Отчёт по первому заданию курса графических моделей «Алгоритм Loopy Belief Propagation для LDPC-кодов»

Кафедра математических методов прогнозирования

Бобров Евгений, 417 группа

5 апреля 2017

Помехоустойчивое кодирование

Рассмотрим задачу помехоустойчивого кодирования. Пусть кодируемая информация делится на фрагменты длиной K бит, которые преобразуются в кодовые слова длиной N бит. Тогда сообщения можно кодировать (N, K) блоковым линейным кодом. $M = N - K$ – число контрольных бит, $R = K/N$ – скорость кода. Предположим, что при передаче по каналу длина сообщения не изменяется, а каждый бит независимо инвертируется с некоторой вероятностью q. Задача алгоритма декодирования состоит в восстановлении переданного слова путём поиска среди всевозможных кодовых слов ближайшего к нему в метрике Хемминга. Найденное слово переводится в декодированное слово исходного сообщения. В данной работе исследовано LDPC-кодирование.

Вероятностная модель информационного канала с шумом:

$$p(e, s) \propto p(e)p(s|e) \propto \prod_n p(e_n) \prod_m [h_m^T e = s_m]$$

$e \in \mathbb{R}^N$ – вектор ошибок, $s \in \mathbb{R}^M$ – вектор синдромов

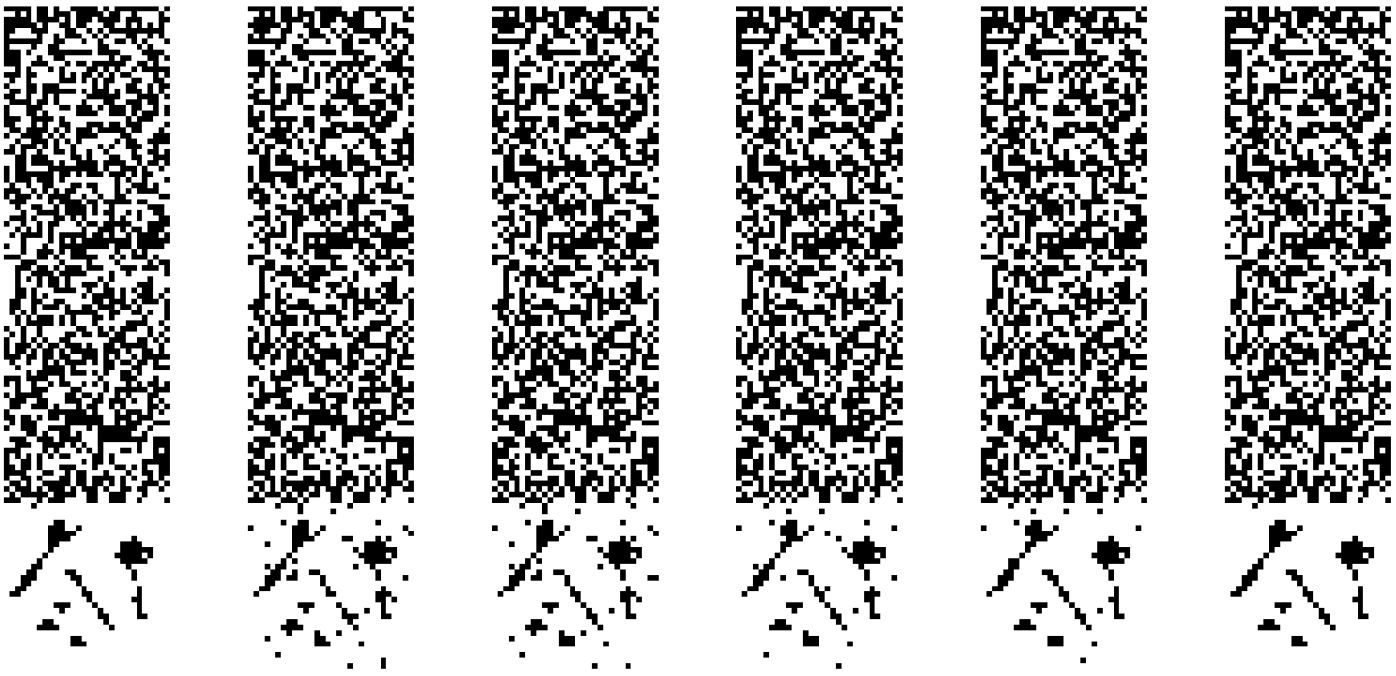
$h_m \in \mathbb{R}^N$ – вектор, m-я строка проверочной матрицы $H \in \mathbb{R}^{M,N}$

