ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ им. проф. М. А. БОНЧ-БРУЕВИЧА»
(СПбГУТ)

С. С. Владимиров

ПРАКТИКА ПОМЕХОУСТОЙЧИВОГО КОДИРОВАНИЯ

Циклические коды

Лабораторный практикум

СПб ГУТ)))

Санкт-Петербург
2020

УДК ХХХ.ХХХ (ХХХ)
ББК ХХ.ХХХ.Х хХХ
В 57

Рецензент
— —

Рекомендован к печати редакционно-издательским советом СПбГУТ

Владимиров, С. С.
В 57 Практика помехоустойчивого кодирования. Циклические коды : лабораторный практикум / С. С. Владимиров, ; СПбГУТ. — СПб, 2020. — 18 с.

Призван ознакомить студентов с циклическими помехоустойчивыми кодами и методами их декодирования. Представленный материал служит справочным и методическим пособием при выполнении курса лабораторных работ по дисциплинам «Практика помехоустойчивого кодирования в современных инфокоммуникационных системах» и «Помехоустойчивое кодирование в инфокоммуникационных системах».

Предназначен для студентов, обучающихся по направлениям 09.03.01 «Информатика и вычислительная техника» и 11.03.02 «Инфокоммуникационные технологии и системы связи».

УДК ХХХ.ХХХ (ХХХ)
ББК ХХ.ХХХ.Х хХХ

Содержание

Лабораторная работа 1
Циклические коды

1.1. Цель работы

Рассмотреть на примере и получить навыки в исследовании циклических кодов с использованием системы компьютерной алгебры GNU/Octave.

1.2. Рекомендуемая литература

1. Практика помехоустойчивого кодирования. Циклические коды : практикум / С. С. Владимиров, О. С. Когновицкий ; СПбГУТ. — СПб, 2019. — 44 с.

2. Е. Р. Алексеев, О. В. Чеснокова «Введение в Octave для инженеров и математиков» М. : ALT Linux, 2012. — 368 с.

3. Documentation // Octave-Forge.
URL: http://octave.sourceforge.net/docs.html

4. Е. Р. Алексеев, О. В. Чеснокова «Введение в Octave» // НОУ ИНТУИТ. URL: http://www.intuit.ru/studies/courses/3677/919/info

1.3. Порядок выполнения задания

Задание выполняется каждым учащимся индивидуально. Результаты заданий лабораторного практикума и соответствующих заданий практикума должны совпадать.

Отчёт формируется в электронном виде в формате PDF и отправляется на электронную почту преподавателя.

1.3.1.

Для заданного циклического (n,k) кода получить вначале образующий полином, а затем порождающую и проверочную матрицы. Параметры n и k заданы в табл. 1.1 по последней цифре номера зачетной книжки (студенческого).

Таблица 1.1

Параметры циклического кода.
(По последней цифре номера зачетной книжки)

Цифра	**(n,k)−код**
1,6	(12,7)
2,7	(15,7)
3,8	(15,9)
4,9	(15,11)
5,0	(16,11)

Для получения образующего полинома необходимо использовать функцию `cyclpoly`, которая получает на вход параметры кода n и k и выводит на экран образующий полином в двоичном виде.

```
> p=cyclpoly(n,k)
```

Более подробно работа функции описана во внутренней справке Octave — команда `help cyclpoly`.

Чтобы получить порождающую и проверочную матрицы необходимо воспользоваться функцией `cyclgen`. На вход она получает длину кода n и образующий полином p. На выходе возвращает проверочную H и порождающую G матрицы (при указанной в примере форме записи).

```
> [H,G]=cyclgen(n,p)
```

Более подробно работа функции описана во внутренней справке Octave — команда `help cyclgen`.

1.3.2.

Закодировать заданный информационный вектор вначале встроенной функцией Octave, затем при помощи умножения на порождающую матрицу. Сравнить результаты. Информационный вектор берется из табл. 1.2 по последней цифре зачетной книжки.

