Московский Государственный Университет
им. М.В. Ломоносова
Механико-математический факультет
Кафедра Математической теории интеллектуальных систем

Дипломная работа по теме:

Динамическое согласование сигналов

Выполнил: Петюшко А.А., группа №508
Научный руководитель: профессор, д.ф.-м.н. Бабин Д.Н.
Рецензент: к.ф.-м.н. Мазуренко И.Л.

Москва, 2006

1 Метод динамического искажения времени.

1.1 Автоматическое распознавание речи.

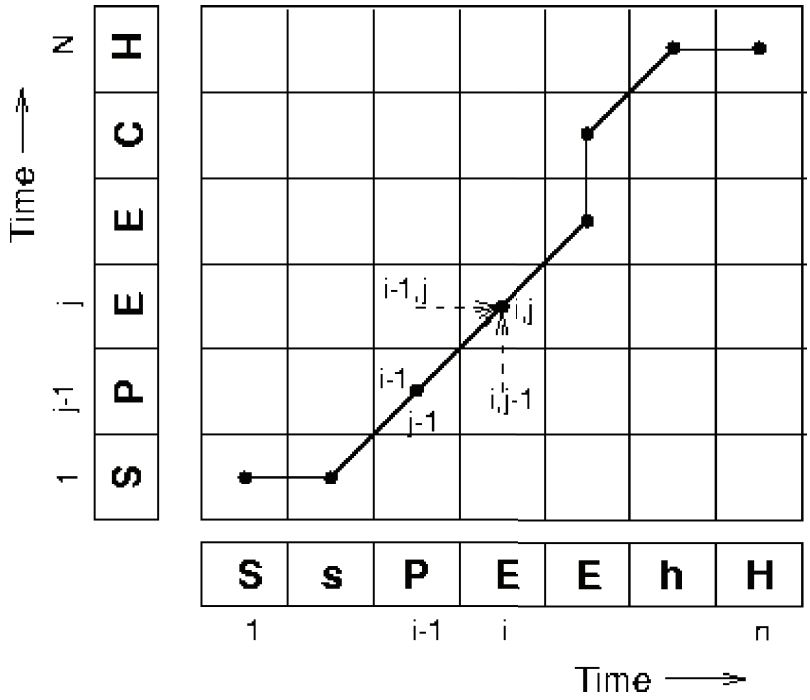
Автоматическое распознавание речи (Automatic speech recognition, ASR) является важной прикладной задачей в настоящее время. Хорошо изученный подход для автоматического распознавания речи основан на методе сравнения распознаваемого сигнала с эталоном для каждого слова в используемом словаре (более подробно о методах распознавания, в т.ч. и о методе сравнения с эталоном, см. последующие главы). Эталон, наиболее близкий к анализируемому сигналу (мера близости выбирается на основе некоторого расстояния или метрики), считается результатом распознавания. Некоторые алгоритмы для нахождения наилучшего соответствия (наиболее близкого эталона) базируются на использовании динамического программирования (Беллман). Рассмотрим метод динамического искажения (деформации) времени, более известный как Dynamic Time Warping - DTW.

1.2 Описание метода.

Речь - это процесс, зависящий от времени. Несколько произношений одного и того же слова обычно имеют разную длительность, однако даже при одинаковой длительности произношения одного слова будут различаться в середине (или любой другой части), т.к. разные части слова произносятся с разной скоростью. Для получения глобального расстояния между двумя образцами речи, представленными как последовательность многомерных векторов, применяется метод динамического искажения времени.

Проблема соответствия сигналов проиллюстрирована на рисунке ниже, на котором изображена матрица типа "время-время": эталон соответствует вертикали (слева), входной сигнал - горизонтали (внизу). В данном примере сравниваются два произнесения английского слова "speech" (речь): входной сигнал "SsPEEhH", который является зашумленным вариантом эталона "SsPEEhH". Звук "h" наиболее похож на "H". Входной сигнал "SsPEEhH" будет сравниваться со всеми эталонами из имеющейся базы данных. Наиболее соответствующим эталоном для входного сигнала будет считаться тот, у которого вес пути соответствия между ним и входным сигналом будет наименьшим. Глобальным расстоянием между сигналами, или весом пути, будет считаться сумма весов отдельных ячеек, составляющих путь.

Рис.: Иллюстрация пути соответствия между эталоном "SPEECH" и зашумленным сигналом "SsPEEhH".



Как найти путь наилучшего соответствия (что равносильно нахождению глобально минимального расстояния) между входным сигналом и эталоном? Мы можем перебрать все возможные пути - но это очень неэффективно, т.к. число всех возможных путей экспоненциально зависит от длины входного сигнала. Вместо этого, мы учтем ограничения на процесс нахождения соответствия (которые существуют или мы сами можем наложить на этот процесс) и с учетом этих ограничений построим эффективный алгоритм. Предположим следующее:

- Соответствие между сигналами не может идти назад по времени,
- Каждое окно входного сигнала должно входить в путь соответствия
- Расстояния между ячейками складываются для получения глобального расстояния

Теперь можно сказать, что каждое окно эталона и каждое окно входного сигнала должно входить в путь соответствия. Это значит, что для точки (i, j) в матрице расстояний типа "время-время" (где i указывает на номер окна входного сигнала, j - на номер окна эталона) предыдущими точками в пути соответствия могут быть лишь $(i-1, j-1)$, $(i-1, j)$ или $(i, j-1)$ (см. рисунок выше). Ключевой идеей динамического программирования является то, что для получения оптимального пути в точке (i, j) мы просто продолжаем оптимальный путь для точек $(i-1, j-1)$, $(i-1, j)$ или $(i, j-1)$.

Этот алгоритм носит название динамического программирования - ДП (Dynamic Programming, DP). В приложении к распознаванию речи методом сравнения с эталоном, он обычно называется методом динамического искажения времени - ДИВ (Dynamic Time Warping, DTW). ДП гарантированно находит оптимальный путь по матрице расстояний (соответствия), в то же время минимизируя вычислительную сложность: для эталона длины N число учитываемых путей в каждый момент времени прохода входного сигнала равно N.

Пусть $D(i, j)$ - это вес оптимального пути до точки (i, j) включительно, а вес самой ячейки (i, j) - это $d(i, j)$. Тогда

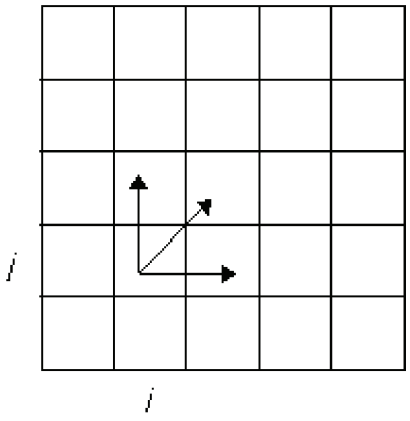
$$D(i, j) = \min[D(i-1, j-1), D(i-1, j), D(i, j-1)] + d(i, j)$$

Принимая за начальное условие $D(1,1) = d(1,1)$ (инициализация), мы имеем базу для эффективного рекурсивного алгоритма для вычисления $D(i,j)$. Итоговый вес оптимального пути $D(n,N)$ дает нам расстояние между эталоном и входным сигналом. Результатом распознавания входного слова считается значение того эталона, который дает минимальный вес оптимального пути (замечание - N будет уникальным для каждого эталона).

Стандартный метод ДП для распознавания речи имеет малые требования по памяти, т.к. кроме эталона и входного сигнала нужно хранить только одну строку (столбец) матрицы расстояний.

