Ty so suy giam nang luong bit  N0 = 6; % Cong suat nhieu  symbols = randi([0 M-1],[1 N]); % Chuoi cac mau tin hieu ngau nhien    % Dieu che tin hieu M-QAM  modulated = qammod(symbols, M);    % Them nhieu AWGN vao tin hieu  noisy = awgn(modulated, 10\*log10(k\*EbNo\*N0), 'measured');  % 10\*log10(k\*EbNo\*N0) la SNR, va 'measured' tinh toan lai SNR thuc te    % Ve gian do chom sao cua tin hieu dieu che  figure;  subplot(2,1,1);  scatter(real(modulated), imag(modulated), 'filled');  xlabel('Phan thuc');  ylabel('Phan ao');  title('Gian do chom sao cua tin hieu sau khi dieu che');    % Ve gian do chom sao cua tin hieu sau khi AWGN  subplot(2,1,2);  scatter(real(noisy), imag(noisy), 'filled');  xlabel('Phan thuc');  ylabel('Phan ao');  title(['Gian do chom sao cua tin hieu sau khi AWGN (Eb/No = ', num2str(EbNo), ' dB)']); |

**[12] Tạo chuỗi bit ngẫu nhiên có độ dài (ax100) bit. Chuyển đổi chuỗi bit này thành dạng sóng NRZ lưỡng cực tại tốc độ (ax10) Mb/s. Sử dụng bộ lọc raise-cosine có độ rộng băng tần gấp 3 lần tốc độ tín hiệu, hệ số roll-off bằng 0,5 để lọc chuỗi tín hiệu này. Vẽ dạng sóng biểu diễn dạng sóng tín hiệu trước và sau khi lọc trên 10 chu kì bit và biểu đồ mắt của tín hiệu trước và sau khi lọc trên cửa sổ 2 chu kì bit.**

|  |
| --- |
| % Tao bit  a = 8;  bit\_stream = randi([0 1], 1, a\*100);    % Toc do tin hieu va kich thuoc lay mau  sampling\_rate = a\*1e7; % Hz  samples\_per\_bit = 1;    % Tao tin hieu NRZ luong cuc  nrz = 2 \* bit\_stream - 1;  n = (0:length(nrz)-1) / sampling\_rate;    % Ve do thi truoc khi loc  subplot(2, 1, 1);  plot(n(1:10\*samples\_per\_bit), nrz(1:10\*samples\_per\_bit));  title('Dang tin hieu truoc khi loc');  xlabel('Thoi gian (s)');  ylabel('Bien do');    % Loc tin hieu voi bo loc raise-cosine  tx\_filter = rcosdesign(0.5, 6, samples\_per\_bit);  tx\_filtered = conv(nrz, tx\_filter, 'same');    % Ve tin hieu sau loc  subplot(2, 1, 2);  plot(n(1:10\*samples\_per\_bit), tx\_filtered(1:10\*samples\_per\_bit));  title('Dang tin hieu sau khi loc');  xlabel('Thoi gian (s)');  ylabel('Bien ?o');    % Bieu do mat truoc khi loc  eyediagram(nrz, 2\*samples\_per\_bit);    % Bieu do mat sau khi loc  eyediagram(tx\_filtered, 2\*samples\_per\_bit); |

**[13] Tạo chuỗi bit ngẫu nhiên có độ dài (ax100) bit. Chuyển đổi chuỗi bit này thành dạng sóng NRZ lưỡng cực tại tốc độ (ax10) Mb/s. Sử dụng bộ lọc butterworth có độ rộng băng tần gấp 2,5 lần tốc độ tín hiệu để lọc chuỗi tín hiệu này. Vẽ dạng sóng biểu diễn dạng sóng tín hiệu trước và sau khi lọc trên 10 chu kì bit và biểu đồ mắt của tín hiệu trước và sau khi lọc trên cửa sổ 2 chu kì bit.**

|  |
| --- |
| a = 8;  bit\_stream = randi([0 1], 1, a\*100);    % Signal frequency and sampling rate  signal\_freq = a\*1e3; % Hz  sampling\_rate = a\*10e3; % Hz  samples\_per\_bit = sampling\_rate/signal\_freq;    % Convert binary stream to NRZ format  nrz = 2 \* bit\_stream - 1;  n = (0:length(nrz)-1) / sampling\_rate;    % Design Butterworth filter  [b, a] = butter(6, signal\_freq\*2.5/(sampling\_rate/2));    % Filter the signal  filtered\_signal = filter(b, a, nrz);    % Plot before and after filtering  subplot(2,1,1);  plot(n(1:10\*samples\_per\_bit), nrz(1:10\*samples\_per\_bit));  title('Signal before filtering');  xlabel('Time (s)');  ylabel('Amplitude');  subplot(2,1,2);  plot(n(1:10\*samples\_per\_bit), filtered\_signal(1:10\*samples\_per\_bit));  title('Signal after filtering');  xlabel('Time (s)');  ylabel('Amplitude');    % Eye diagrams  eyediagram(nrz, 2\*samples\_per\_bit);  eyediagram(filtered\_signal, 2\*samples\_per\_bit); |

**BÀI TẬP LỚN**

1. **Mô tả nhiệm vụ:**

Mô phỏng hệ thống truyền dẫn số tại tốc độ dữ liệu N Mb/s, với N là số cuối cùng của mã số sinh viên (nếu số đó là 0 thì sẽ lựa chọn số liền kề bên cạnh). Nguồn tin của hệ thống được lấy từ file nhạc thực hiện trong nhiệm vụ 1, trong trường hợp không thực hiện lấy nguồn tin từ nhiệm vụ 1 được hãy thay thế bằng một chuỗi tín hiệu nhị phân ngẫu nhiên tương đương. Mỗi sinh viên sẽ lựa chọn một trong các kĩ thuật điều chế sau cho hệ thống mô phỏng của mình:

* Điều chế M-QAM nếu số cuối cùng trong mã sinh viên là lẻ, với M = 16 nếu số liền kề là lẻ và M = 4 nếu số liền kề là chẵn.
* Điều chế M-DPSK nếu số cuối cùng trong mã sinh viên là chẵn, với M = 4 nếu số liền kề là lẻ và M = 2 nếu số liền kề là chẵn.  
  Sử dụng mô hình mô phỏng tương đương băng gốc, tín hiệu phát có thể được biểu diễn như sau:



trong đó *dk* là các kí hiệu (symbol) phức được xác định từ chuỗi bản tin đầu vào và kỹ thuật điều chế; *Tsym* là chu kỳ của symbol; ϕ*0* là pha của tín hiệu phát và *p(t)* xác định dạng xung được phát, với:  
 cho tín hiệu M-DPSK  
 cho tín hiệu M-QAM  
với *Es* là năng lượng mỗi symbol.