Таблица 1.2

Информационный вектор. По последней цифре номера зачетной книжки

Цифра	Вектор	Цифра	Вектор
1	1 0 0 1 1 1 0	6	1 1 1 0 1 1 0
2	0 1 1 1 0 1 0	7	0 1 0 0 0 1 1
3	1 0 0 1 0 0 1 1 0	8	1 0 1 1 0 0 0 1 1
4	1 0 0 1 0 0 0 1 1 0 1	9	0 1 0 1 1 0 0 1 0 0 1
5	1 0 0 0 1 1 1 0 1 0 1	0	0 1 0 1 0 0 1 1 0 1 1

Для кодирования используется функция `encode`. В качестве параметров задаются исходное сообщение в двоичном виде, параметры кода n и k и указание использовать циклический код.

```
> Menc=encode(Msg,n,k,"cyclic")'
```

Оператор транспонирования «'» указывается, чтобы выводить сообщение строкой, а не столбцом.

Для умножения на порождающую матрицу предварительно необходимо задать информационный вектор и саму матрицу как структуры над простым полем Галуа GF(2).

```
> G2=gf(G,1,3);
> Msg2=gf(Msg,1,3);
```

Далее можно умножать обычным способом.

1.3.3.

Последовательно наложить заданные векторы ошибки на кодовый вектор и декодировать полученные векторы с ошибкой при помощи встроенной функции Octave. Векторы ошибки берутся из табл. 1.3 по последней цифре зачетной книжки. Заданы векторы с одной, двумя и тремя ошибками.

Таблица 1.3

Вектор ошибки. По последней цифре номера зачетной книжки

Цифра	Вектор	Цифра	Вектор
1	0 1 0 0 0 0 0 0 0 0 0 0	6	0 0 0 0 0 0 1 0 0 0 0 0
	0 1 0 0 1 0 0 0 0 0 0 0		1 0 0 0 0 0 1 0 0 0 0 0
	0 1 0 0 1 0 0 1 0 0 0 0		1 0 0 0 0 0 1 0 0 0 1 0
2	0 0 1 0 0 0 0 0 0 0 0 0 0 0 0	7	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
	0 0 1 0 1 0 0 0 0 0 0 0 0 0 0		0 1 0 0 0 0 0 1 0 0 0 0 0 0 0
	0 0 1 0 1 0 0 1 0 0 0 0 0 0 0		0 1 0 0 0 0 0 1 0 0 1 0 0 0 0
3	0 0 0 1 0 0 0 0 0 0 0 0 0 0 0	8	0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
	0 0 0 1 0 1 0 0 0 0 0 0 0 0 0		0 0 1 0 0 0 0 0 1 0 0 0 0 0 0
	0 0 0 1 0 1 0 1 0 0 0 0 0 0 0		0 0 1 0 0 0 0 0 1 0 1 0 0 0 0
4	0 0 0 0 1 0 0 0 0 0 0 0 0 0 0	9	0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
	0 0 0 0 1 0 1 0 0 0 0 0 0 0 0		1 0 0 0 0 0 0 0 0 1 0 0 0 0 0
	0 0 0 0 1 0 1 0 1 0 0 0 0 0 0		1 0 0 0 0 1 0 0 0 1 0 0 0 0 0
5	0 0 0 0 0 1 0 0 0 0 0 0 0 0 0	0	0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
	0 0 0 0 0 1 0 1 0 0 0 0 0 0 0		0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0
	0 0 0 0 0 1 0 1 0 1 0 0 0 0 0		0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0

Для наложения ошибки используется функция `xor`.

```
> Merr1=xor(Menc,Err1)
```

Для декодирования используется функция `decode`. В качестве параметров задаются исходное сообщение в двоичном виде, параметры кода n и k и указание использовать циклический код. На выходе функция возвращает декодированное сообщение `Mdec` и вектор ошибок `err`.

```
> Mdec=decode(Merr1,n,k,"cyclic")'
```

Оператор транспонирования «'» указывается, чтобы выводить сообщение строкой, а не столбцом.