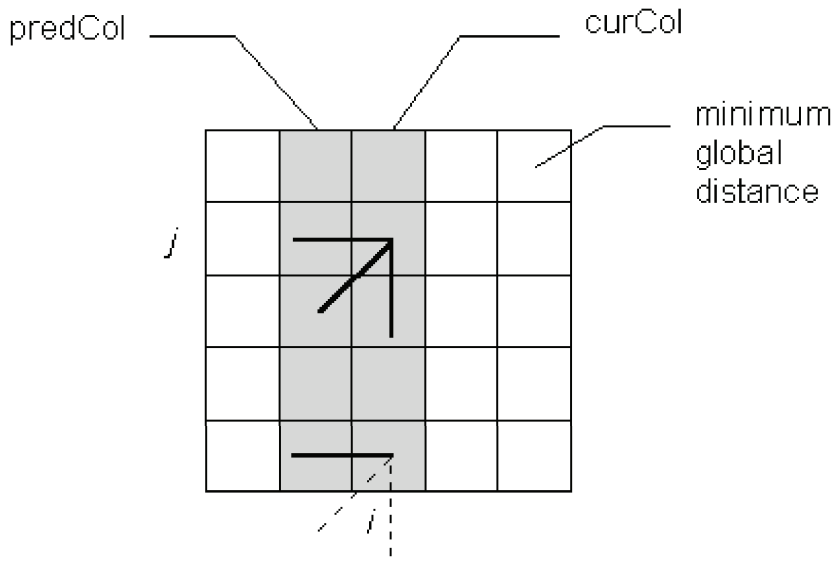
Считаем, что номера строк и столбцов нумеруются с 0 и далее по возрастающей. Это значит, что три направления, в которых мы можем двигаться для поиска оптимального пути из точки (i,j), это (см. рисунок):

Figure 1. The three possible directions in which the best match path may move from cell (i,j) in symmetric DTW.



Вычислительно приведенная выше формула для расчета $D(i,j)$ уже находится в той форме, которую можно рекурсивно запрограммировать. Однако, если язык программирования не оптимизирован под рекурсию, такая реализация метода будет очень медленной даже для относительно малых размеров сигналов. Есть другой метод, который и быстрее, и требует меньше памяти - и этот метод использует два вложенных цикла (например, for). Данная реализация метода требует памяти только под хранение двух массивов, которые содержат два смежных столбца матрицы расстояний.

Figure 2. The cells at (i,j) and (i,0) have different possible originator cells. The path to (i,0) can only originate from (i-1,0). However, the path to (i,j) can originate from the three standard locations as shown.



Опишем алгоритм нахождения оптимального пути (и, соответственно, расстояния между сигналами):

- Рассчитываем значения для нулевого столбца. Значение $D(0,0) = d(0,0)$. Затем вычисляем оставшиеся значения $D(0,j)$ как $D(0,j) = D(0,j-1)+d(0,j)$. Называем этот столбец предыдущим.

- Переходим к следующему столбцу и называем его текущим. Вычисляем значение для нижней ячейки для текущего столбца $D(i,0)$ как сумму веса оптимального пути в нижний столбец предыдущего и веса самой нижней ячейки текущего столбца, т.е. $D(i,0) = D(i-1,0) + d(i,0)$.

- Вычисляем оставшиеся значения $D(i,j)$ по формуле $D(i,j) = \min[D(i-1,j-1), D(i-1,j), D(i,j-1)] + d(i,j)$.

- Текущий столбец переименовываем в предыдущий и повторяем шаг 2 до тех пор пока не вычислим значения $D(i,j)$ для всех столбцов.

- Вес оптимального пути (и, соответственно, расстояния между сигналами) будет помещен в самый верхний элемент последнего столбца.

1.3 Реализация алгоритма DTW для распознавания отдельных команд.

На языке VC++ 6.0 с использованием пакетов RPL и SPL, а также в математической среде программирования MatLab 6.5 был реализован алгоритм DTW для распознавания отдельных команд (являющихся произнесением цифр от 0 до 9), записанных в wav-файлы.

Процедура работы программы была следующая:

- Обучение.
  Сначала надиктовывалось по 10 произнесений каждой команды, затем с помощью метода DTW находились попарные расстояния между этими произнесениями. В качестве эталона бралось то произнесение, которое было минимаксом.

- Распознавание.
  Опять же с помощью метода DTW находились расстояния от произнесенной распознаваемой команды до каждого из эталонов. Результатом распознавания считалось значение того эталона, до которого расстояние было наименьшим.

- Замечание к распознаванию.
  Чтобы избежать вышеописанных проблем с тем, что разные эталоны имеют различную длину (например, "два" и "четыре"), вследствие чего вес оптимального пути между короткими командами заведомо меньше, чем меньше длинными, несмотря на их похожесть, вес пути делился на длину пути (или площадь матрицы).

- Результаты.
  В результате была получена вероятность распознавания порядка 70-80%.

- Замечание к результатам.
  Нужно заметить, что система была дикторозависимой - т.е. точность распознавания падала на порядок, если эталоны и распознаваемая команда записана от разных людей (это является следствием применения параметров - кепстральных коэффициентов, которые использовались для представления окон звуков).

2 Метод нелинейного сжатия-растяжения матриц.

2.1 Постановка задачи.

К уже существующим методам распознавания требуется добавить еще один, вычисляющий некоторым образом функцию расстояния между звуковыми сигналами (командами), чтобы:

- Сократить перебор в словаре,
- Эта дополнительная функция расстояния была пригодна для увеличения надежности результатов распознавания в условиях нестационарного шума.

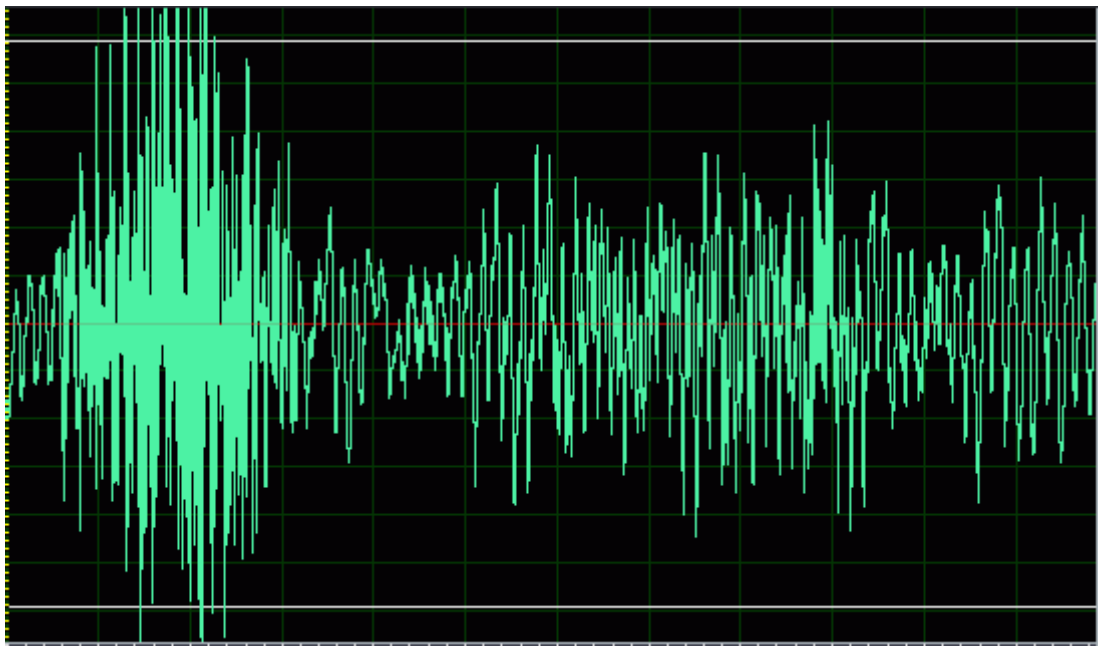
Т.к. этот метод необходим для повышения надежности распознавания, то от него не требуется распознавания как такового, т.е. это не означает, что расстояние между командами должно быть маленьким только для сигналов, соответствующих одному и тому же слову.