$p(e_n) = q^{e_n}(1 - q)^{1-e_n}$ – априорное распределение шума в канале связи

При использовании побитовой функции потерь $\lambda(e, \hat{e}) = \sum_n [e_n \neq \hat{e_n}]$ оптимальная процедура декодирования связана с максимизацией маргиналов отдельных переменных: $\hat{e}_n = \arg\max p(e_n|s)$. Распределения $p(e_n|s)$ рассчитываются циклическим алгоритмом передачи сообщений (sum-product loopy BP) на фактор-графе марковской сети, определяемой исходной вероятностной моделью. Тогда вершинами графа будут координаты e_n вектора e, а факторами – векторы h_m. Вершина e_n связана ребром с фактором h_m, тогда и только тогда когда n-я координата h_{mn} вектора h_m равна единице: $e_n = 1 \Leftrightarrow h_{mn} = 1$, что следует из условия проверок на чётность $h_m^T e$. Парадигма кодов с малой плотностью проверок на чётность указывает на сильную разреженность векторов h_m и матрицы H в целом.

Визуализация процесса декодирования

От исходного кодового слова, с последующей передачей по шумовому каналу, до полного декодирования принятого сообщения.



Эксперименты

Теорема Шеннона определяет пропускную способность канала как максимально допустимую скорость кода R, при которой возможно осуществление надежной коммуникации. Пропускная способность рассматриваемого канала определяется величиной $1 + q \log_2 q + (1 - q) \log_2(1 - q)$. Тогда при $q = 0.05$ пропускная способность рассматриваемого канала равна 0.7.

Построим LDPC-код с параметрами BCH-кода Хэмминга. Число информационных бит: $K = 4$, контрольных бит: $M = 3$. Построенные алгебраически, коды Хэмминга гарантируют исправление одной ошибки в полученном сообщении. Посмотрим, что будет, если применить вероятностный подход. Эксперимент разворачивается вокруг шести сообщений, переданных по каналу с шумом $q = 0.05$. Скорость рассматриваемого кода равна $R = \frac{K}{M+K} = 0.57$ и не превосходит пропускную способность. Исходные и исправленные ошибки представлены таблицами **??** и **??**. В столбцах записаны сообщения, в строках – их координаты. Синей единицей выделена верно исправленная ошибка, красным нулём – не исправленная ошибка, и красной единицей – ошибочно исправленная. Эксперимент показывает, что на кодах малой размерности алгебраические коды показывают существенно лучшие результаты по качеству декодирования. На малых длинах кода алгоритмы LDPC и BCH срабатывают одинаково быстро.

e_1	0	0	0	0	0	0
e_2	0	1	0	0	0	1
e_3	0	1	0	0	0	0
e_4	1	0	0	0	0	0
e_5	0	0	0	0	0	0
e_6	0	0	0	0	0	0
e_7	0	0	0	0	1	0

Таблица 1: Исходные ошибки

$\hat{e}_1$	0	0	0	0	0	0
$\hat{e}_2$	0	0	0	0	0	0
$\hat{e}_3$	1	0	0	0	0	0
$\hat{e}_4$	1	0	0	0	0	0
$\hat{e}_5$	1	0	0	0	0	0
$\hat{e}_6$	0	0	0	0	0	0
$\hat{e}_7$	0	1	0	0	1	0

Таблица 2: Ошибки, исправленные LDPC

Действительно, рассмотрим по порядку все сообщения от первого к шестому. При передаче первого сообщения была допущена одна ошибка в четвёртой координате. Алгоритм LDPC исправил ошибку, но и привнёс две ноновые в e_3 и e_5. Во втором сообщении LDPC пропустил обе ошибки e_2 и e_3. Третье и четвёртое сообщения были переданы без искажений, и без искажений декодированы. В пятом сообщении появилась одна ошибка e_7 и была успешно исправлена. В шестом сообщении ошибка обнаружена не было. Код BCH гарантированно без ошибок декодирует все сообщения, кроме второго, где появились две ошибки.