1.3.4.

Сравнить по методу Монте-Карло вероятностные характеристики двух циклических кодов с разной избыточностью. Параметры кодов приведены в

табл. 1.4. Для сравнения воспользоваться написанной в листинге 1.1 программой, изменив ее для своих нужд, подставив необходимые параметры кодов. В результате выполнения будет получен график, который необходимо проанализировать. График и выводы должны быть представлены в отчете. Также необходимо проанализировать текст самой программы и разобраться в ее работе.

Листинг 1.1

Листинг программы для сравнения двух циклических кодов по вероятности битовой ошибки в канале ДСК

```
n1=12;
n2=15;
k=7;
%
s1=sprintf("Cyclic code (%d,%d)",n1,k);
s2=sprintf("Cyclic code (%d,%d)",n2,k);
s3=sprintf("Without coding");
%
p0=[1e-3 5e-3 1e-2 5e-2 1e-1];
stat=zeros(3,5);
%
msg=randi([0 1],1e5,k);
%
menc1=encode(msg,n1,k,"cyclic");
menc2=encode(msg,n2,k,"cyclic");
%
for i=1:1:5
  mrec1=bsc(menc1,p0(i));
  mdec1=decode(mrec1,n1,k,"cyclic");
  [num,rate]=biterr(msg,mdec1);
  stat(1,i)=rate;
  mrec2=bsc(menc2,p0(i));
  mdec2=decode(mrec2,n2,k,"cyclic");
  [num,rate]=biterr(msg,mdec2);
  stat(2,i)=rate;
  mrec3=bsc(msg,p0(i));
  [num,rate]=biterr(msg,mrec3);
  stat(3,i)=rate;
end
%
format long;
%
stat
%
mfig=figure;
L3=loglog(p0,stat(3,:));
set(L3,"LineWidth",3,"Color","r");
hold on;
L1=loglog(p0,stat(1,:));
```

```
40 set(L1,"LineWidth",1,"Color","k");
41 hold on;
42 L2=loglog(p0,stat(2,:));
43 set(L2,"LineWidth",3,"Color","b");
44 hold on;
45 title(sprintf("Cyclic codes (%d,%d) and (%d,%d) in BSC
      channel",n1,k,n2,k));
46 xlabel("BER in BSC channel, p0");
47 ylabel("Error rate after decoding");
48 legend(s3,s1,s2,0);
49 legend("show");
50 grid on;
51 print(mfig,'-dpng',sprintf("cycl-%d-%d_cycl-%d-%d_bsc_err-
      rate",n1,k,n2,k));
```

Программу необходимо сохранить в виде файла с расширением `*.m` и запускать из командной строки (из каталога, в котором лежит программа) так, как указанно ниже.

```
user@name:[~]$ octave -q cycl_compar.m
```

Таблица 1.4

Параметры циклических кодов для сравнения.
(По последней цифре номера зачетной книжки)

Цифра	**Код 1** (n,k)	**Код 2** (n,k)	**Цифра**	**Код 1** (n,k)	**Код 2** (n,k)
1	(12,7)	(15,7)	6	(12,7)	(18,7)
2	(15,7)	(18,7)	7	(15,7)	(22,7)
3	(15,9)	(17,9)	8	(15,9)	(21,9)
4	(15,11)	(21,11)	9	(15,11)	(23,11)
5	(16,11)	(21,11)	0	(16,11)	(23,11)

1.4. Порядок защиты лабораторной работы

Защита работы может осуществляться одним из нижеперечисленных способов или их сочетанием на усмотрение преподавателя.

1. Устный ответ по теме работы.
2. Тестирование по теме работы
3. Задача по теме работы.
4. Иные варианты на усмотрение преподавателя.

Лабораторная работа 2
Коды Боуза–Чоудхури–Хоквингема

2.1. Цель работы

Рассмотреть на примере и получить навыки в исследовании циклических кодов Боуза–Чоудхури–Хоквингема (БЧХ) с использованием системы компьютерной алгебры GNU/Octave.

