Иллюстрация эффективности методов FEC. Часть 3 - Разработка программы. Анализ результата

Сергей Милованов, 28 мая 2012г.

ИЛЛЮСТРАЦИЯ ЭФФЕКТИВНОСТИ МЕТОДОВ

ПОМЕХОУСТОЙЧИВОГО КОДИРОВАНИЯ

(С.А. Милованов, РГРТУ, гр.8110)

Часть 3. Разработка программы. Анализ результата

1.Почему был выбран язык программирования MatLAB?

Язык программирования системы MatLAB весьма прост, близок к языку BASIC, посилен любому начинающему; он содержит всего несколько десятков операторов; незначительное количество операторов здесь компенсируется большим числом процедур и функций, содержание которых легко понятно пользователю с соответствующей математической и инженерной подготовкой [1].

Запись программ в системе является традиционной и потому обычной для большинства пользователей персональных компьютеров. И вдобавок система дает возможность редактировать программы при помощи любого привычного для пользователя текстового редактора [1].

С системой MatLAB поставляются свыше ста m-файлов, которые содержат демонстрационные примеры и определения новых операторов и функций. Эта библиотека, все файлы которой подробно прокомментированы, - подлинная сокровищница прекрасных примеров программирования на языке системы. Изучение этих примеров и возможность работы в режиме непосредственных вычислений значительно облегчают знакомство с системой серьезных пользователей, заинтересованных в использовании математических расчетов [1].

2. Как выглядит исходный код программы?

```
clc, clear;
 n = 7;% Длина кодового слова
k = 4;% Длина сообщения
nbit=10^7;        % Кол-во пропускаемых бит
d=round(nbit/n);   % Счетчик цикла подсчета Рош
count=1;          % Нач. знач. счетчика повторений
for snr=0:20% Цикл изменения ОСШ
   s_num_fec_cyc=0;   % нач. знач. суммы Рош
s_num_fec=0;
s_num=0;
for z=1:d     % Цикл подсчета Рош при фиксир. ОСШ
msg = randi([0,1],1,k);% исходное сбщ
%% Кодирование и наложение шума
code_cyc = encode(msg,n,k,'cyclic/binary');% цикл.кодир.
code = encode(msg,n,k,'hamming/binary');% кодир. Хэмм.
noisycode_cyc = awgn(code_cyc,snr); % доб. шумав цикл.
noisycode = awgn(code,snr);% доб. шумав Хэмм.
noisy_msg = awgn(msg,snr);% доб. шумав исх. сбщ
%% Транспонируем шум в битовую ошибку
for i=1:n
if noisycode_cyc(i)<0.5   % циклический код
noisycode_cyc(i)=0;
else
noisycode_cyc(i)=1;
end
if noisycode(i)<0.5      % код Хэмм.
noisycode(i)=0;
else
noisycode(i)=1;
end
end
for i=1:k% исх. сбщ
if noisy_msg(i)<0.5
noisy_msg(i)=0;
else
noisy_msg(i)=1;
```

```
end
end
%% Декодированиеи подсчет ошибок
dec_cyc = decode(noisycode_cyc,n,k,'cyclic/binary');% декодированиецикл.
decodedhamm = decode(noisycode,n,k,'hamming/binary');% декодированиеХэмм.
[num_fec_cyc,ratio_fec_cyc] = biterr(dec_cyc,msg);% подсчет ошибок в декодир. цикл.
[num_fec,ratio_fec] = biterr(decodedhamm,msg);% подсчет ошибок в декодир. Хэмм.
[num,ratio] = biterr(noisy_msg,msg);% подсчет ошибок в исх. сбщ
s_num_fec_cyc=s_num_fec_cyc+num_fec_cyc;% накопл.ош. цикл.
s_num_fec=s_num_fec+num_fec;% накопл.ош. Хэмм.
s_num=s_num+num;              % накапл.ош. исх.
end
rat_fec_cyc=s_num_fec_cyc/nbit;
% подсчет Рош в системе с цикл.
rat_fec=s_num_fec/nbit;
% подсчет Рош в системе Хэмм.
rat=s_num/nbit;
% подсчет Рош в системе без кодир.
count=count+1; % инкремент счетчика номера массива
RAT_FEC_CYC(count)=rat_fec_cyc;% запись данных в массивы
RAT_FEC(count)=rat_fec;
RAT(count)=rat;
SNR(count)=snr;
end
%% Построениеграфиков
semilogy(SNR,RAT,SNR,RAT_FEC_CYC,SNR,RAT_FEC),grid,xlabel('SNR, dB'),ylabel('Рош'),title('Зависимость Рош от ОСШ');
```

