

گزارش 6 آزمایشگاه مخابرات دیجیتال

دانشگاه خلیج فارس

نام: محمد صالح رایانی شماره دانشجویی: 990291614

هدف: شبیه‌سازی رابطه‌ی نرخ خطای سمبل‌ و *SNR*

نرخ خطای سمبل‌ (*SER*) معیاری است که برای ارزیابی عملکرد سیستم های ارتباطی دیجیتال استفاده می شود. این کمیت، احتمال تشخیص یا رمزگشایی نادرست سمبل‌‌های ارسالی در حضور نویز، تداخل و اختلالات کانال را نشان می‌دهد. در ارتباطات دیجیتال، اطلاعات معمولاً به عنوان سمبل‌های مجزا از یک الفبای محدود، مانند سمبل‌های باینری (0ها و 1ها) یا سیستم‌های مدولاسیون مرتبه بالاتر مانند مدولاسیون دامنه تربیعی (QAM) منتقل می‌شوند. سمبل‌های دریافتی ممکن است به دلیل عوامل مختلفی مانند نویز وارد شده در حین انتقال، فیدینگ یا محو شدن، تداخل سیگنال های دیگر و اعوجاج در کانال ارتباطی خراب شوند.

هدف یک گیرنده در یک سیستم ارتباطی دیجیتال شناسایی و رمزگشایی صحیح سمبل‌های ارسالی است که امکان بازیابی اطلاعات اصلی را فراهم می‌کند. اما به دلیل وجود نویز و سایر اختلالات، ممکن است در سمبل‌های دریافتی خطا رخ دهد. *SER* احتمال این خطاهای سمبل‌ را اندازه‌گیری می کند.

به طور ریاضی، *SER* به عنوان نسبت تعداد خطاهای سمبل‌ به تعداد کل سمبل‌های ارسالی تعریف می‌شود و معمولاً به صورت کسری یا درصدی بیان می‌شود. فرمول محاسبه *SER* به صورت زیر است:  
*SER* = تعداد خطاهای سمبل‌ / تعداد کل سمبل‌های ارسالی

برای تعیین *SER*، باید سمبل‌های دریافتی را با سمبل‌های مورد انتظار در گیرنده مقایسه کرد. سمبل‌های مورد انتظار شناخته شده اند زیرا فرستنده و گیرنده یک روش مدولاسیون از پیش تعریف شده و الگوریتم های رمزگذاری/رمزگشایی مشترک دارند. گیرنده سمبل‌های دریافتی را تشخیص داده و آنها را با سمبل‌های مورد انتظار مقایسه می کند تا خطاها را شناسایی کند. برای محاسبه دقیق *SER*، داشتن دانش از سمبل‌های ارسالی در گیرنده ضروری است.

*SER* تحت بستگی به عوامل متعددی دارد از جمله نسبت سیگنال به نویز (*SNR*)، سیستم مدولاسیون، شرایط کانال، طراحی گیرنده و وجود تداخل. به طور کلی، با افزایش *SNR*، *SER* کاهش می‌یابد که نشان‌دهنده عملکرد بهتر است.

کد مطلب:

clc;clear;close all;

snr = -4:0.5:14;

psk\_ser(snr, 6)

در خط اول کدهای خالی کردن پنجره فرمان و متغیر ها و بستن فیگر ها را می‌نویسیم، سپس *snr* را در واحد dB وارد می‌کنیم. تابع ما به نام psk\_ser مقادیر *snr* و میزان بیت در هر سمبل را به عنوان ورودی دریافت می‌کند و نمودار *SER* بر اساس *snr* به ما می‌دهد.

function psk\_ser(snr\_db, n)

for i=1:n

M = 2.^(i);

ایجاد تابع psk\_ser. ایجاد حلقه for برای مقادیر مختلف بیت در سمبل. *M* تعداد سمبل است که 2 به توان تعداد بیت است.

if M==2 %BPSK%

snr\_lin = 10.^(snr\_db/10);

gamma\_b = snr\_lin;

SER = erfc(sqrt(gamma\_b))./2;

semilogy(snr\_db, SER, 'DisplayName','B-PSK', LineWidth=1.5, LineStyle='-')

hold on

اگر *M* 2 بود مدولاسیون B-PSK است و این کد اجرا می‌شود. اول *snr* در واحد dB را به واحد خطی تبدیل می‌کنیم. متغیر انرژی بیت *gamma\_b* را تشکیل می‌دهیم که همان *snr* خواهد بود. فرمول *SER* طبق روابط از طریق تابع Q بدست می‌آید. برای محاسبه تابع Q از تابع erfc متلب استفاده می‌کنیم که به معنای complementary error function یا تابع خطای متمم است. تابع خطای متمم انتگرال از منفی بی‌نهایت تا عدد یک تابع گوسین با آرگومان درجه دو را محاسبه می‌کند. دستور semilogy محور عمودی نمودار را در مقیاس‌های 10 به توان اعداد صحیح رسم می‌کند. ویژگی DisplayName نام این تابع را مشخص می‌کند و LineWidth و LineStyle ضخامت و فرم نمایش خط را مشخص می‌کنند. دستور hold on باعث می‌شود که نمودارهای بعدی روی این فیگر نمایش داده شوند و نمودار قبلی پاک نشود.

elseif M==4 %QPSK%

snr\_lin = 10.^(snr\_db/10);

gamma\_b = snr\_lin;

SER = erfc(sqrt(gamma\_b)) - (erfc(gamma\_b)./2).^2;

semilogy(snr\_db, SER, 'DisplayName','Q-PSK', LineWidth=1.5, LineStyle='--')

hold on

این کد برای حالت Q-PSK اجرا می‌شود و تفاوت زیادی با کد قبل ندارد. فرمول *SER* و نام و فرم نمایش نمودار عوض شده است.

elseif M>4 %MPSK%

snr\_lin = 10.^(snr\_db/10);

gamma\_s = log2(M).\*snr\_lin;

SER = erfc(sin(pi./M).\*sqrt(gamma\_s));

semilogy(snr\_db, SER, ':o', 'MarkerSize', 5, 'DisplayName',num2str(M)+"-PSK", LineWidth=1.5)

hold on

end

برای حالت بیت‌های بیشتر مدولاسیون M-PSK این کد اجرا خواهد شد. به جای *gamma\_b* متغیر *gamma\_s* را داریم که ضرب لگاریتم دوی *M* در *snr* خطی است. فرمول *SER* عوض شده است و در تابع semilogy، ‘:o’ یعنی نمودار با نقطه چین و نشان های دایره‌ای نمایش داده شود و MarkerSize سایز دایره را مشخص می‌کند. نام num2str(M)+”PSK” باعث می‌شود که هر بار *M* تغییر کرد نام نمودار هم تغییر کند



نتیجه‌گیری: با افزایش تعداد بیت و سمبل، احتمال خطا در سیستم مخابراتی افزایش می‌یابد و نیاز به انرژی بیشتری برای کاهش نرخ خطا خواهیم داشت