2.2. Рекомендуемая литература

1. Практика помехоустойчивого кодирования. Циклические коды : практикум / С. С. Владимиров, О. С. Когновицкий ; СПбГУТ. — СПб, 2019. — 44 с.

2. Е. Р. Алексеев, О. В. Чеснокова «Введение в Octave для инженеров и математиков» М. : ALT Linux, 2012. — 368 с.

3. Documentation // Octave-Forge.
URL: http://octave.sourceforge.net/docs.html

4. Е. Р. Алексеев, О. В. Чеснокова «Введение в Octave» // НОУ ИНТУИТ. URL: http://www.intuit.ru/studies/courses/3677/919/info

2.3. Порядок выполнения задания

Задание выполняется каждым учащимся индивидуально. Результаты заданий лабораторного практикума и соответствующих заданий практикума должны совпадать.

Отчёт формируется в электронном виде в формате PDF и отправляется на электронную почту преподавателя.

2.3.1.

Для (n,k)-кода БЧХ $(15,5)$ определить образующий полином $g_{(15,5)}(x)$, минимальные множители, составляющие образующий полином, и разложение соответствующего поля Галуа на классы вычетов (корни минимальных полиномов). Для этого воспользоваться функцией

```
> [g,minpol,c] = bchpoly(n,k)
```

На вход функции задаются параметры кода n и k. На выходе выводятся образующий полином g (в виде двоичного вектора), матрица минимальных полиномов *minpol*, разложение поля на корни минимальных многочленов c. Более подробно работа функции описана во внутренней справке Octave — команда `help bchpoly`.

Результат выполнения задания необходимо сверить с практической работой.

2.3.2.

Сформировать по номеру студенческого (зачетной книжки) информационный полином $u(x)$ для заданного в предыдущем пункте кода. Для этого взять последние две цифры номера, прибавить к ним 37, перевести в двоичный вид и отделить младшие k разрядов. (Выполнялось в практической работе по кодам БЧХ.)

Пример.
Пусть номер оканчивается на 93 и $k = 6$. Тогда $u(x)$ будет равно.

1. $98 + 37 = 135$.
2. $135_{DEC} = 11100001_{BIN}$.
3. $u(x) = [1\ 1\ 1\ 0\ 0\ 0] = 1 + x + x^2$.

Перевод из десятичной системы в двоичную осуществлялся при помощи системы Octave, поэтому младшие разряды располагаются слева.

2.3.3.

Закодировать полученный в предыдущем пункте информационный вектор при помощи встроенных функций Octave. Результат выполнения задания необходимо сверить с практической работой.

Для кодирования используются функции `bchenco` (в качестве параметров задаются исходное сообщение в двоичном виде и параметры кода n и k) и `encode` (в качестве параметров задаются исходное сообщение в двоичном виде, параметры кода n и k и указание использовать код БЧХ).

```
> Menc=bchenco(Msg,n,k)
> Menc=encode(Msg,n,k,"bch")'
```

Оператор транспонирования «'» указывается, чтобы выводить сообщение строкой, а не столбцом.

2.3.4.

Наложить на полученную в предыдущем пункте кодовую комбинацию $v(x)$ циклического систематического кода полином ошибки $e(x)$ (см. табл. 2.1) и декодировать полученную комбинацию $r(x)$ синдромным методом декодирования.

Таблица 2.1

Полином ошибки.
По последней цифре номера зачетной книжки

Цифра	Полином ошибки	Цифра	Полином ошибки
1	$x + x^3$	6	$x^5 + x^9$
2	$x^3 + x^{10}$	7	$x^4 + x^9$
3	$x^2 + x^{11}$	8	$x^9 + x^{11}$

Продолжение табл. 2.1

Полином ошибки.
По последней цифре номера зачетной книжки

Цифра	Полином ошибки	Цифра	Полином ошибки
4	x^4+x^6	9	x^7+x^{12}
5	$1+x^7$	0	$x+x^8$

Для наложения ошибки используется функция `xor`.

```
> Merr=xor(Menc,Err)
```

Для декодирования используется функция `decode`. В качестве параметров задаются исходное сообщение в двоичном виде, параметры кода n и k и указание использовать циклический код. На выходе функция возвращает декодированное сообщение `Mdec`.

```
> Mdec=decode(Merr,n,k,"bch")'
```

Оператор транспонирования «'» указывается, чтобы выводить сообщение строкой, а не столбцом.