Данный эксперимент подверждает то, что алгоритм Loopy Belief Propagation для марковских сетей имеет плохую сходимость, когда сеть имеет короткие циклы, то есть достаточно плотную проверочную матрицу H. Причём нельзя сделать матрицу сильно разреженной на таких коротких кодах.

Существенно лучшие результаты LDPC показывает на кодах большей размерности. Уже для визуализации декодирования параметры линейного кода были: информационных бит $K = 900$ (картинка размера 30 на 30 двух чайных ложек, карандаша и трёх монеток), контрольных бит $M = 2700$. При том, что изображение было достаточно сильно зашумлено с $q = 0.05$, оно было полностью восстановлено за четыре итерации алгоритма.

Дальше, исследуем время работы методов на кодах разной размерности. Фиксируем $q = 0.05, R \sim 0.5$. Тогда $K = RN$.

K	4	573	1079	2033
N	7	1023	2047	4095
LDPC time in sec	0.04	1.75	3.3	11.6
BCH time in sec	0.001	1.3	4.8	19.2

Таблица 3: Исследования времени декодирования BCH против LDPC

При достаточно большой длине кодового слова алгоритм LDPC срабатывает быстрее BCH. Однако имея вероятностную природу LDPC не всегда исправляет ошибки там, где этой делает BCH. LDPC следует применять на кодах сверхбольших размеров, когда использование BCH технически невозможно. Также LDPC имеет смысл объединять в композиции. Так как после одного прохода LDPC фактически остаётся меньше ошибок в переданных блоках сообщений, то, используя несколько уровней декодирования, можно добиться высокого качества.

Дальше, методом стат испытаний оценим битовую $\frac{1}{N}\sum_n p(\hat{e}_n \neq e_n)$ и блоковую $p(\hat{e} \neq e)$ ошибки низкоплотностного кода в зависимости от различных показателей: N, q, R (введены выше), j – число единиц в столбцах проверочной матрицы H. j определяет количество смежных факторов для данной вершине, разреженность матрицы. Также исследуем время работы и долю расходимости алгоритма LDPC, параллельную и последовательную его реализации.

Последовательный алгоритм сходится значительно лучше параллельного. На низких скоростях R первый качественнее, а на высоких – второй. Последовательный работает до сотни раз дольше параллельного. Поэтому для остальных экспериментов возьмём именно параллельную версию. Интересно, что наибольшая расходимость достигается на скорости, равной половине порога пропускной способности канала.

При увеличении скорости R кода монотонно увеличиваются средние битовые и блоковые ошибки декодирования. Когда скорость кода превышает пропускную способность, то блоковая ошибка становится равной единице, а битовая – вероятности шума q. Что подтверждает теорему Шеннона. Качество растет при увеличении длины кодового слова N, равно как и при увеличении количества единиц j в столбце проверочной матрицы H. При увеличении интенсивности q шума ошибки увеличиваются.