* **Hệ thống 2-DPSK với N = 6 Mb/s.**

* **Xác định tham số của hệ thống 2-DPSK:**
* Tốc độ dữ liệu Rb = 6 Mb/s.
* Năng lượng symbol Es = 10 J.
* Tần số lấy mẫu Fs = 300 KHz.
* Tần số sóng mang Fc = 10 KHz.
* SNR = 5 dB.
* BER tại các mức SNR (Es/No) = [2 5 10] dB.
* Pha ban đầu của tín hiệu ϕ0 = 0.

**Bằng việc sử dụng MATLAB, viết chương trình mô phỏng hệ thống truyền dẫn số sử dụng kỹ thuật điều chế đã lựa chọn trên kênh AWGN với nguồn tín hiệu là tín hiệu thu được ở nhiệm vụ 1. Ước tính xác suất lỗi tại các mức tỉ số tín hiệu trên nhiễu SNR lần lượt bằng 2, 5 và 10 dB theo phương pháp Monte Carlo.**

**Code:**

|  |
| --- |
| %% -----Nguyen Di Dan - B20DCVT086 (2-DPSK N= 6MB/s) ---------------  clear,clc,close all  M = 2; % So muc dieu che  Es =10; % Nang luong cho mot ky hieu  Rb=6e6; % Toc do bit- bit rate  Nsym=Rb/log2(M); % Toc do ki hieu- Symbol rate  Tsym=1/Nsym;% Chu k? kí hieu  n = 1e4; % So bit dau vao  phi=0; % Pha tin hieu phat  fs=3e5; % Tan so lay mau  fc=1e4; % Tan so song mang  ts=1/fs; % Chu ky lay mau  bf = 7e6; % Bang thong bo loc  %% Nhap tin hieu phat  bit = randi([0 1], 1, n); % Chuoi bit dau vao ngau nhien  dec = bi2de(reshape(bit,log2(M),length(bit)/log2(M)).','left-msb'); % Chuyen doi nhi phan sang thap phan.  d = dpskmod(dec,M); % Dieu che 2-DPSK  %% Xay dung ham s(t)  t=0:ts:n/log2(M)\*Tsym;  for i = 1:length(t)  s(i) = 0;  for k = 1:n/log2(M)  pt(k) = sqrt((2\*Es)/Tsym)\*((1-cos(2\*pi\*t(i)))/Tsym);  s(i) = s(i)+ d(k)\*pt(k);  end  end  %% Do thi tin hieu bang goc  plot(t,s);  xlabel(' Thoi gian (s)');  ylabel( ' Bien do ');  title(' Do thi tin hieu goc ');  signalA=s.\*exp(1i\*phi); %tin hieu sau dieu che  signalE=signalA.\*exp(1i\*2\*pi\*fc\*t); %Tin hieu de xac dinh mau mat  %% Do thi va pho cua tin hieu sau dieu che 2-DPSK  figure  subplot(2,1,1);  plot(t,real(signalE))  xlabel(' Thoi gian (s)');  ylabel( ' Bien do ');  title('Do thi tin hieu sau dieu che');  subplot(212);  spectrum(signalE,fs);  title('Pho cua tin hieu sau dieu che');  %% Mau mat cua tin hieu sau khi dieu che 2-DPSK  eyediagram(real(signalE),10);  title(' Bieu do mat tin hieu 2-DPSK');  %% Chom sao cua tin hieu sau khi dieu che  scatterplot(d,1,0,'or');  title(' Bieu do chom sao cua tin hieu sau dieu che ' );  %% Do thi va pho cua tin hieu sau khi qua AWGN  figure  signal\_noise=addNoiseAWGN(signalE,5);  subplot(211);  plot(t, real(signal\_noise));  xlabel(' time s'); ylabel( ' Bien do ');  title(' Do thi khi bi AWGN ');  subplot(212);  spectrum(signal\_noise,fs);  title('Pho cua tin hieu bi AWGN');  %% Mau mat cua tin hieu sau khi qua kenh AWGN  eyediagram(real(signal\_noise),10);  title('Bieu do mat tin hieu 2-DPSK khi qua AWGN');    %% Chom sao cua tin hieu sau khi qua kenh AWGN va so sanh voi chom sao goc  d\_noise = addNoiseAWGN(d,5);  h = scatterplot(d\_noise,1,0,'x');  hold on;  scatterplot(d,1,0,'or',h);  title('Bieu do chom sao cua tin hieu sau khi qua kenh AWGN');  %% Xu ly va khoi phuc tai bo thu  figure  source1 = signal\_noise.\*exp(-1i\*phi).\*exp(-1i\*2\*pi\*fc\*t); %Tin hieu thu truoc bo loc  source = raisedCosFilter(source1,bf,ts,2); % Khoi phuc lai tin hieu  plot(t,real(source)) % Do thi tin hieu sau khi khoi phuc va pho cua no  xlabel('Thoi gian(s)');  ylabel('Bien do');  title('Do thi tin hieu khoi phuc');  figure  spectrum(source,fs);  title('Pho cua tin hieu duoc khoi phuc');  %% Chom sao tin hieu duoc khoi phuc  h = scatterplot(source,1,0,'xb');  title('Bieu do chom sao cua tin hieu duoc khoi phuc');  %% Mau mat tin hieu sau khi khoi phuc  eyediagram(real(source),10);  title('Mau mat duoc khoi phuc');  %% Tinh BER bang phuong phap Monte-Carlo  SNR\_dB=[2 5 10] ;  %======vong lap=====%  for i=1:length(SNR\_dB)  SNR=exp(SNR\_dB(i)\*log(10)/10); %Chuyen dB sang so lan  theoryBer(i)=2\*qfunc(sqrt(SNR)); %Tinh BER theo ly thuyet  simBer(i)=monteCarlo(SNR\_dB(i), n, bit); %Tinh BER theo thuc nghiem  end  % Uoc tinh xac suat loi tai BER = [ 2 5 10 ]  disp('Ket qua BER tai SNR lan luot la 2 5 va 10dB la')  simBer = single(simBer) %Lam tron 5 so.  %Do thi  semilogy(SNR\_dB,theoryBer,'LineWidth',2);hold on;  semilogy(SNR\_dB,simBer,'r-\*','LineWidth',2);grid on;  xlabel('SNR (dB)'); ylabel('BER');  title(' Truyen tin hieu DPSK qua kenh AWGN');  legend('BER\_l\_y\_ \_t\_h\_u\_y\_e\_t','BER\_m\_o\_ \_p\_h\_o\_n\_g');  %% ============== FUNCTION ======================  %% ======= Ham them kenh nhieu AWGN =============== %%  function yNoise = addNoiseAWGN(y,SNR\_dB)  % y - Tin hieu dau vao  % SNRdB - Muc SNR dB  % yNoise – Tin hieu nhieu dau ra  SNR = 10^(SNR\_dB/10);  VarN = var(y)/SNR;  if (isreal(y))  yNoise = y + sqrt(VarN)\*randn(size(y));  else  yNoise = y + sqrt(VarN/2)\*(randn(size(y))+1i\*randn(size(y)));  end  end    %% ==== Ham them kenh nhieu Gauss ============== %%  function yNoise = addNoiseGauss(y, SNR\_dB)  % y - Tin hieu dau vao  % SNR\_db - Ty le tin hieu nhieu dB  % yNoise - Tin hieu dau ra qua nhieu  SNR\_lin = 10^(SNR\_dB/10);  noise = randn(size(y));  power\_signal = mean(abs(y).^2);  power\_noise = power\_signal/SNR\_lin;  noise = noise\*sqrt(power\_noise);  yNoise = y + noise;  end  %% ==== Ham xu ly bo loc (Dung bo loc Raised Cosine ======= %%  function y = raisedCosFilter(x,bf,Ts,beta)  % x - dau vao  % bf - bang thong cua bo loc  % Ts - chu ky lay mau  % beta - he so giam doc (rolloff factor)  % y - dau ra bo loc  Ns = length(x);  Tb = 1/bf;  beta = beta\*bf;  % Mien tan so  f = [0:Ns/2-1 -Ns/2:-1]/(Ns\*Ts);  Xf = fft(x);  Yf = zeros(size(Xf));  ind = (abs(f) <= (bf/2-beta));  Yf(ind) = Xf(ind).\*Tb;  ind = (abs(f) <= (bf/2+beta) & abs(f) > (bf/2-beta));  Yf(ind) = Xf(ind).\*(Tb\*cos(pi/(4\*beta)\*(abs(f(ind))-bf/2+beta)).^2);  ind = (abs(f) > (bf/2+beta));  Yf(ind) = Xf(ind).\*0;  % Chuyen doi sang mien thoi gian  y = ifft(Yf)./Tb;  end    %% ======== Ham tinh BER bang Monte Carlo ============ %%  function berMonteCarlo = monteCarlo(SNR\_dB, n, bit)  % SNR\_dB: Ty le tin hieu nhieu  % n: So lan lap danh gia hieu nang  % berMonteCarlo: BER trung binh dua tren ky thuat Monte Carlo  M = 2; % 2-DPSK  % BER trung binh  berSum = 0;  for i = 1:n  % Tin hieu dieu che  bitmodulated = dpskmod(bit, M);  % Tao ra tin hieu qua nhieu Gauss  signal\_noise = addNoiseGauss(bitmodulated, SNR\_dB);  % Giai dieu che tin hieu  bitdemodulated = dpskdemod(signal\_noise, M);  % Tinh so luong bit sai khac giua bit ban dau va bit sau giai dieu che  num\_errors = sum(bit ~= bitdemodulated);  % Tinh loi bit  berSum = berSum + num\_errors/n;  end  berMonteCarlo = berSum/n;  end      %% ========= Ham ve pho tin hieu ======================== %%  function st\_fft\_fre = spectrum(x,fs)  % pho cua tin hieu x  % fs - tan so lay mau  st\_fft = fft(x);  st\_fft = fftshift(st\_fft);  st\_fft\_fre = fs/2\*linspace(-1, 1, length(st\_fft));  plot(st\_fft\_fre, abs(st\_fft));  grid;  xlabel('Tan so (Hz)');  ylabel('Bien do');  end |

**Ngân hàng mô phỏng :**

***Dạg câu 3 điểm: vi phân, tích phân, tính gần đúng*Bài 10:** Cho hàm arctan(x) tính gần đúng bằng việc khai  
triển theo chuỗi Taylor như sau : arctan(x) **=** đối với |x|<1 Viết chương trình MatlaB để tính gần đúng hàm arctan(x) với x= 0,8 theo công thức trên để đạt được độ chính xác 1e-4 hiển thị số vòng lặp được thực hiện. Dựa vào giải thuật của chương trình hãy tính giá trị hàm gần đúng của 5 vòng lặp đầu tiên.

|  |
| --- |
| **x=0.8; epsilon=1; n=0; arta=0; f=@(n)(((-1)^n)/(2\*n+1))\*x^(2\*n+1);** *%nhap ham* **while (epsilon >1e-4) arta=arta+f(n); epsilon =abs(arta-atan(x)); n=n+1; end disp(‘so vong lap la : ’ ); disp(n); disp(‘gia tri gan dung arctan la : ’ ); disp(arta); arta1 =0; disp(‘gia tri gan dung 5 vong lap dau tien la :’ ); for k=0:4**  **arta1 = arta1+f(k); disp(arta1);**  **end** |