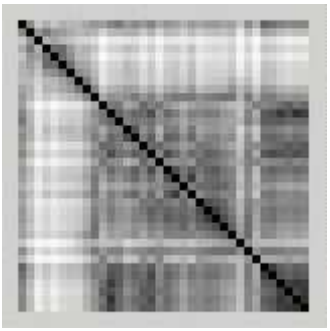
Предлагается в дополнение к известному методу динамической деформации времени (DTW) применить метод динамического программирования для нелинейного растяжения-сжатия для нахождения расстояния между уже не одномерными сигналами, а их матрицами самосравнения.

Матрица самосравнения для сигнала определенной длины вычисляется следующим образом:

1. Сигнал разбивается на окна одинаковой длины (возможно, частично перекрывающиеся),
2. В каждом окне вычисляется вектор параметров (спектральных, кепстральных, коэффициентов линейного предсказания и т.п. – в зависимости от реализации),
3. На основе некоторой метрики (например, евклидовой) вычисляется расстояние для каждой пары векторов из соответствующих окон,
4. На место (i, j) в матрице самосравнения заносится вычисленное в п. 3 расстояние между i-м и j-м вектором параметров.

Например, на рис. изображены команда ”один” и ее матрица самосравнения:





На функцию расстояния между матрицами самосравнения сигнала налагаются следующие ограничения:

1. Расстояние инвариантно относительно изменения длины сигналов, т.е. размеров квадратных матриц, которые мы сравниваем.

2. Полученная функция расстояния соответствует визуальной похожести матриц, определяемой человеком.

3. Расстояние между "родными"матрицами (являющимися самосравнениями произнесений одних и тех же команд) в среднем НЕ БОЛЬШЕ чем расстояние между матрицами из разных классов.

4. Описанное в п.2. условие "инвариантно"относительно сравниваемых пар классов - т.е., например, если у двух команд матрицы совсем разные, то расстояния между этими матрицами должны быть почти всегда большими, и наоборот: если при каких-то условиях у двух матриц из разных классов расстояние получилось очень маленьким, то такая ситуация должна быть не случайностью, а закономерностью, понятно, откуда взявшейся.

5. Желательно, чтобы вышеописанные требования выполнялись не только для матриц самосравнений сигналов в одном уровне шума, но и для матриц, вычисленных по сигналам для разных типов шума.

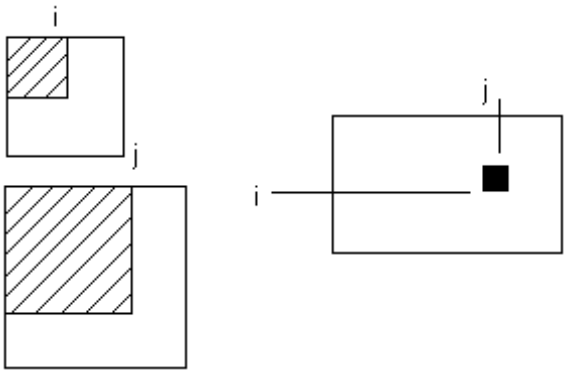
2.2 Описание метода.

По аналогии с методом динамической деформации времени для одномерного сигнала, предлагается использовать так называемый метод нелинейного сжатия-растяжения матриц. Необходимость его применения в том, что, как и в случае с одномерным сигналом, два произнесения последнего обычно находятся в нелинейном соответствии друг с другом (например, один звук растянут в 1.5 раза, а другой, наоборот, сжат в 2 раза). Соответственно, и матрицы самосравнения одной команды также нелинейно зависят друг от друга. Следовательно, надо таким образом их растянуть (будем именно растягивать матрицы) до одинакового размера, чтобы:

- Последовательность растяжений минимизировала расстояние между матрицами,

- Можно было ввести функцию расстояния между матрицами одинаковой размерности.

Далее опишем процедуру для растяжения двух матриц для нахождения расстояния между ними. Для этого строится третья матрица – матрица растяжений (или матрица пути), на месте (i, j) которой находится последовательность растяжений, минимизирующая расстояние между главными минорами (квадратными матрицами из левого верхнего угла) размера $i * i$ и $j * j$ соответственно, и само это расстояние (см. рис.):



Первая строка (и первый столбец) матрицы растяжений формируются просто: левый верхний элемент первой матрицы (для столбца – левый верхний элемент второй матрицы) копируется на все координаты вектора длины k, и на место $(1, k)$ заносится сумма расстояния между этим вектором и вектором такой же длины, являющемся началом k-ой строки второй матрицы, и числа, записанного в клетке $(1, k-1)$ (для первого столбца все аналогично с точностью до замены первой матрицы на вторую). Одновременно в каждую клетку заносится путь, которым мы в нее пришли, т.е. последовательность пар чисел, в котором первое число отвечает за номер строки (или столбца, что то же самое, т.к. матрицы самосравнения симметричны) первой матрицы, а второе – за номер строки (столбца) второй матрицы. Для остальных клеток матрицы растяжений поступаем следующим образом:

1. Пусть надо посчитать путь и расстояние для клетки с номером (i, j).

2. Берем клетку с номером $(i-1, j-1)$. Пусть длина пути, ведущего в нее, равна n, а сам путь – $d(i-1, j-1)$.

3. Считаем расстояния между следующими векторами, длина которых равна $(n+1)$:

   - на k-ом месте $(1 \leqslant k \leqslant n)$ первого вектора стоит элемент (i, 1-ое число k-ой пары пути $d(i-1, j-1)$) первой матрицы,
   - на k-ом месте $(1 \leqslant k \leqslant n)$ второго вектора стоит элемент (j, 2-ое число k-ой пары пути $d(i-1, j-1)$) второй матрицы,
   - на $(n+1)$ месте первого вектора стоит элемент (i, i) первой матрицы,
   - на $(n+1)$ месте второго вектора стоит элемент (j, j) второй матрицы.

   Замечание. Таким образом мы учитываем предыдущие растяжения матриц.

4. Считаем аналогично расстояния для клеток $(i-1, j)$ и $(i, j-1)$ (с путями соответственно $d(i-1, j)$ и $d(i, j-1)$).

5. Суммируем полученные расстояния со значениями в клетках $(i-1, j-1)$, $(i-1, j)$ и $(i, j-1)$ соответственно. Получим три числа.

6. Выбираем из этих трех чисел наименьшее, записываем его в клетку с номером (i, j), а в путь для этой клетки заносим путь, соответствующий наименьшему числу, в конец которого добавлена клетка, из которой пришли (т.е. для которой получилось наименьшее число).

В итоге в последней (правой нижней) ячейке матрицы растяжений мы получим путь – последовательность растяжений, которая минимизирует расстояние между матрицами.

Далее, в качестве расстояния можно взять число, записанное в последней ячейке матрицы растяжений, а можно сначала растянуть матрицы до одинакового размера в соответствии с полученным минимизирующим путем, а затем взять какую-нибудь матричную норму от разности уже растянутых матриц.