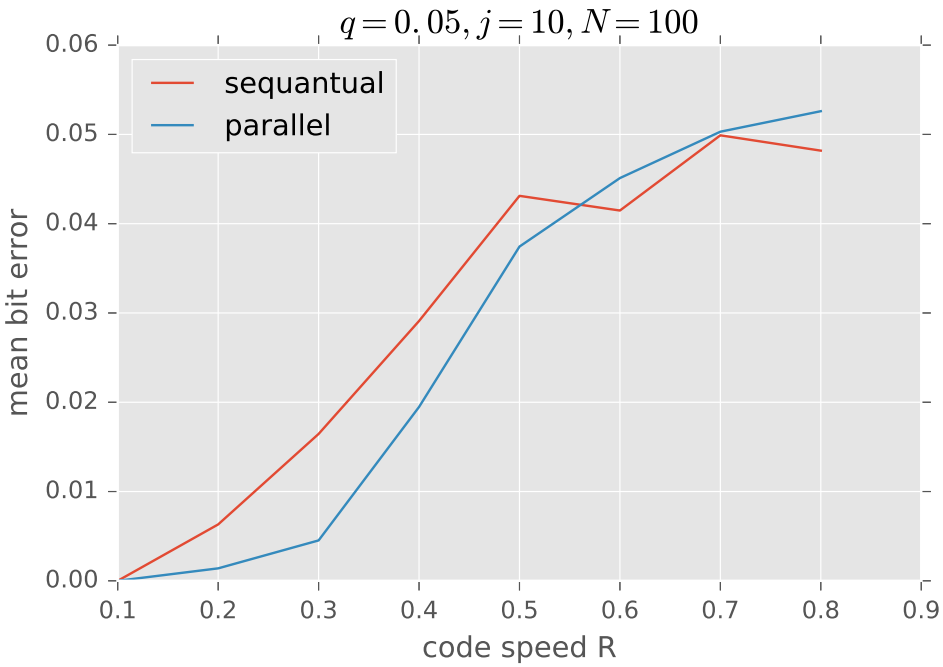


Рис. 1: Оценка битовой ошибки низкоплотностного кода для последовательного и параллельного алгоритмов

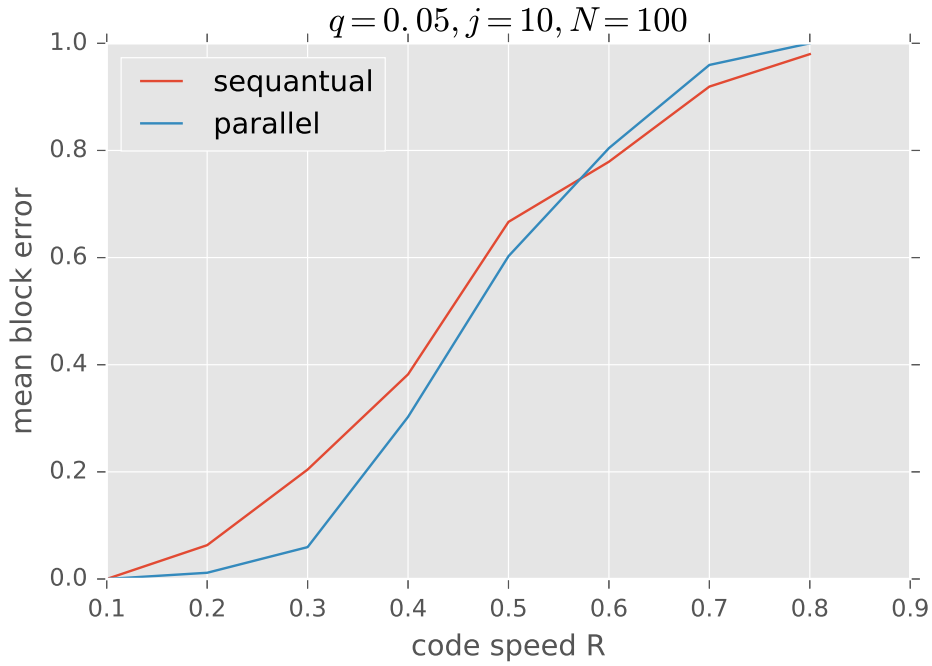


Рис. 2: Оценка блоковой ошибки низкоплотностного кода для последовательного и параллельного алгоритмов