***Dạng câu 3 điểm : tạo tín hiệu chuỗi xung*Bài 11:**Viết *function* MatlaB để tạo chuỗi xung với cấu trúc như sau: ***function*[t,y]= *chuoixung*( R,Np ),** trong đó R là tốc độ xung , Np là số lượng xung trong chuỗi, t và y là các vector thời gian và mẫu chuỗi tín hiệu đầu ra tương ứng. Biết chuỗi xung được lấy mẫu tại tần số **fs= 16R.** Biết trong một chu kỳ xung được mô tả bởi  **** với 0 ≤ t ≤ Tp và **Tp = 1/R**. Sử dụng *function* này để vẽ một chuỗi xung gồm **5** (Np=5)xung tại tốc độ **10kHz(R=1e4).**

|  |
| --- |
| **function[t,st] = chuoixung(R,Np) Tp=1/R; Tw=Tp/2; fs=16\*R; f=@(t)exp(t/(2\*Tp))-1; t=[]; st=[]; for i=0:Np-1 t1=[i\*Tp:1/fs:(i+1)\*Tp]; st1=f(t1-i\*Tp); t=[t,t1];**  **st=[st,st1];**  **end**  *Chương trình chính* **close all; clear all;**  **R=1e4;**  **Np=5;**  **fs=16\*R;**  **[t,y]=chuoixung(R,Np); f=0:1/(max(t)-min(t)):fs; yf1=fft(y,length(y)); yf2=fftshift(yf1); plot(t,y); figure; plot(f,yf2(1:length(f)));** |

**Bài 12:** Viết *function* MatlaB để tạo chuỗi xung với cấu trúc như sau: ***function*[t,y]= *chuoixung*( R,Np ),** trong đó R là tốc độ xung , Np là số lượng xung trong chuỗi, t và y là các vector thời gian và mẫu chuỗi tín hiệu đầu ra tương ứng. Biết chuỗi xung được lấy mẫu tại tần số **fs= 16R.** Biết trong một chu kỳ xung được mô tả bởi với 0 ≤ t ≤ Tp và **Tp = 1/R**. Sử dụng *function* này để vẽ một chuỗi xung gồm **5** (Np=5)xung tại tốc độ **1kHz(R=1e3).** Sau đó tính toán và biểu diễn phổ của chuỗi xung được tạo ra bằng Matlab

|  |
| --- |
| **function[t,st] = chuoixung(R,Np) Tp=1/R; Tw=Tp/2; fs=16\*R; f=@(t)exp(-4\*(t)/Tp); t=[]; st=[]; for i=0:Np-1 t1=[i\*Tp:1/fs:(i+1)\*Tp]; st1=f(t1-i\*Tp);** **t=[t,t1]; st=[st,st1]; end** *Chương trình chính* **close all; clear all; R=1e3; Np=5; fs=16\*R; [t,y]=chuoixung(R,Np); f=0:1/(max(t)-min(t)):fs; yf1=fft(y,length(y)); yf2=fftshift(yf1); plot(t,y); figure; plot(f,yf2(1:length(f)));** |

**Bài 13:**Viết *function* MatlaB để tạo chuỗi xung với cấu trúc như sau: ***function*[t,y]= *chuoixung*( R,Np ),** trong đó R là tốc độ xung , Np là số lượng xung trong chuỗi, t và y là các vector thời gian và mẫu chuỗi tín hiệu đầu ra tương ứng. Biết chuỗi xung được lấy mẫu tại tần số **fs= 16R.** Biết trong một chu kỳ xung được mô tả bởi  với 0 ≤ t ≤ Tp và **Tp = 1/R**. Sử dụng *function* này để vẽ một chuỗi xung gồm **5** (Np=5)xung tại tốc độ **1MHz(R=1e6).** Sau đó tính toán và biểu diễn phổ của chuỗi xung được tạo ra bằng MatlaB

|  |
| --- |
| **function[t,st] = chuoixung(R,Np) Tp=1/R; Tw=Tp/2; fs=16\*R;**  **f=@(t)exp(-25\*(t-Tp)/(Tp\*Tp)); t=[]; st=[]; for i=0:Np-1 t1=[i\*Tp:1/fs:(i+1)\*Tp]; st1=f(t1-i\*Tp); t=[t,t1]; st=[st,st1]; end** *Chương trình chính* **close all; clear all; R=1e6; Np=5; fs=16\*R; [t,y]=chuoixung(R,Np); f=0:1/(max(t)-min(t)):fs; yf1=fft(y,length(y)); yf2=fftshift(yf1); plot(t,y);** **figure; plot(f,yf2(1:length(f)));** |

**Bài 14:** Viết *function* MatlaB để tạo chuỗi xung với cấu trúc như sau: ***function*[t,y]= *ham*( f1,f2,fs ),** trong đó f1,f2 là giá trị tần số của tín hiệu đầu vào , fs là tần số lấy mẫu tín hiệu, t và y là các vector thời gian và tín hiệu đầu ra tương ứng, với y được xác định theo công thức sau : **y= cos(2πf1t)+ cos(2πf2t+/4)** trên 2 chu kỳ của tín hiệu f2**.**

|  |
| --- |
| **function [t,y ] = ham(f1,f2,fs) T1=1/f1; T2=1/f2; t=0:1/fs:2\*T2; y=cos(2\*pi\*f1\*t)+cos(2\*pi\*f2\*t+pi/4); end** |

**Bài 15 :** Viết *function* MatlaB để tạo chuỗi xung với cấu trúc như sau: ***function*[t,y]= *xungtamgiac*( R,fs,Np ),** trong đó R là tốc độ xung , fs là tần số lấy mẫu tín hiệu, Np là số lượng xung trong chuỗi, t và y là các vector thời gian và mẫu chuỗi tín hiệu đầu ra tương ứng. Biết trong một chu kỳ xung được mô tả bởi  
**y(t) = 1-|(2t-Tp)/Tp|** với 0 ≤ t ≤ Tp và **Tp = 1/R**. Sử dụng *function* này để vẽ một chuỗi xung gồm **5** (Np=5)xung tại tốc độ **1MHz(R=1e6).** Sau đó tính toán và biểu diễn phổ của chuỗi xung được tạo ra bằng MatlaB

|  |
| --- |
| **function[t,st] =tamgiac(R,fs,Np) Tp=1/R; f=@(t)1-abs(((2\*t-Tp))/Tp); t=[]; st=[]; for k=0:Np-1 t1=[k\*Tp:1/fs:(k+1)\*Tp]; function [t,y ] = ham(f1,f2,fs) T1=1/f1; T2=1/f2; t=0:1/fs:2\*T2; y=cos(2\*pi\*f1\*t)+cos(2\*pi\*f2\*t+pi/4); end** **st1=f(t1-k\*Tp); t=[t,t1]; st=[st,st1]; end** |