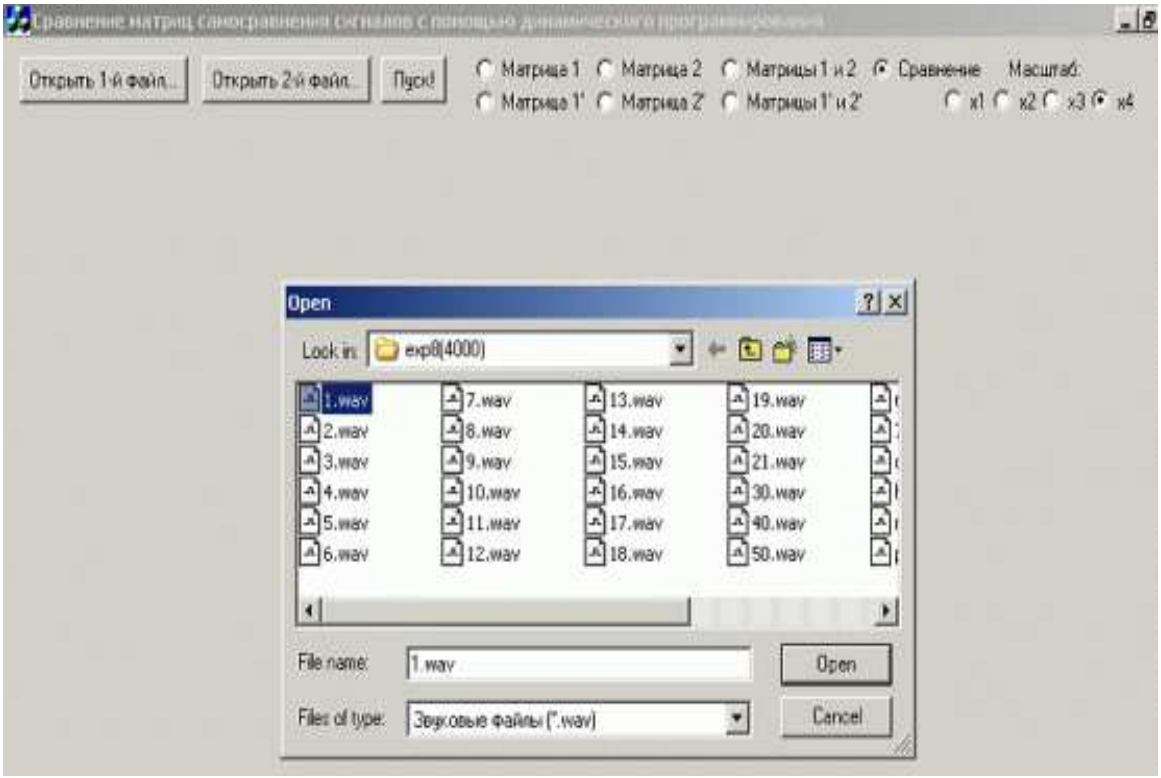
2.3 Конкретная реализация метода (программа).

Чтобы вышеописанный метод удовлетворял поставленным требованиям на функцию расстояния между матрицами, во-первых, расстояния, вычисляемые в п. 3 предыдущего параграфа (а также при вычислении расстояния между векторами при заполнении первой строки и первого столбца), делятся на длину векторов (т.е. на $(n+1)$), и, во-вторых, матричная норма от разности растянутых матриц делится на эквивалент длины пути – например, на площадь растянутых матриц, либо на длину диагонали. Таким образом, п. 2 требования должен быть выполнен.

П.3 требования должен быть выполнен, исходя из построения алгоритма.

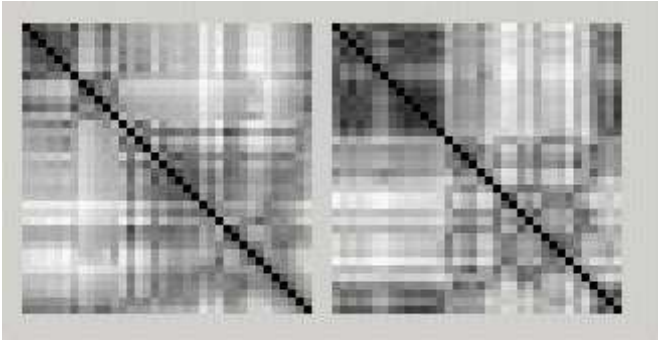
Для вычисления параметров будем использовать: шаг анализа wav-файлов - 128 отсчетов; размер окна анализа – 512; число коэффициентов линейного предсказания – 15; количество компонент спектра линейного предсказания – 256.

Будем работать с командами, записанными в отдельных wav-файлах. Следовательно, пользователю предлагается выбрать как файл, содержащий первую команду, так и файл, содержащий вторую команду. Внешний вид программы при выборе файла команды см. на рис.:



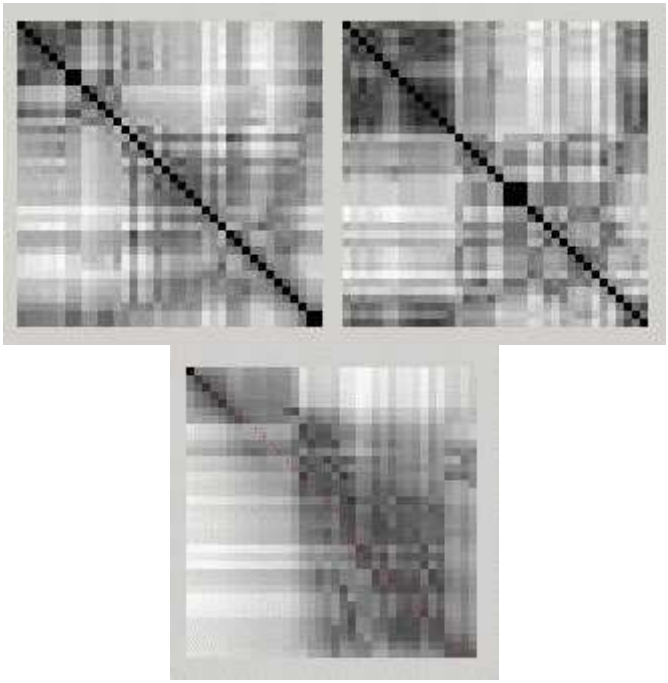
После открытия файла можно сразу же посмотреть графическое изображение матрицы самосравнения (в ее ячейках также нормированные числа, т.е. от 0 до 1), причем чем больше число, тем оно ближе к белому (и наоборот, чем меньше, ближе к нулю, тем темнее). После того, как загрузили файлы для обеих команд, можно посмотреть

обе матрицы одновременно, и визуально попробовать определить, насколько они близки (см. рис.):

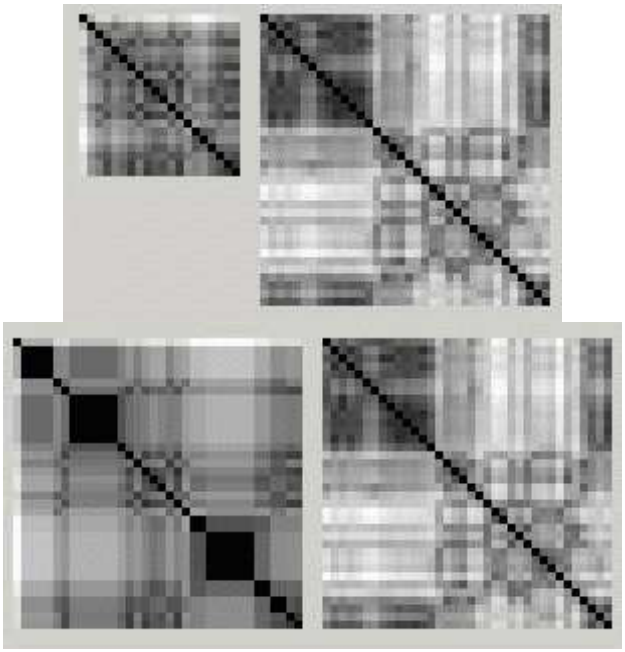


Затем после нажатия кнопки “Пуск!” программа вычисляет расстояние между матрицами и искомое растяжение матриц, причем последнее сразу выдается внизу экрана (см. рис.):

Далее можно просмотреть как обе растянутые матрицы, так и по отдельности, а также матрицу растяжений с помеченным в ней минимизирующим путем (см. рис.):



Также, в программе предусмотрена возможность масштабировать картинки, увеличивая их в 2, 3, 4 раза. В данном случае использовались команды для слова “4”, записанные в автомобиле с оборотами двигателя 4000 и 800 соответственно. А вот для команд, например, “1” и “4” получатся следующие матрицы самосравнения (справа - растянутые), расстояние между которыми 0.3128:



2.4 Введение весов

И при сравнении растянутых матрицы поэлементно, и в методе динамического программирования при формировании растяжения матриц, мы как бы считаем равноправными все пары окон звука внутри сигналов, но реально это не так. Если произнесенное слово – длинное и в нем есть два одинаковых звука (в начале и в конце слова, например), то его начало и конец могут произноситься по-разному хотя бы потому, что шумовой фон за это время успел измениться, или голос человека изменил окраску и т.п. Соответственно, при вычислении матрицы самосравнения может получиться и так, что соответствующие этим звукам ячейки таблицы окрашены в черный цвет, и в белый, т.е. справедливо следующее:

- два одинаковых звука в одном слове могут произноситься по-разному, но
- два разных звука в одном и том же слове не могут быть близкими в смысле используемой метрики.