Другая функция, используемая для декодирования — `bchdeco`. В качестве параметров задаются исходное сообщение в двоичном виде и параметры кода k и t. На выходе функция возвращает декодированное сообщение `Mdec`.

```
> Mdec=bchdeco(Merr,k,t)
```

2.3.5.

Сравнить по методу Монте-Карло вероятностные характеристики двух кодов БЧХ с разной избыточностью. Параметры кодов приведены в табл. 2.2. Для сравнения воспользоваться написанной в листинге 2.1 программой, изменив ее для своих нужд, подставив необходимые параметры кодов. В результате выполнения будет получен график, который необходимо проанализировать. График и выводы должны быть представлены в отчете. Также необходимо проанализировать текст самой программы и разобраться в ее работе.

Листинг 2.1

Листинг программы для сравнения двух кодов БЧХ по вероятности битовой ошибки в канале ДСК

```
1  n1=63;
2  k1=39;
3  n2=37;
4  k2=19;
5  %
6  s1=sprintf("BCH code (%d,%d)",n1,k1);
7  s2=sprintf("BCH code (%d,%d)",n2,k2);
8  s3=sprintf("Without coding");
9  %
10 p0=[1e-3 5e-3 1e-2 5e-2 1e-1];
```

```
stat=zeros(3,5);
%
msg1=randi([0 1],1e3,k1);
msg2=randi([0 1],1e3,k2);
%
menc1=encode(msg1,n1,k1,"bch");
menc2=encode(msg2,n2,k2,"bch");
%
for i=1:1:5
  mrec1=bsc(menc1,p0(i));
  mdec1=decode(mrec1,n1,k1,"bch");
  [num,rate]=biterr(msg1,mdec1);
  stat(1,i)=rate;
  mrec2=bsc(menc2,p0(i));
  mdec2=decode(mrec2,n2,k2,"bch");
  [num,rate]=biterr(msg2,mdec2);
  stat(2,i)=rate;
  mrec3=bsc(msg1,p0(i));
  [num,rate]=biterr(msg1,mrec3);
  stat(3,i)=rate;
end
%
format long;
%
stat
%
mfig=figure;
L3=loglog(p0,stat(3,:));
set(L3,"LineWidth",3,"Color","r");
hold on;
L1=loglog(p0,stat(1,:));
set(L1,"LineWidth",1,"Color","k");
hold on;
L2=loglog(p0,stat(2,:));
set(L2,"LineWidth",3,"Color","b");
hold on;
title(sprintf("BCH codes (%d,%d) and (%d,%d) in BSC channel",
   n1,k1,n2,k2));
xlabel("BER in BSC channel, p0");
ylabel("Error rate after decoding");
legend(s3,s1,s2,0);
legend("show");
grid on;
print(mfig,'-dpng',sprintf("bch-%d-%d_cycl-%d-%d_bsc_err-rate
   ",n1,k1,n2,k2));
```

Программу необходимо сохранить в виде файла с расширением `*.m` и запускать из командной строки (из каталога, в котором лежит программа) так, как указанно ниже.

```
user@name:[~]$ octave -q bch_compar.m
```

Таблица 2.2

Параметры кодов БЧХ для сравнения.
(По последней цифре номера зачетной книжки)

Цифра	**Код 1** (n,k)	**Код 2** (n,k)	**Цифра**	**Код 1** (n,k)	**Код 2** (n,k)
1	(15,11)	(31,21)	6	(31,16)	(37,19)
2	(15,7)	(31,21)	7	(31,11)	(63,30)
3	(15,5)	(37,19)	8	(31,6)	(63,36)
4	(31,26)	(63,51)	9	(63,39)	(37,19)
5	(31,21)	(63,45)	0	(63,45)	(15,7)

2.3.6.