**Bài 16 :** Sử dụng Matlab để :

a, Viết chương trình tạo 5 chuỗi xung đầu ra. Biết trong một chu kỳ dạng xung được biểu diễn bởi chuỗi hàm : **x(t)=et** với 0<t<1

b, Viết chương trình vẽ dạng phổ của 5 chuỗi xung trên.

|  |
| --- |
| **function[t,st] = chuoixung(R,Np) Tp=1/R; Tw=Tp/2; fs=16\*R; f=@(t)exp(t); t=[]; st=[]; for i=0:Np-1 t1=[i\*Tp:1/fs:(i+1)\*Tp]; st1=f(t1-i\*Tp); t=[t,t1]; st=[st,st1]; end** *Chương trình chính* **close all; clear all; R=1; Np=5; fs=16\*R; [t,y]=chuoixung(R,Np); f=0:1/(max(t)-min(t)):fs; yf1=fft(y,length(y)); yf2=fftshift(yf1); plot(t,y); figure; plot(f,yf2(1:length(f)));** |

***Dạng câu hỏi 3 điểm : điều chế tín hiệu*Bài 17:** Viết một chương trình MatlaB thực hiện các yêu cầu sau :  
• Tạo chuỗi bản tin ngẫu nhiên phân bố đều phù hợp với điều chế tín hiệu QPSK có độ dài 1000 ký hiệu.  
• Thực hiện điều chế bản tin tạo ra sử dụng kỹ thuật điều chế QPSK  
• Vẽ biểu đồ chòm sao tín hiệu sau khi điều chế.

|  |
| --- |
| **SNR=10; sokyhieu=1000; somuc\_dc=4 data=randint(1,sokyhieu,somuc\_dc); dieuche=qammod(data,somuc\_dc);** **scatterplot(dieuche);** |

**Bài 18:**Viết một chương trình MatlaB thực hiện các yêu cầu sau :  
• Tạo chuỗi bản tin ngẫu nhiên phân bố đều phù hợp với điều chế tín hiệu 4-QAM có độ dài 1000 ký hiệu.  
• Thực hiện điều chế bản tin tạo ra sử dụng kỹ thuật điều chế 4-QAM  
• Vẽ biểu đồ chòm sao tín hiệu sau khi điều chế.

|  |
| --- |
| **SNR=10; sokyhieu=1000; somuc\_dc=4 data=randint(1,sokyhieu,somuc\_dc); dieuche=qammod(data,somuc\_dc); scatterplot(dieuche);** |

**Bài 19:**Viết một chương trình MatlaB thực hiện các yêu cầu sau :  
• Tạo chuỗi bản tin ngẫu nhiên phân bố đều phù hợp với điều chế tín hiệu 4-QAM có độ dài 1000 ký hiệu.  
• Thực hiện điều chế bản tin tạo ra sử dụng kỹ thuật điều chế 4-QAM  
• Thêm nhiễu AWGN vào tín hiệu tạo ra tại SNR =10 dB. Giả sử công suất tín hiệu trung bình được ước tính bởi hàm *Ps = var(y).*• Vẽ biểu đồ chòm sao tín hiệu trước và sau khi thêm nhiễu.

|  |
| --- |
| **SNR=10; sokyhieu=1000; somuc\_dc=4 data=randint(1,sokyhieu,somuc\_dc); dieuche=qammod(data,somuc\_dc);** **Ps=var(dieuche); Pn=Ps/(10^(SNR/10)); themnhieu = dieuche+sqrt(Pn/2)\*(randn(1,sokyhieu) +1i\*randn(1, sokyhieu)); scatterplot(themnhieu)** |

**Bài 20:**Viết một chương trình MatlaB thực hiện các yêu cầu sau :  
• Tạo chuỗi bản tin ngẫu nhiên phân bố đều phù hợp với điều chế tín hiệu QPSK có độ dài 1000 ký hiệu.  
• Thực hiện điều chế bản tin tạo ra sử dụng kỹ thuật điều chế QPSK  
• Thêm nhiễu AWGN vào tín hiệu tạo ra tại SNR=10 dB. Giả sử công suất tín hiệu trung bình được ước tính bởi hàm *Ps = var(y).*• Vẽ biểu đồ chòm sao tín hiệu trước và sau khi thêm nhiễu.

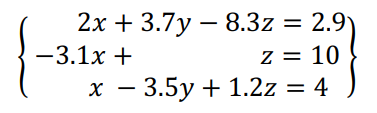
|  |
| --- |
| **SNR=10; sokyhieu=1000; somuc\_dc=4 data=randint(1,sokyhieu,somuc\_dc); dieuche=pskmod(data,somuc\_dc); Ps=var(dieuche); Pn=Ps/(10^(SNR/10)); themnhieu=dieuche+sqrt(Pn/2)\*(randn(1,sok yhieu)** **+1i\*randn(1, sokyhieu)); scatterplot(themnhieu)** |

**Bài 21:**Sử dụng hàm tạo tín hiệu được cho ở câu 14.Viết một chương trình MatlaB thực hiện các yêu cầu sau :

• Tạo tín hiệu bằng việc gọi hàm cho ở câu 14 với ***f1=10Hz***, ***f2=15Hz*** và ***fs =16 f2.***• Thêm nhiễu AWGN vào tín hiệu tạo ra tại SNR =8 dB. Giả sử công suất tín hiệu trung bình được ước tính bởi hàm *Ps = var(y).*• Vẽ tín hiệu được tạo ra và tín hiệu bị nhiễu trên 2 đồ thị riêng biệt trong cùng 1 hình.

|  |
| --- |
| **f1=10; f2=15;fs=16\*f2; SNR=8; [t,y]=ham(f1,f2,fs); Ps=var(dieuche); Pn=Ps/(10^(SNR/10)); tinhieu=y+Pn\*randn(1,length(y)); subplot(2,1,1); plot(t,y); subplot(2,1,2); plot(t,tinhieu);** |