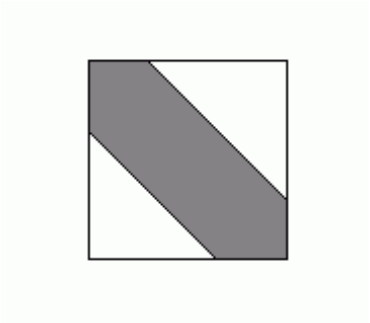
Чтобы учесть это обстоятельство, были введены весовые коэффициенты, которые тем меньше, чем дальше сравниваемые звуки друг от друга. Во-первых, их можно ввести в методе динамического программирования при формировании растяжения матриц, т.е. путь в матрице растяжений будет зависеть и от того, близки или далеки звуки друг от друга, и, во-вторых, их можно использовать при определении метрики на растянутых матрицах, т.к. и при сравнении матриц мы должны требовать близость элементов вокруг их диагонали (т.е. соответствующие звукам, расположенным во времени близко), в то время как отдаленные от диагонали элементы матриц (соответствующие звукам, расположенным далеко во времени) не обязаны иметь близкие значения, т.к. это результат сравнения очень далеких друг от друга звуков.

Также весовые коэффициенты помогут избежать ситуацию, когда в одном слове два одинаковых слова произнесены одинаково, а в другом - по-разному. В этом случае без введения весов сравнение матриц даст большое расстояние.

Предлагается весовые коэффициенты вводить либо как обратно пропорциональные расстоянию до диагонали (тогда больший вес будут иметь близко расположенные звуки), либо соответствующие любой другой зависимости, т.ч. при отдалении от диагонали

веса уменьшаются, - например, весовые коэффициенты, равномерно увеличивающиеся при приближении к диагонали, но не обратно пропорциональные (разность с единицей имеет квадратичную зависимость, коэффициенты меньше меняются и ближе к единице).

С другой стороны, так как алгоритм несимметричен (растяжение при выходе из левой верхней ячейки не равно растяжению при выходе из правой нижней ячейки матрицы), предлагается вообще "отсекать" - т.е. брать с нулевым весом – клетки матриц, лежащих вне некоторой полосы, параллельной главной диагонали (см. рис.):



Такое введение весовых коэффициентов может сделать алгоритм сравнения матриц более "симметричными", т.е. устранит эффект накопления длины вектора параметров по мере передвижения к правому нижнему углу матриц. Тогда сравнение матриц при прямом проходе (из левого верхнего угла в правый нижний) и наоборот должны получаться более-менее одинаковыми.

2.5 Проведенные эксперименты.

Изначально пробовались различные комбинации весовых коэффициентов для 15 коэффициентов линейного предсказания для матриц самосравнения команд, произносимых в автомобиле с 4000 и 800 оборотами двигателя (именно для этих значений оборотов двигателя наблюдается самая большая разница в отношении сигнал\шум). Были испробованы следующие комбинации: в методе динамического программирования без весовых коэффициентов, полосой, обратно пропорциональные и уменьшающиеся более медленно (разность их с единицей имеет квадратичную зависимость); для метрики между матрицами без весовых коэффициентов, обратно пропорциональные и уменьшающиеся более медленно. Полоса для подсчета расстояния между уже растянутыми матрицами не применялась, поскольку в силу обнуления значительной части клеток матрицы расстояние получалось слишком маленьким, и разница для разных команд совсем незначительная.

Далее, на основе выявленных закономерностей, были проведены опыты с меньшим числом коэффициентов линейного предсказания (от 14 до 1) сначала без весовых коэффициентов, а затем для тех значений числа коэффициентов линейного предсказания, показавших наилучшие результаты, были проведены опыты с различными комбинациями весовых коэффициентов.

На основе опытов были выявлены как наилучшие, наиболее подходящие для нашей задачи комбинации весовых коэффициентов и число коэффициентов линейного предсказания, так и команды, которые наиболее соответствуют поставленным требованиям.

2.6 Результаты экспериментов.

- Чем более похожи матрицы самосравнения, тем меньше расстояние.

- Обратно пропорциональные коэффициенты показывают результаты немного хуже, чем квадратичная зависимость, а она дает результат хуже, чем вообще без коэффициентов, однако разница небольшая.

- Применение полосы, да еще в сочетании с весовыми коэффициентами, дает ощутимо лучшие результаты.

- Хотя и не намного, но наиболее хорошие результаты дают 14 коэффициентов линейного предсказания.

- Что касается похожести матриц, то, насколько можно судить по экспериментам, все методы (с весовыми коэффициентами, или без них) прежде всего делают похожими (визуально) структуру матриц (всю или только около диагонали, зависит от применения весов), однако это не означает маленькое расстояние между ними. Это также объясняет, почему маленькие матрицы могут быть похоже на большие – хотя структура не очень похожа, зато, т.к. отдельные строки много раз повторяются, то если они просто, например, близки к среднему значению соответствующим им строкам во второй матрице (некоторая аналогия проекции второй матрицы на подпространство меньшей размерности), то расстояние может быть маленьким.

2.7 Планы по развитию данной идеи.

- Решить задачу “зашумления”. Возможно сначала определить средний спектр шума (1-2-3 секунды перед началом произнесения зашумленного сигнала), потом модифицировать как-то с его помощью ”чистый” сигнал (зашумить), и уже после сравнивать матрицы самосравнения.

- Не просто копировать строки или столбцы матрицы, а брать, например, среднее арифметическое между соседними – тогда, возможно, удастся избежать ситуации, описанной выше – т.е. когда при растяжении более маленькая и не очень похожая матрица дает меньшее расстояние.

3 Сравнение методов распознавания команды (сравнения с эталоном) в условиях шума

3.1 Необходимые сведения и основные определения

Звуковая волна распространяется в воздухе по известным законам физики. Как и любое колебание, звук характеризуется частотой и фазой, точнее, спектром колебания как зависимости амплитуды и фазы колебания от частоты (для данного момента времени).

Не вдаваясь в тонкости физиологического строения слухового аппарата, можно отметить следующую особенность процесса восприятия звукового сигнала человеком: звук воспринимается ухом не в амплитудно-временной, а в частотной области, - различные участки внутреннего уха отвечают за восприятие различных частот звука. При этом влияние фазы звуковых колебаний на процесс восприятия практически отсутствует.

Речи обычно сопутствуют иные источники звука, которые можно в совокупности назвать источниками шума. Источниками шума могут быть работающий компьютер, лампа дневного освещения, говорящий в коридоре человек и т.п.

Шум накладывается на речь по аддитивному закону (в амплитудно-временной области).

Глобальной характеристикой звука является ***громкость***, измеряемая в децибелах.

Речевой сигнал $s(t) : \mathbb{R}_+ \to \mathbb{R}$ можно рассматривать как функцию зависимости громкости (как величины со знаком) от времени.

Чтобы по дискретной последовательности значений амплитуды восстановить сигнал, используется так называемое ***дискретное преобразование Фурье (ДПФ)***, которое текущему окну (под ***окном звука*** понимается последовательность значений амплитуд звука некоторой определенной длины) длины N ставит в соответствие вектор коэффициентов разложения в ряд Фурье $c_0, ..., c_{N-1}$, причем $c_j = \overline{c_{N-j}}, \quad j = \overline{N/2+1, N-1}$, т.к. значения амплитуд действительны.