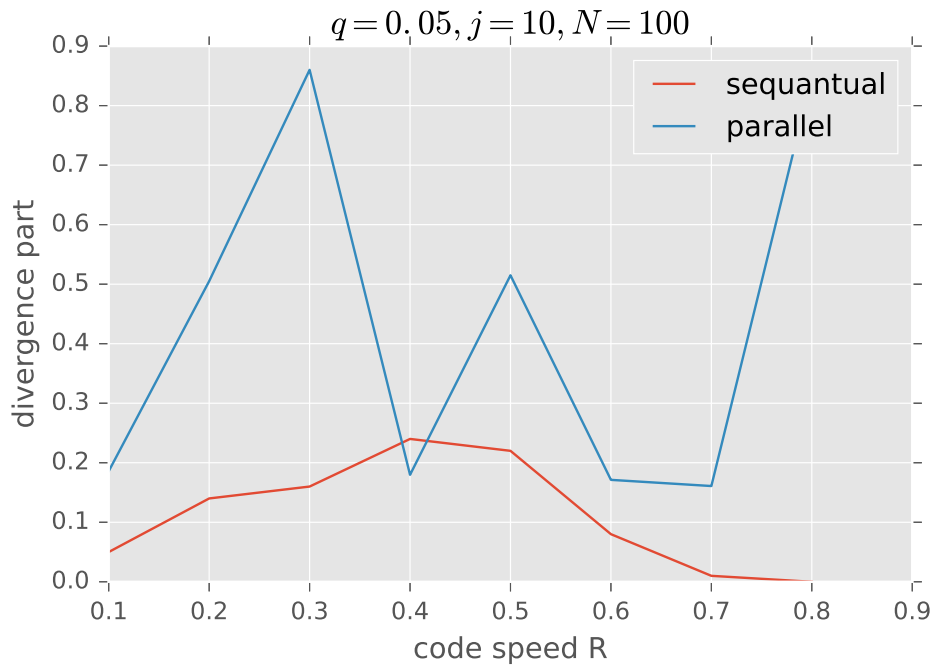


Рис. 3: Доля расходимости алгоритма для последовательного и параллельного алгоритмов