**Dạng bài toán 2 điểm  
Bài 22 :**Sử dụng matlab để giải hệ phương trình tuyến tính sau:



|  |
| --- |
| **A = [2 3.7 -8.3; -3.1 0 1; 1 -3.5 1.2];**  **B = [2.9; 10; 4];**  **Ai = [];**  **result = [];**  **if det(A) == 0**  **disp('PT vo nghiem hoac vo so nghiem');**  **else**  **for i=1:size(A,2)** *% Lay kich thuoc cot cua ma tran*  **Ai = A;**  **Ai(:,i) = B;**  **result(i) = det(Ai)/det(A);**  **end**  **end**  **disp(result');** |

**Bài 24** Hãy giải thích các câu lệnh Matlab sau đây:  
>>x=[2 3 5 8; 3 4 1 5; 3 2 6 7];  
>>n = size(x,1);  
>>ind=mod(1:n,2)==0; >>X(ind,:) =0;

|  |
| --- |
| **Chương trình Matlab này được sử dụng để đưa các phần tử thuộc các hàng chẵn bằng không. >>x=[2 3 5 8; 3 4 1 5; 3 2 6 7]; Tạo ma trận x >>n = size(x,1); n bằng số hàng của x; >>ind=mod(1:n,2)==0; ind là các hàng có chỉ số chẵn trong n hàng của x.** **>>x(ind,:) =0; gắn các hàng chẵn của ma trận X=0; Kết quả thu được x=[2 3 5 8; 0 0 0 0; 3 2 6 7];** |

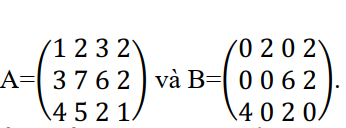
**Bài 25** Hãy giải thích các câu lệnh Matlab sau đây:  
x=[0.2 0.3 0.5 0.8 0.3 0.4 0.1 0.5 0.3 0.2];  
d=x>=0.3;  
x(mod(x,0.3)==0)=1; e=x~=d;

|  |
| --- |
| **Chương trình matlab này thực hiện nhiệm vụ tạo ma trận logic nếu ma trận phần tử của ma trận x lớn hơn 0.3 và chia hết cho 0.3 giá trị của phần tử tương ứng bằng 0 khác thì bằng một); x=[0.2 0.3 0.5 0.8 0.3 0.4 0.1 0.5 0.3 0.2]; tạo ma trận x d=x>=0.3; d là ma trận logic các vị trí của x lớn hơn 0.3 thì các vị trí tương ứng của d bằng 1 x(mod(x,0.3)==0)=1; gắn các phần tử của x chia hết 0.3 bằng 1;** **e=x~=d; e là ma trận logic các vị trí nếu phần tử x khác d thì các vị trí tương ứng của e bằng 1; kết quả: e =[1 0 1 1 0 1 1 1 0 1]** |

**Bài 26** Hãy giải thích các câu lệnh Matlab sau đây:  
x=[-0.2 0.3 0.5 0.8 -0.3 0.4 -0.1 -0.5 0.3 0.2];  
d=x>=0;  
x(mod(x,0.5)==0)=1;  
e=x~=d;

|  |
| --- |
| **Chương trình matlab này thực hiện nhiệm vụ tạo ma trận logic nếu ma trận phần tử của ma trận x lớn hơn 0 và chia hết cho 0.5giá trị của phần tử tương ứng bằng 0 khác thì bằng một); Hãy giải thích các câu lệnh Matlab sau đây: x=[-0.2 0.3 0.5 0.8 -0.3 0.4 -0.1 -0.5 0.3 0.2]; tạo ma trận x d=x>=0;**  **nếu x(mod(x,0.5)==0)=1; e=x~=d;** **kết quả: e =[ 1 0 1 1 0 1 1 1 0 1]** |

**Bài 27** Cho 2 ma trận sau



Hãy viết các lệnh cần thiết trong matlab để tạo ma trận A và chuyển đổi ma trận a thành ma trận B.

|  |
| --- |
| **Chuyển ma trận A thành ma trận B tức là gắn các phần tử lẻ của A là 0; A=[1 2 3 2; 3 7 6 2; 4 5 2 1]; B=A.\*(mod(A,2)==0)** |

**Bài 28**Cho một hàm f(x)=x2-cos(2x). hãy viết các lệnh để vẽ đồ thị hàm f(x) trong khoảng [-2;+2] với tổng số 40 mẫu trên cửa sổ biểu diễn. Từ tập mẫu trên của số biểu diễn. Từ tập mẫu giá trị tính được giảm một nửa số mẫu trên cùng khoảng giá trị và vẽ lại hàm trong một hình vẽ khác trong matlab

|  |
| --- |
| **f=@(x)x.^2-cos(2\*x); x=linspace(-2,2,40); y=f(x); plot(x,y); ind=mod(1:length(y),2)==0; y1=y(ind); x1=x(ind); figure; plot(x1,y1);** |

**MỘT SỐ HÀM ĐỂ LÀM BÀI TÍN HIỆU MÔ PHỎNG**

**Mã hóa PCM:**

|  |
| --- |
| **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 = quatized sequence before encoding*  *% sqnr = signal to quantization noise ra/o in dB*  **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));**  **code = de2bi(xcode,Nb,'left-msb');**  *% Mã hóa không đều theo luật u*  *%% 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* |

**Mã đường xung vuông:**

|  |
| --- |
| **function [t,y] = rectpulse(Tw,Rp,Ns,Np)**  *% Chuong trinh vi du tao chuoi xung vuong*  *% Tw - the pulsewidth*  *% 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*  **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** |

**Mã NRZ:**

|  |
| --- |
| **function [t,y,code] = nrzcode(d,R,Ns,type)**  *% Chuong trinh 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')*  *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** |

**Mã AMI:**

|  |
| --- |
| **function [t,y,code] = amicode(d,R,Ns,type)**  *% Chuong trinh 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')*  *...*  **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**  *..* |

**Mã hóa kênh, mã khối:**

|  |
| --- |
| *% Chuong trinh 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 1 0 1 1 1;**  **1 1 1 0 0 0 1 1 1 0;**  **0 1 1 0 1 1 0 1 0 1;**  **1 1 0 1 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,:))'));** |