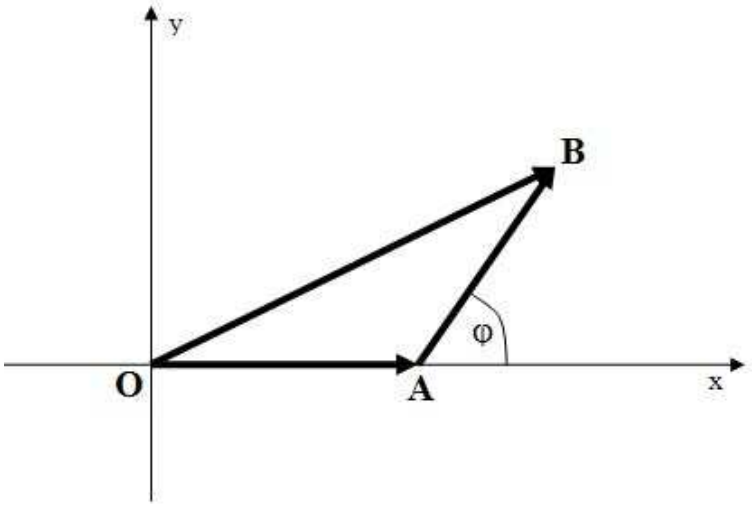
Назовем *"вектором Фурье"* элемент множества $\mathbb{R}_+^{N/2+1}$, составленный из модулей первых $N/2+1$ коэффициентов в разложении в ряд Фурье.

Порядковый номер коэффициента в векторе Фурье определяет частоту сигнала.

3.2 Геометрическая модель зашумления

Фактически каждый элемент из полученного вышеописанным образом вектора Фурье представляет собой коэффициент (амплитуду) при синусоиде, соответствующей некоторой частоте. Таким образом, после ДПФ двух сигналов и взятия модуля от коэффициентов мы имеем амплитуды синусоид сигналов на соответствующих частотах, причем фазы у синусоид каждого преобразования, вообще говоря, свои.

Значит, чтобы сложить два сигнала, имея их вектора Фурье, нам фактически нужно сложить (по каждой частоте отдельно) две синусоиды с разными фазами и амплитудами, что геометрически эквивалентно сложению двух векторов с длинами, равными амплитудам синусоид, и углом между ними, равным разности фаз:



В данном случае на рисунке изображено сложение двух векторов $\overrightarrow{OA}$ и $\overrightarrow{AB}$, соответствующих, например, амплитудам речи и шума, с разностью фаз между соответствующими векторами φ. Таким образом, в данной модели за зашумленный сигнал (например, зашумленная речь) мы принимаем $\overrightarrow{OB}$.

Назовем это *"геометрической моделью зашумления"*.

3.3 Распознавание звуковых сигналов. Методы распознавания в условиях шума

Часто, прежде всего для практических нужд, встает задача ***распознавания звуковых сигналов*** [1] - т.е. определения, какому типу (классу) принадлежит исследуемый звуковой сигнал. Деление на классы может быть совершенно различным, будь то классы речь-шум, голос определенного диктора или конкретное слово из какого-либо словаря.

Одним из основных методов распознавания является ***метод сравнения с эталонами*** (эталон - заранее записанный образец того или иного звукового сигнала). Он основан на том, что на основе выбранной метрики (расстояния) между звуковыми сигналами результатом распознавания звукового сигнала считается значение того эталона, который *ближе* (расстояние от сигнала до которого *меньше*) к рассматриваемому сигналу.

Но в распознавании в шуме есть свои трудности. В подавляющем большинстве случаев значение разности фаз между сигналом и шумом на любой отдельно взятой частоте неизвестно, т.к. нет возможности одновременно извлекать параметры для шума и речи отдельно. По аналогичным причинам неизвестно точное значение амплитуды и фазы шума на отдельной частоте, поэтому обычно сначала (например, на начальном отрезке времени зашумленного сигнала) производится настройка на шум и вычисляются какие-то его средние параметры.

Традиционно применяются следующие методы распознавания в шуме, основанные на методе сравнения с эталонами:

1. Использование эталонов, записанных в условиях отсутствия шума. Самый простой метод.

   Преимущества: простой математически и вычислительно.
   Недостатки: при увеличении уровня шума качество распознавания резко ухудшается.

2. ***"Метод спектрального вычитания"*** [2].

   Сначала вычитаем из амплитуды на каждой частоте оценку амплитуды шума, затем сравниваем полученный сигнал с эталоном.

   Преимущества: дает хорошие результаты при малом уровне шума.
   Недостатки: при большом уровне шума возможна ситуация, когда в результате применения метода отношение "сигнал/шум" уменьшается.

3. ***"Метод адаптивной фильтрации"*** (обучения в шуме) [3], [4].

   На каждом шаге производим настройку параметров адаптивного фильтра (например, с помощью метода наименьших квадратов), используя признак "речь-шум".

   Преимущества: хорошо работает для широкого класса квазистационарных шумов.
   Недостатки: плохой процент распознавания в случае сильно меняющихся или импульсных шумов; также зависит от правильности определения признака "речь-шум".

4. ***"Метод маскирования шума"*** [5].

   На каждой отдельной частоте путем применения введенной выше "геометрической модели зашумления" зашумляем эталон с шумом со случайной фазой. Затем сравниваем распознаваемый сигнал с "зашумленным" эталоном.

3.4 Цели работы

A. Найти распределение зашумленного сигнала $\overrightarrow{OB}$.

B. Получить общую формулу для вычисления вероятности правильного распознавания методом сравнения с эталонами.

C. Получить аналитические формулы для попарного сравнения методов 1), 2) и 3).

D. Более детально сравнить методы 1) и 2).

3.5 Сделанные допущения

- Без ограничения общности будем рассматривать сигнал на какой-нибудь фиксированной частоте. Все выкладки очевидным образом распространяются на все частоты.

- Полагаем, что разность фаз для пары сигнал-шум на этой частоте - это случайная величина φ, равномерно распределенная на отрезке $[0, 2\pi]$.

- Для простоты положим, что амплитуды незашумленного сигнала и шума - две случайные равномерно распределенные величины на отрезках $[a, b]$ и $[c, d]$ соответственно: $s \in [a, b]$ - сигнал и $n \in [c, d]$ - шум , значения которых принадлежат $\mathbb{R}_+$ (т.е. $a \geq 0$ и $c \geq 0$).

- В среднем сигнал "больше" шума - т.е. $\frac{a+b}{2} \geq \frac{c+d}{2}$.

- Случайные величины s, n, φ - независимы.

- Без ограничения общности в качестве эталонов будем рассматривать 2 неотрицательных числа $0 \leq e_1 < e_2$.

- В качестве метрики между двумя величинами x и $y \in \mathbb{R}$ возьмем модуль разности между ними, т.е.
$$\rho(x, y) = |x - y|.$$

- Сигнала s равномерно распределен на отрезке $[e_1, e_2]$ (т.е. $a = e_1, b = e_2$).

- Также полагаем, что ни при каком значении разности фаз сигнал-шум первый эталон не станет дальше от нуля, чем второй (т.е. человек способен будет еще их различить). Это порождает условие на максимальную амплитуду шума, которая не может быть больше половины расстояния между эталонами, т.е.
$$n \leq \frac{L}{2}, \text{ где } L = e_2 - e_1.$$

- Для решения задачи D) предположим, что расстояние между эталонами меньше расстояния от первого эталона до нуля, т.е. $L < e_1$.

3.6 Постановка задач

В рамках вышеперечисленных допущений необходимо решить следующие задачи:

A. Пусть $s \oplus n$ - зашумленный сигнал (см. "геометрическую модель зашумления" - там фигурировал вектор $\overrightarrow{OB}$). Требуется найти распределение $p(s \oplus n)$.