Подобрать три кода БЧХ, обеспечивающих исправляющую способность не хуже заданной. Выбрать из них код с наименьшей избыточностью. В качестве критерия исправляющей способности использовать вероятность ошибки декодирования (на выходе декодера). Для выбираемого кода она не должна превышать заданной при определенной вероятности ошибки в канале ДСК. Требования к исправляющей способности заданы в табл. 2.3 Для построения графиков вероятности ошибки декодирования можно воспользоваться представленной в листинге 2.1 программой (строит кривые вероятности ошибки декодирования одновременно для двух кодов).

Таблицу кодов БЧХ можно посмотреть командой `bchpoly`, запущенной без параметров. Она выдает параметры n, k и t для всех кодов БЧХ, поддерживаемых системой Octave. В качестве кодов БЧХ могут быть использованы и укороченные коды БЧХ, образованные из обычных кодов БЧХ путем уменьшения количества информационных символов. Например, укороченный код БЧХ (37,19), исправляющий три ошибки, был образован путем укорочения кода БЧХ (63,45).

Таблица 2.3

Требования к исправляющей способности кода БЧХ.
(По последней цифре номера зачетной книжки)

Цифра	**Вер-ть ош. в канале** p_0	**Вер-ть ош. декодир.**	**Цифра**	**Вер-ть ош. в канале** p_0	**Вер-ть ошибки декодир.**
1	$3\cdot10^{-2}$	$1\cdot10^{-3}$	6	$7\cdot10^{-3}$	$7\cdot10^{-5}$
2	$2\cdot10^{-2}$	$2\cdot10^{-3}$	7	$6\cdot10^{-3}$	$1\cdot10^{-5}$
3	$1\cdot10^{-2}$	$9\cdot10^{-4}$	8	$5\cdot10^{-3}$	$1\cdot10^{-5}$
4	$9\cdot10^{-3}$	$1\cdot10^{-4}$	9	$5\cdot10^{-3}$	$1\cdot10^{-4}$
5	$8\cdot10^{-3}$	$1\cdot10^{-4}$	0	$5\cdot10^{-3}$	$5\cdot10^{-5}$

2.4. Порядок защиты лабораторной работы

Защита работы может осуществляться одним из нижеперечисленных способов или их сочетанием на усмотрение преподавателя.

1. Устный ответ по теме работы.
2. Тестирование по теме работы
3. Задача по теме работы.
4. Иные варианты на усмотрение преподавателя.

Лабораторная работа 3
Коды Рида–Соломона

3.1. Цель работы

Рассмотреть на примере и получить навыки в исследовании недвоичных циклических кодов Рида–Соломона с использованием системы компьютерной алгебры CNU/Octave.

3.2. Рекомендуемая литература

1. Практика помехоустойчивого кодирования. Циклические коды : практикум / С. С. Владимиров, О. С. Когновицкий ; СПбГУТ. — СПб, 2019. — 44 с.

2. Е. Р. Алексеев, О. В. Чеснокова «Введение в Octave для инженеров и математиков» М. : ALT Linux, 2012. — 368 с.

3. Documentation // Octave-Forge.
URL: http://octave.sourceforge.net/docs.html

4. Е. Р. Алексеев, О. В. Чеснокова «Введение в Octave» // НОУ ИНТУИТ. URL: http://www.intuit.ru/studies/courses/3677/919/info

3.3. Порядок выполнения задания

Задание выполняется каждым учащимся индивидуально. Результаты заданий лабораторного практикума и соответствующих заданий практикума должны совпадать.

Отчёт формируется в электронном виде в формате PDF и отправляется на электронную почту преподавателя.

3.3.1.