**Điều chế biên độ AM:**

|  |
| --- |
| *%% 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);** |

**Điều chế tần số FM:**

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

**Điều chế pha 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);** |

**Điều chế biên độ ASK:**

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

**Điều chế pha PSK:**

|  |
| --- |
| *% 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);** |

**Điều chế pha mã hóa vi sai:**

|  |
| --- |
| *% 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);** |

**Điều chế QAM:**

|  |
| --- |
| **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;** |

**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,[],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);** |

**Bộ lọc raised cosine:**

|  |
| --- |
| **function y = raisedcosflt(x,Rb,Ts,beta)**  *% Function bo loc raised cosine*  *% x - input samples*  *% Rb - filter bandwidth*  *% Ts - sampling time*  *% beta - rolloff factor*  *% y - filtered output*  **Ns = length(x);**  **Tb = 1/Rb;**  **beta = beta\*Rb;**  *% 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;** |

**PP 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** |

|  |  |
| --- | --- |
| **Giải phương trình**  **phươngpháp chia nửa**  function x=bisection(f,a,b,er)  while abs(er)> 1e-4  x0=(a+b)/2;  if f(x0)\*f(a)>0;  a=x0;  else  b= x0;  end  er=(b-a)/b;  end;  disp(' Nghiem phuong trinh la: ');  disp(x0);  end  %%%%%%%%%%%%%%%%%  Trên command Window nhập  bisection(@(x)3\*x^4-7\*x^2+2\*x+2,-1,2,0.0005)  **Giải phương trình vi phân**  **bằng PP euler** y’(t)=(yt)+y(t)+…  ap=3; beta=1.5  f=@(t,y)ap\*y.^beta-y;  h=0.25; a=0; b=2; y(1)=1;  t=a:h:b;  for k=2:length(t)      y(k)=y(k-1)+f(t(k-1),y(k-1))\*h;  end  plot(t,y); | **Tính tích phân bằng**  **phương pháp điểm giữa**  function [I] = midpoint( f,a,b,N)  h=(b-a)/N;  x=a:h:b;  I=0;  i=1;  while i<=N      I=I+h\*f(x(i)+h/2);      i=i+1;  end  end  %%%%%%%%%%%%%  Trên command Window nhập  midpoint(@(x)x\*exp(-x),-2,0,50) |

**Bài tập vẽ xung**

|  |  |
| --- | --- |
| **Xung tam giác**  function[t,y]=xungtamgiac(Tp,fs,Np)  %Tp chu ki xung  %fs tan so xung  %Np: so luong xung co trong chuoi  Tw=Tp/2;  tN=Np\*Tp;  Ts=1/fs;  y=[];  for t=0:Ts:tN      y1=1-abs((mod(t,Tp)-Tw)/Tw);      y=[y y1];  end  t=0:Ts:tN;  plot(t,y)  Ns=length(t);  Ts=1/fs;  f=[-Ns/2:Ns/2-1]/(Ts\*Ns);  yf=fft(y,Ns);  yf=fftshift(yf)/Ns;  figure  plot(f,yf);  end  **Xung e mũ –t**  Np=4;    %so xung can ve  fs=1e2;  t=0:1/fs:Np;  dem=1;  Tp=2;  fs=100;  f=@(t)exp((-t));  t=[];  y=[];  for i=0:Np-1      t1=[i\*Tp:1/fs:(i+1)\*Tp-1/fs];      st1=f(t1-i\*Tp);      t=[t,t1];      y=[y,st1];  end   plot(t,y); | **Xung vuông**  clear all; clf;  N=4; % so xung can ve  fs=1000;  Tp=1; % do rong xung  t=[]; y=[];  for i=0:N-1      t1=[i\*Tp+1/fs:1/fs:(i+1/2)\*Tp]      y1(1:length(t1))=1;      t2=[(i+1/2)\*Tp+1/fs:1/fs:(i+1)\*Tp];      y2(1:length(t2))=0;      t=[t,t1,t2];      y=[y,y1,y2];  end  plot(t,y);  **%phan b ve dang pho**   f=0:1/(max(t)-min(t)):fs;   sf1=fft(y,length(y));   sf2=fftshift(sf1);   figure;   plot(f,sf2);  **Bài có nhiều hàm**  function [t,y] = temp(T,Ns)      N=1;      f= @(t) 1-t/T;      t=[],y=[];      for k=0:N-1      t1=k\*3\*T:1/Ns:k\*3\*T+T-1/Ns;      y1(1:length(t1))=-1;      t2= k\*3\*T+T:1/Ns:k\*3\*T+2\*T-1/Ns;      y2=f(t2-k\*3\*T);      t3= k\*3\*T+2\*T:1/Ns: k\*3\*T+3\*T-1/Ns;      y3(1:length(t3))= 1;      t=[t,t1,t2,t3];      y=[y,y1,y2,y3];      end      plot(t,y);  end |

|  |  |
| --- | --- |
| **Đề cho SNR**  clear all  close all  *M = 16;*  *SNRdB = 10;* %dB  x = randint(200,1,M);  h = modem.qammod(M);  h.symbolorder = 'gray';  y = modulate(h,x);  Ps = 10\*log10(var(y)); % in dBW  y\_noisy = awgn(y,SNRdB,Ps);  h=modem.qamdemod(M);  h.symbolorder = 'gray';  xt = demodulate(h,y\_noisy);  p = scatterplot(y\_noisy,1,0,'rx');  hold on  scatterplot(y,1,0,'b\*',p)  eyediagram(y,2)  eyediagram(y\_noisy,2)  BER=sum(xor(x,xt));  disp('so bit loi: ');  disp(BER); | **Đề cho N0**  M = 8;  No = 9; %dB  x = randint(500,1,M);  h = modem.pskmod(M);  h.symbolorder = 'gray';  y = modulate(h,x);  Ps = 10\*log10(var(y)); % in dBW  SNRdB= Ps-No;  y\_noisy = awgn(y,SNRdB,Ps);  h=modem.pskdemod(M);  h.symbolorder = 'gray';  xt = demodulate(h,y\_noisy);  p = scatterplot(y\_noisy,1,0,'rx');  hold on  scatterplot(y,1,0,'b\*',p)  eyediagram(y,2)  eyediagram(y\_noisy,2)  BER=sum(xor(x,xt));  disp('so bit loi: ');  disp(BER); |