B. Пусть α - знак неравенства ($<$ или $>$). Далее, пусть имеем изначально неравенство

$$\rho(s, e_1) \ \alpha \ \rho(s, e_2).$$

Требуется найти вероятность правильного распознавания, т.е. вероятность того, что после преобразования (которого, вообще говоря, может и не быть - например, метод 1) сигнала и эталонов $s \to s', e_1 \to e_1', e_2 \to e_2'$ сигнал не станет ближе к другому, т.е.

$$\mathbf{P}_0 = \mathbf{P}(\rho(s, e_1') \ \alpha \ \rho(s, e_2') \bigcup \rho(s, e_1) \ \alpha \ \rho(s, e_2))$$

C. Пусть $\mathbf{P}_0^i$ - вероятность правильного распознавания для метода с номером i. Получить аналитические формулы для $\mathbf{P}_0^1, \mathbf{P}_0^2, \mathbf{P}_0^3$ (и, соответственно, их попарных разностей $\mathbf{P}_0^1 - \mathbf{P}_0^2, \mathbf{P}_0^1 - \mathbf{P}_0^3, \mathbf{P}_0^2 - \mathbf{P}_0^3$).

D. Для некоторых значений s и n выяснить знак α в неравенстве $\mathbf{P}_0^1 \ \alpha \ \mathbf{P}_0^2$.

3.7 Вычисление плотности распределения $s \oplus n$

Теорема 1

Плотность распределения $s \oplus n$ вычисляется по формуле

$$p(s \oplus n) = \frac{1}{\pi(b-a)(d-c)} \int_a^b \int_c^d \frac{s \oplus n}{sn\sqrt{1 - (\frac{s^2+n^2-s\oplus n^2}{2sn})^2}} dnds.$$

▶ Верна следующая [6]
Теорема (о замене переменных под знаком тройного интеграла):
Пусть $\varphi, \psi, \chi : \mathbb{R}^3 \to \mathbb{R}$ - обратимые дифференцируемые отображения. Тогда верно следующее равенство:

$$\iiint_{\Omega'} g(x_1', x_2', x_3') dx_1' dx_2' dx_3' = \iiint_{\Omega} g(\varphi(\overline{x}), \psi(\overline{x}), \chi(\overline{x})) \times |D(\overline{x})| dx_1 dx_2 dx_3,$$

где $\overline{x} = (x_1, x_2, x_3), x_1' = \varphi(\overline{x}), x_2' = \psi(\overline{x}), x_3' = \chi(\overline{x})$, а $D(\overline{x})$ - якобиан отображения $F = (\varphi, \psi, \chi) : \mathbb{R}^3 \to \mathbb{R}^3$.

Пусть $f : \mathbb{R}^3 \to \mathbb{R}^3$ - обратимое дифференцируемое отображение:

$$a = f(x, y, z)_1, b = f(x, y, z)_2, c = f(x, y, z)_3;$$

$$x = f^{-1}(a, b, c)_1, y = f^{-1}(a, b, c)_2, z = f^{-1}(a, b, c)_3.$$

Пусть (x, y, z) - независимые случайные величины, имеющие плотности p_x, p_y и p_z соответственно. Т.к. справедливо [7]
Утверждение Плотность совместного распределения независимых случайных величин - произведение их плотностей,
то совместная плотность распределения $p(a, b, c)$ вычисляется по формуле

$$p(x, y, z) = p_x(x) p_y(y) p_z(z) =$$

$$= p_x(f^{-1}(a,b,c)_1)p_y(f^{-1}(a,b,c)_2)p_z(f^{-1}(a,b,c)_3)J_{f^{-1}} = p(a,b,c),$$

где $J_{f^{-1}}$ - якобиан функции f^{-1}

У нас есть независимые равномерно распределенные случайные величины s, n, φ. Хотим узнать плотность распределения случайной величины $s \oplus n$, которая по формуле косинусов равна (см. рисунок к разделу "Геометрическая модель зашумления"):

$$s \oplus n = \sqrt{s^2 + n^2 - 2sn\cos(\varphi)}.$$

Т.к. $s \oplus n$ - непрерывно зависящая и монотонно возрастающая функция от φ на отрезке $[0, \pi]$, то максимум и минимум эта функция принимает на концах отрезка - т.е. в точках $\varphi = 0$ и $\varphi = \pi$, и ее значения принадлежат отрезку $[s \oplus n(\varphi = 0), s \oplus n(\varphi = \pi)] = [|s - n|, s + n]$.

Следовательно, в нашем случае можно взять в качестве функции f следующее отображение:

$$f : s \to s, n \to n, \varphi \to s \oplus n = \sqrt{s^2 + n^2 - 2sn\cos(\varphi)}.$$

Таким образом, обратная функция f^{-1}:

$$f^{-1} : s \to s, n \to n, s \oplus n \to \varphi = \arccos\left(\frac{s^2 + n^2 - s \oplus n^2}{2sn}\right).$$

Т.к. нет никакой разницы в вышеупомянутом рисунке, какому отрезку принадлежит φ - $[0, \pi]$ или $[0, 2\pi]$ (правда, это справедливо в том случае, если φ равномерно распределена на соответствующем отрезке) - картинки симметричны относительно оси Ox, а арккосинус - обратимое дифференцируемое отображение при $\varphi \in [0, \pi]$, то естественно положить разность фаз равномерно распределенной на отрезке $[0, \pi]$.

Величина $J_{f^{-1}}$ определяется значением

$$\frac{\partial\varphi}{\partial(s \oplus n)} = \frac{s \oplus n}{sn\sqrt{1 - (\frac{s^2+n^2-s\oplus n^2}{2sn})^2}},$$

т.к. сам якобиан имеет вид

$$J_{f^{-1}} = \begin{Vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \dots & \dots & \frac{\partial\varphi}{\partial(s\oplus n)} \end{Vmatrix}$$

Затем получившуюся трехмерную плотность $p(s, n, s \oplus n)$ интегрируем по первой (от a до b) и второй (от c до d) координатам. С учетом того, что $p_s = \frac{1}{b-a}, p_n = \frac{1}{d-c}$ и $p_\varphi = \frac{1}{\pi}$, в итоге получим плотность распределения случайной величины $s \oplus n$:

$$p(s \oplus n) = \frac{1}{\pi(b-a)(d-c)} \int_a^b \int_c^d \frac{s \oplus n}{sn\sqrt{1 - (\frac{s^2+n^2-s\oplus n^2}{2sn})^2}} dnds. \qquad \blacksquare$$

3.8 Вычисление вероятности правильного распознавания методом сравнения с эталоном в условиях шума

Пусть имеется два эталона p и $q, p < q$, и сигнал r. Преобразованные (зашумленные/расшумленные/оставленные без изменений) эталоны и сигнал - p', q' и r' соответственно.

Утверждение

Вероятность правильного распознавания вычисляется по формуле

$$\mathbf{P}_0 = \mathbf{P}(r < \frac{p+q}{2}, r' < \frac{p'+q'}{2}) + \mathbf{P}(r \geq \frac{p+q}{2}, r' \geq \frac{p'+q'}{2}). \qquad (0)$$

► Очевидно, что близость сигнала r к соответствующему эталону будет определяться его положением относительно середины отрезка $[p, q]$, т.е. если $r < \frac{p+q}{2}$, то $\rho(r, p) < \rho(r, q)$, в противном случае $\rho(r, p) \geq \rho(r, q)$. Совершенно аналогично: если $r' < \frac{p'+q'}{2}$, то $\rho(r', p') < \rho(r', q')$, иначе $\rho(r', p') \geq \rho(r', q')$.