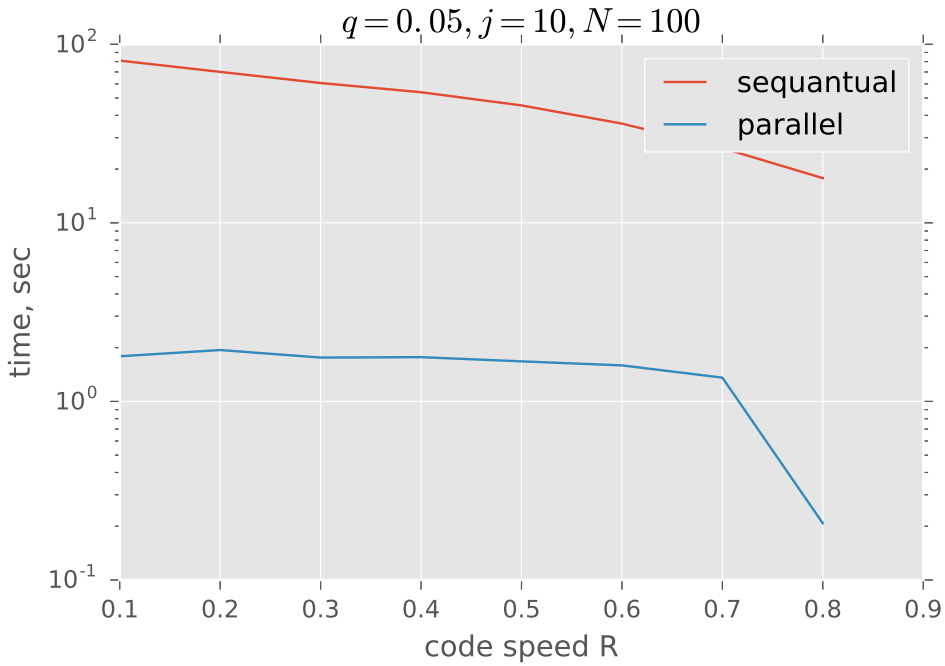


Рис. 4: Время декодирования для последовательного и параллельного алгоритмов

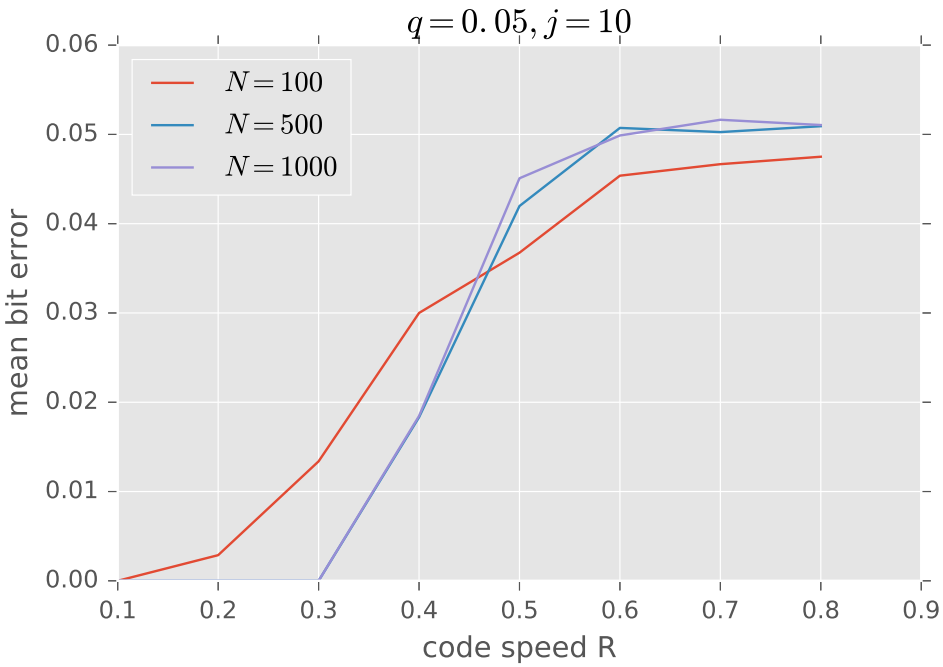


Рис. 5: Оценка битовой ошибки низкоплотностного кода для различных значений длины N кодового слова и скорости R кода

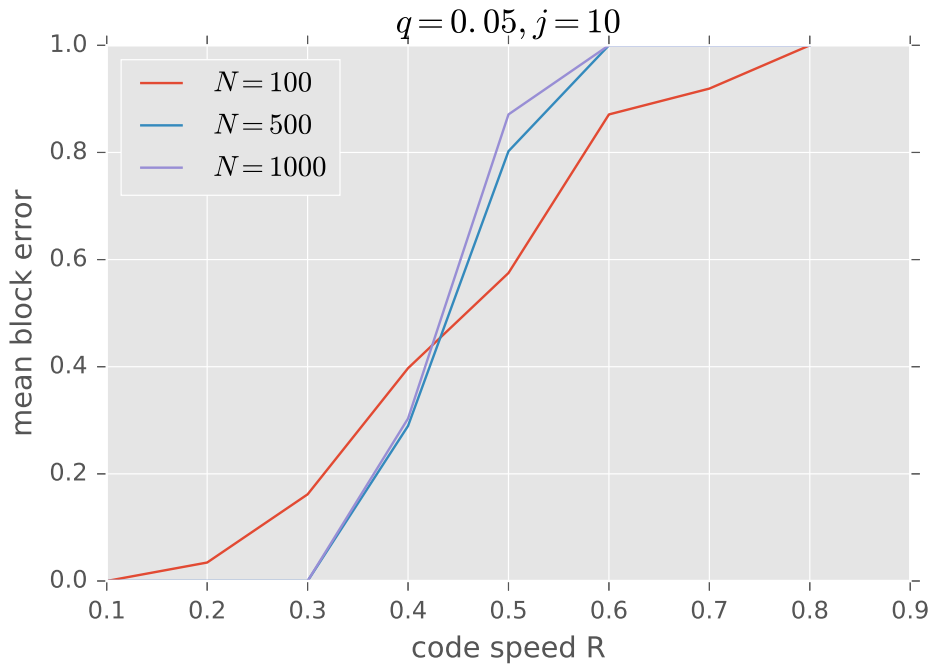


Рис. 6: Оценка блоковой ошибки низкоплотностного кода для различных значений длины N кодового слова и скорости R кода