Что получилось в итоге?

На рис.1 представлен график зависимости вероятности битовой ошибки от отношения сигнал/шум для трёх ситуаций:

- канальное кодирование в системе не применялось;
- был использован циклический код (7:4);
- был использован код Хэмминга (7:4).

Следует отметить, что построенные графики были получены по исходным данным, несколько отличным от указанных в программном коде, а именно - «nbit=10^4», т.е. количество бит, пропущенных через систему, примерно равно 1e4. Это вызвано довольно невысокой производительностью компьютера, который, по моим грубым подсчётам, с параметром «nbit=10^7» строил бы необходимые графики около недели. Для обеспечения теоретической вероятности ошибки равной 1e-6 пропустить через систему 1e4 бит не достаточно, о чем уже говорилось в части 2, но в целом проанализировать графики и решить поставленную задачу данное допущение позволяет (будем считать, что графики продолжают прежнюю тенденцию и для Рош<1e-4).

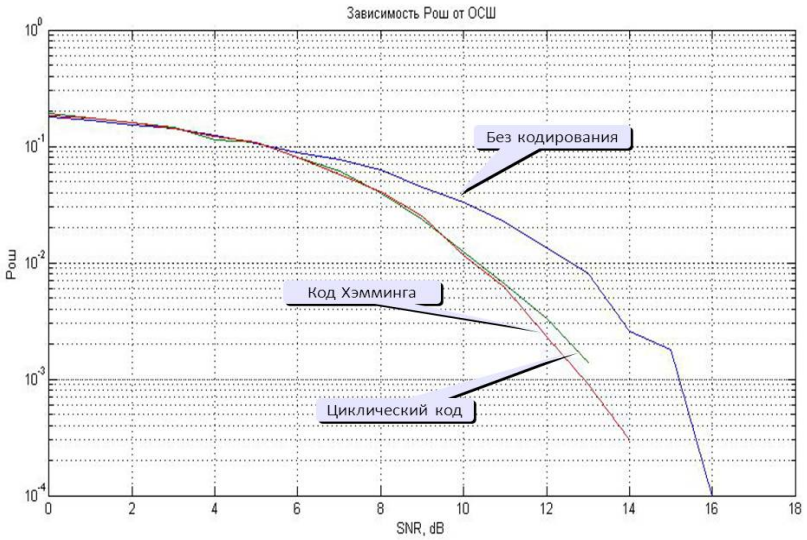

рис.1 Зависимость Рош от ОСШ

Полученные графики наглядно демонстрируют энергетический выигрыш при использовании помехоустойчивого кодирования (FEC). Например, для обеспечения вероятности битовой ошибки 1e-2 системе без использования FEC необходимо ОСШ ≈ 13 дБ, а системе с использованием FEC - ОСШ ≈ 10 дБ.

Необходимо обратить внимание, что указанная в данном примере вероятность битовой ошибки 1e-2 на самом деле очень велика, и реальные системы связи перестанут работать в таких условиях. Данное допущение было принято в связи с невозможностью компьютера смоделировать работу системы дляверотности битовой ошибки 1e-6 в сложившихся временных рамках.

Список использованной литературы

1. Лазарев Ю.Ф. - Начала программирования в среде MatLAB: Учебное пособие. - К.:НТУУ "КПИ", 2003. - 424 с.

2. www.mathworks.com