*m=16; data=randint(1,1500,m);  dc=pskmod(data,m); No=10;*

Es=5e-5; phi=0; Rs=1e6; fs=20\*Rs;c=1; Ts=1/Rs;

 t=0:1/fs:length(dc)\*Ts;

for i=1:length(t)

    Tx(i)=dc(c)\*sqrt(2\*Es/Ts)\*exp(1j\*phi);

    if(t(i)>=c/Rs)

        c=c+1;

    end

end

nhieu=sqrt(No/2)\*(randn(1,length(Tx))+1i\*randn(1,length(Tx)));

Rx=Tx+nhieu; d=1;

for i=1:length(t)

    if abs(t(i)-(d-1/2)/Rs)<=1/(2\*fs)

    giaidc(d)=Rx(i)/(sqrt(2\*Es/Ts)\*exp(1j\*phi));

    d=d+1;

    end

end

giaima=pskdemod(giaidc,m);

%muc bien do tin hieu truoc khi qua kenh truyen

plot(t(1:1000),Tx(1:1000));

%muc bien do tin hieu sau khi qua kenh truyen

figure;

plot(t(1:1000),Rx(1:1000));

%bieu do chom sao tin hieu truoc khi qua kenh truyen

scatterplot(dc);

%bieu do chom sao tin hieu sau khi qua kenh truyen

scatterplot(giaidc);

%bieu do mat tin hieu truoc khi qua kenh truyen

eyediagram(Tx,100);

eyediagram(Rx,100); %bieu do mat tin hieu sau khi qua kenh truyen

**Tạo chuỗi kí tự ngẫu nhiên có độ dài 1000 kí tự để thực hiện điều biến 16-QAM. Sử dụng mô hình mô phỏng tương đương băng gốc, tín hiệu phát có thể được biểu diễn như sau:**

**st=k=-∞dkp(t-kT)ej∅0**

**Trong đó dk là các kí hiệu phức được phát độc lập với xác suất như nhau; Tsym là chu kỳ của symbol; ∅0 là pha của tín hiệu phát và p(t) xác định dạng xung được phat:**

**pt=2EsTsym1-cos2πtTsym**

**cho tín hiệu16-QAM. Tín hiệu sau điều biến được đưa qua kênh AWGN có mật độ phổ công suất nhiễu N0=9. Tại nơi thu, tín hiệu được giải điều chế và khôi phục chuỗi kí tự ban đầu.**

**a. Vẽ mức biên độ của tín hiệu 16-QAM tại vị trí trước và sau khi qua kênh truyền**

**b. vẽ giản đồ chòm sao của tín hiệu 16-QAM tại vị trí trước và sau khi qua kênh truyền**

**c. Vẽ giản đồ mắt của tín hiệu 16-QAM tại vị trí trước và sau khi qua kênh truyền.**

**Lời giải:**

Chương trình chính:

clear all; clc;close all;

Rs=1e6;fs=20e6;phi=pi/2; Es=2;SNR=10; m=16;

v=randint(1,1000,m);

qam=qammod(v,m);

st=awgn(qam,SNR,'measured');

giaima=qamdemod(st,m);

 [t,st1]=taoxung(qam,Es,phi,Rs,fs);

plot(t(1:1000),st1(1:1000));

figure;

st2=awgn(st1,SNR,'measured');

plot(t(1:1000),st2(1:1000));

figure;

sf1=fft(st1,length(st1));

f=0:1/(max(t)-min(t)):fs;

sf2=fftshift(sf1);

plot(f,sf2)

figure;

sf3=fft(st2,length(st2));

sf4=fftshift(sf3);

plot(f,sf4);

h=scatterplot(st,1,0);

hold on

scatterplot(qam,1,0,'ro',h);

eyediagram(st1,100);

eyediagram(st2,100);

Hàm tạo xung:

function [ t,y ] = taoxung( s,Es,phi,Rs,Ns )

c=1; t=0:1/Ns:length(s)/Rs;

for i=1:length(t)

    y(i)=s(c)\*sqrt(2\*Es\*Rs)\*(1-cos(2\*pi\*t(i)\*Rs))\*exp(1j\*phi);

    if(t(i)>=c/Rs)

        c=c+1;

 end

 end

ends

|  |  |
| --- | --- |
| % xung RZ qua bo loc Butterworth%%%%  clear all;  d=randint(1,20);  Tb=10;  Nb=length(d);  Ns=1000;  Timewindow = Nb\*Tb;  Ts = Timewindow/(Ns-1);  t=[0:Ts:Timewindow];  x=zeros(size(t));  for k=1:Ns  n=fix(t(k)/Tb)+1;  if n> Nb  n=Nb;  end  if mod(t(k),Tb) <= Tb/2;  x(k)=2\*d(n)-1;  else  x(k) = 0;  end  end  plot(t,x);  title('tin hieu truoc bo loc')  f=[0:length(t)/2-1 -length(t)/2:-1]/(Ts\*length(t));  Xf=fft(x);  Hf = 1./(1+(f./3).^(2\*n));  Yf=Xf.\*Hf;  y=ifft(Yf);  figure  plot(t,y);  title('tin hieu tai dau ra bo loc'); | % xung NRZ qua bo loc Gauss%%%%  clear all;  d=randsrc (1,10,[0 1]);  Tb=10;  Nb=length(d);  Ns=1000;  Timewindow = Nb\*Tb;  Ts = Timewindow/(Ns-1);  t=[0:Ts:Timewindow];  x=zeros(size(t));  for k=1:Ns  n=fix(t(k)/Tb)+1;  if n> Nb  n=Nb;  end  x(k)=d(n);  end  plot(t,x);  title('tin hieu truoc bo loc')  f=[0:length(t)/2-1 -length(t)/2:-1]/(Ts\*length(t));  Xf=fft(x);  Hf =exp(-f.^2/2);  Yf=Xf.\*Hf;  y=ifft(Yf);  figure  plot(t,y);  title('tin hieu tai dau ra bo loc'); |