Для (n,k)-кода РС $(15,9)$ определить образующий полином $g_{(15,9)}(x)$ и количество исправляемых кодом ошибок. Для этого воспользоваться функцией

```
> [g,t] = rsgenpoly(n,k)
```

На вход функции задаются параметры кода n и k. На выходе выводятся образующий полином g (в виде вектора элементов поля) и количество исправляемых кодом ошибок t. Более подробно работа функции описана во внутренней справке Octave — команда `help rsgenpoly`. Также нужно обратить внимание на то, какое поле по умолчанию использует функция. Это поле в последующих пунктах работы необходимо использовать для формирования информационного массива.

3.3.2.

Сформировать по номеру студенческого (зачетной книжки) информационный полином *Msg* для заданного в предыдущем пункте кода. Считать, что каждая цифра номера — это элемент поля Галуа, над которым строится код. Недостающую длину дополнить нулями.

Пример.
Пусть номер равен 143193 и поле Галуа имеет степень 4 и полином 19 (в десятичном виде). Код РС (15,9). Тогда информационное слово *Msg* задаётся как

```
> Msg = gf([0 0 0 1 4 3 1 9 3],4,19)
```

3.3.3.

Закодировать полученный в предыдущем пункте информационный вектор при помощи встроенных функций Octave.

Для кодирования используются функции `rsenc` (в качестве параметров задаются исходное сообщение в двоичном виде и параметры кода n и k).

```
> Menc=rsenc(Msg,n,k)
```

3.3.4.

Наложить на полученную в предыдущем пункте кодовую комбинацию *Menc* кода РС двукратную ошибку и декодировать комбинацию. Позиции ошибок взять как две последние (отличающиеся друг от друга) цифры номера зачетной книжки. Значения ошибок принять такими же.

Для рассматриваемого примера 143193.

Вначале задается нулевой массив.

```
> Err=gf(zeros(1,15),4,19)
```

Затем инициализируются соответствующие элементы массива.

```
> Err(3)=3;
> Err(9)=9;
> Err
```

Последняя команда выводит полученный массив на экран.

Для наложения ошибки суммируем массивы.

```
> Merr=Menc+Err
```

Для декодирования используется функция `rsdec`. В качестве параметров задаются исходное сообщение и параметры кода n и k. На выходе функция возвращает декодированное сообщение `Mdec`.

```
> Mdec=rsdec(Merr,n,k)
```

3.3.5.

По аналогии с ранее использованными программами для циклических кодов и кодов БЧХ составить программу, вычисляющую по методу Монте-Карло вероятностные характеристики кода РС и выводящую информацию в виде графика зависимости вероятности необнаруженной ошибки от битовой ошибки в канале.

Для определения числа ошибок в случае кода РС вместо функции `biterr` удобно использовать функцию `symerr`, считающую число символьных ошибок и вероятность символьной ошибки.

Вместо передачи кодовых комбинаций по каналу можно сформировать массив ошибки и сложить его с массивом информационных элементов.

Текст полученной программы и график вероятностных характеристик необходимо привести в отчете.

3.4. Порядок защиты лабораторной работы

Защита работы может осуществляться одним из нижеперечисленных способов или их сочетанием на усмотрение преподавателя.

1. Устный ответ по теме работы.
2. Тестирование по теме работы
3. Задача по теме работы.
4. Иные варианты на усмотрение преподавателя.

Владимиров Сергей Сергеевич

ПРАКТИКА ПОМЕХОУСТОЙЧИВОГО КОДИРОВАНИЯ. ЦИКЛИЧЕСКИЕ КОДЫ

Лабораторный практикум

Редактор *Х. Х. Хxxxxxxxx*

План издания 2020, п. ХХ

Подписано к печати ХХ.ХХ.ХХХХ
Объем 2,75 печ. л. Тираж ХХ экз. Заказ ХХХХ

Редакционно-издательский отдел СПбГУТ
193232 СПб., пр. Большевиков, 22
Отпечатано в СПбГУТ