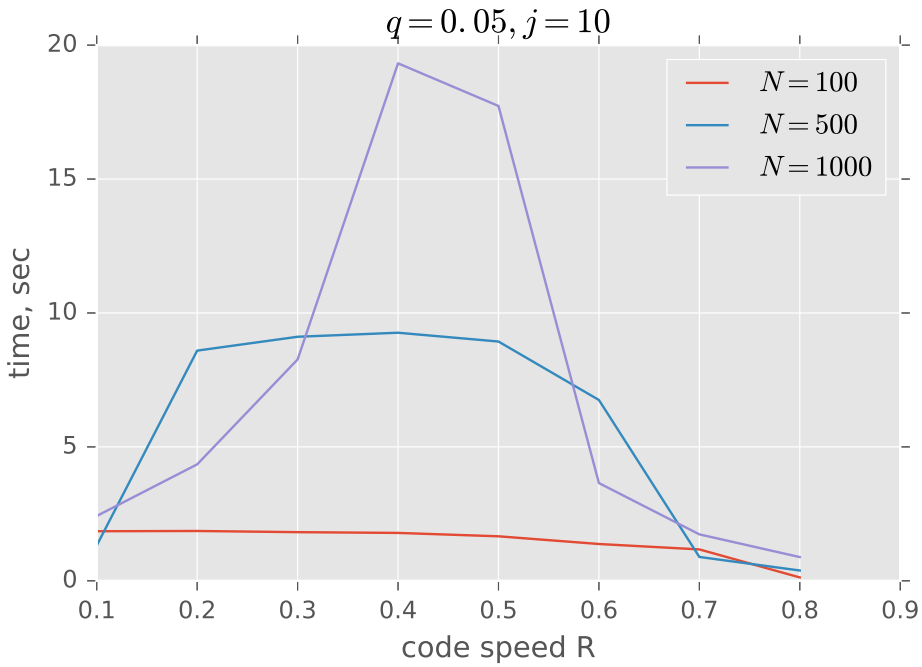


Рис. 7: Время декодирования для различных значений длины N кодового слова и скорости R кода

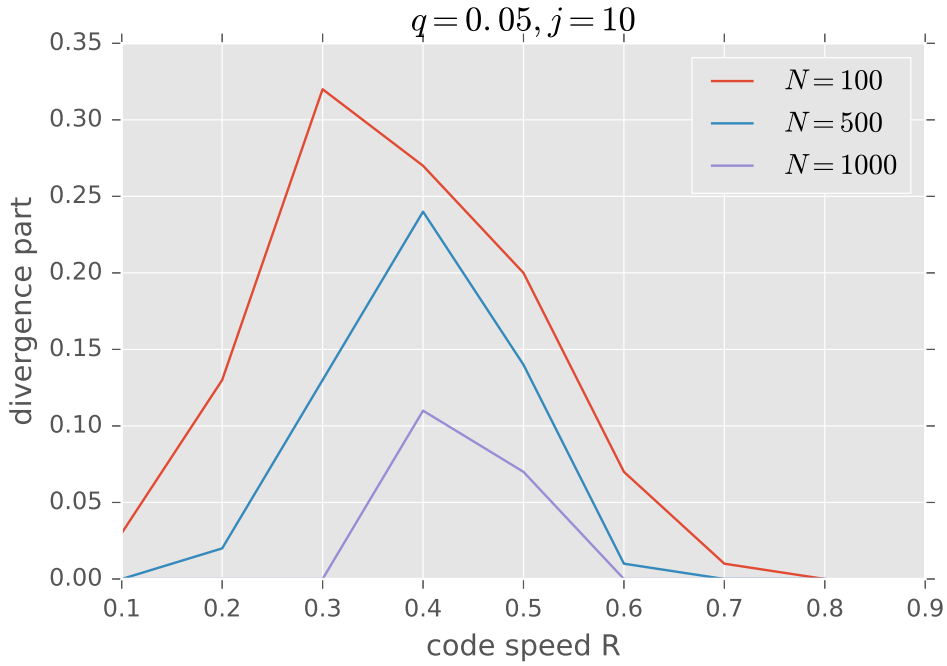


Рис. 8: Доля расходимости алгоритма для различных значений длины N кодового слова и скорости R кода

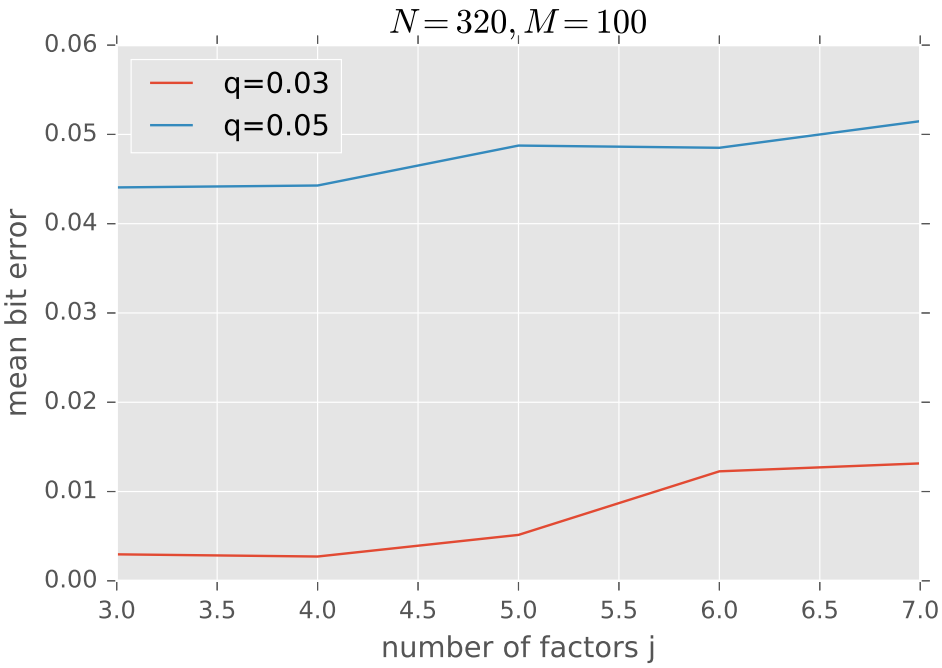


Рис. 9: Оценка битовой ошибки низкоплотностного кода для различного числа j единиц в проверочной матрице и различных скоростях R кода

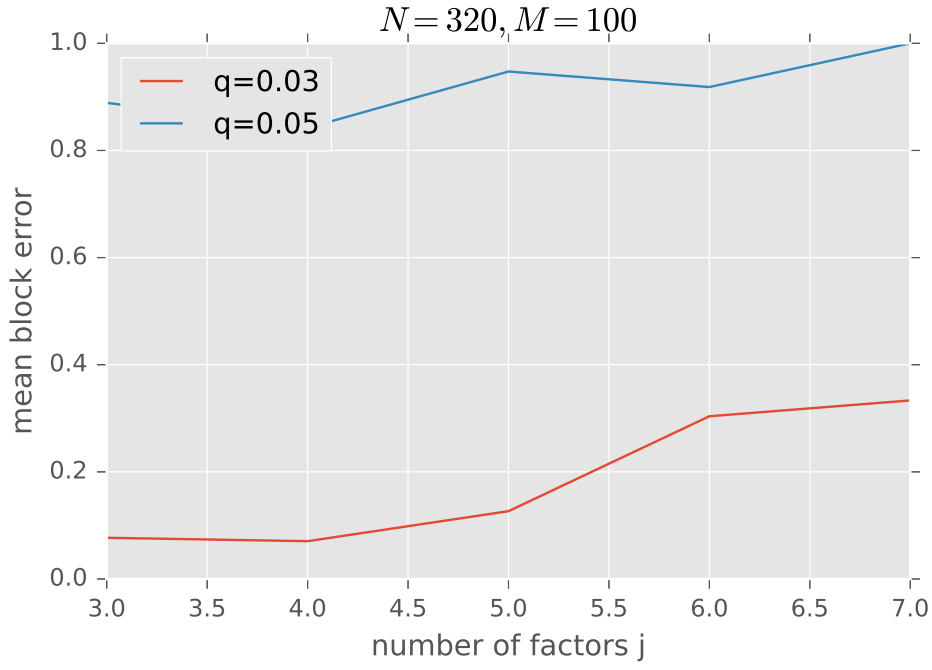


Рис. 10: Оценка блоковой ошибки низкоплотностного кода для различного числа j единиц в проверочной матрице и различных скоростях R кода