Вероятность правильного распознавания - это вероятность того, что после преобразования сигнал не стал ближе к другому эталону. Т.к. сигнал может быть ближе только к одному из двух эталонов, то вероятность правильного распознавания в условиях шума складывается из двух вероятностей: вероятности того, что чистый сигнал и преобразованный сигналы были ближе к первому эталону (исходному и преобразованному соответственно) и вероятности того, что они были ближе ко второму эталону. Формальная запись:

$$\mathbf{P}_0 = \mathbf{P}(r < \frac{p+q}{2}, r' < \frac{p'+q'}{2}) + \mathbf{P}(r \geq \frac{p+q}{2}, r' \geq \frac{p'+q'}{2}). \qquad \blacksquare$$

3.9 Вывод аналитических формул для расчета вероятности правильного распознавания для методов 1), 2) и 3)

Рассчитаем величину $\mathbf{P}_0^1$.

Имеем:

$$p = p' = e_1, q = q' = e_2, r = s, r' = s \oplus n.$$

Значит, по формуле (0) получим:

$$\mathbf{P}_0^1 = \mathbf{P}(s < \frac{e_1+e_2}{2}, s \oplus n < \frac{e_1+e_2}{2}) + \mathbf{P}(s \geq \frac{e_1+e_2}{2}, s \oplus n \geq \frac{e_1+e_2}{2}) =$$

$$= \frac{1}{\pi(b-a)(d-c)} \times \left(\int\limits_{|s-n|}^{\frac{e_1+e_2}{2}} \int\limits_{a}^{\frac{e_1+e_2}{2}} \int\limits_{c}^{d} \frac{s \oplus n}{sn\sqrt{1 - (\frac{s^2+n^2-s\oplus n^2}{2sn})^2}} dndsd(s \oplus n) + \right.$$

$$\left. + \int\limits_{\frac{e_1+e_2}{2}}^{s+n} \int\limits_{\frac{e_1+e_2}{2}}^{b} \int\limits_{c}^{d} \frac{s \oplus n}{sn\sqrt{1 - (\frac{s^2+n^2-s\oplus n^2}{2sn})^2}} dndsd(s \oplus n) \right). \qquad (1)$$

Следовательно, верна

Лемма 1. *Вероятность правильного распознавания методом 1) выражается формулой (1).*

Вывод формул для случаев 2) и 3) совершенно аналогичен (меняются только пределы интегрирования и увеличивается количество слагаемых).

3.10 Сравнение методов 1) и 2)

Воспользуемся сделанными выше допущениями, что $a = e_1, b = e_2$ и $n \leq \frac{L}{2}$, где $L = e_2 - e_1$. Теперь обозначим за a - середину отрезка $[e_1, e_2]$, т.е. $a = \frac{e_1+e_2}{2}$. За x обозначим расстояние между сигналом s и серединой отрезка $[e_1, e_2]$, т.е. $x = |s - a|$.

Теорема 2. *При $x \geq 2n$ оба методы не работают.*
При $\frac{3}{4} \leq x < 2n$ лучше работает метод 1).

► Преобразуем выражение $\mathbf{P}_0^1 - \mathbf{P}_0^2 =$

$$= \mathbf{P}(s < a, s \oplus n < a) + \mathbf{P}(s \geq a, s \oplus n \geq a) - \mathbf{P}(s < a, s \oplus n < n+a) - \mathbf{P}(s \geq a, s \oplus n \geq n+a) =$$

$$= \mathbf{P}(s < a, s \oplus n < a) - \mathbf{P}(s < a, s \oplus n < n+a) + \mathbf{P}(s \geq a, s \oplus n \geq a) - \mathbf{P}(s \geq a, s \oplus n \geq n+a) =$$

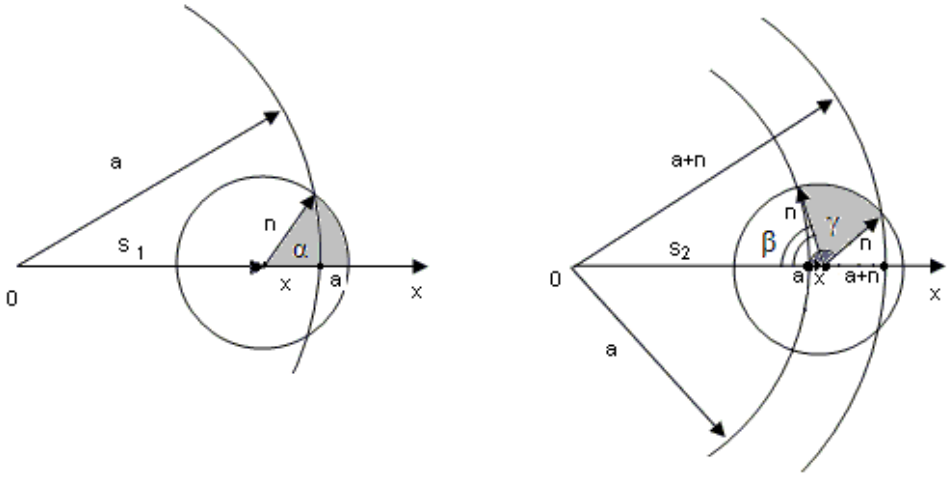
$$= \mathbf{P}(s \geq a, s \oplus n \in [a, n+a]) - \mathbf{P}(s < a, s \oplus n \in [a, n+a]). \quad (4)$$

Далее, рассматриваем, так же как и в Теореме 1, только углы $\varphi \in [0, \pi]$ - т.е. верхнюю полуплоскость.

Каким образом будем оценивать формулу (4). План доказательства теоремы следующий: берем всевозможные $x \in [0, L/2]$, для каждого конкретного x рассматриваем сигналы $s_1 = a - x$ и $s_2 = a + x$, берем конкретное значение $n \in [0, L/2]$, для них вычисляем величину $s \oplus n_1$ и $s \oplus n_2$, и смотрим, "больше" ли (под мерой понимаем лебегову меру отрезка) множество углов, задающих разность фаз, удовлетворяющих условию $s \oplus n_2 \in [\frac{e_1+e_2}{2}, n + \frac{e_1+e_2}{2}]$, множества углов, удовлетворяющих $s \oplus n_1 \in [\frac{e_1+e_2}{2}, n + \frac{e_1+e_2}{2}]$.

В том случае, когда "больше", считаем, что выполняется условие $У$. Если "меньше", то считаем, условие $У$ не выполняется. Если мера множества допустимых x (и, соответственно, s) и n (в данном случае мера Лебега на плоскости), при которых условие $У$ выполняется, больше меры множества, на которых условие $У$ не выполняется (на самом деле, есть область допустимых x и n, в которой множество углов, удовлетворяющих соответствующим условиям, пусто), то условие теоремы выполнено.

Ясно, что $s \oplus n \in [a, a+n]$ в том и только том случае, если конец вектора $\overrightarrow{s} + \overrightarrow{n}$ лежит внутри кольца, образованного двумя окружностями с радиусами a и $a+n$. Обозначим за α угол, образованный вектором $\overrightarrow{n}$, который находится в крайнем допустимом положении для сигнала $a - x$, и положительным направлением оси, а за β, γ - углы, образованные вектором $\overrightarrow{n}$, который находится в крайнем допустимом положении для сигнала $a + x$, и отрицательным направлением оси Ox - см. рисунки:



Серым цветом закрашена область допустимых углов, которая определяется величинами α и $\gamma - \beta$ соответственно.

Ясно, что при $x \geq 2n$, множество допустимых углов в обоих случаях пустое, т.к.

$$s \oplus n_1 \leq s_1 + n \leq a - 2n + n = a - n \leq a,$$

$$s \oplus n_2 \geq s_2 - n \geq a + 2n - n = a + n.$$

В случае $n \leq x \leq 2n$ множество допустимых углов для $s \oplus n_2$ ненулевой меры, а для $s \oplus n_1$ - пусто, т.к.

$$s \oplus n_1 \leq s_1 + n \leq a - n + n = a.$$

Таким образом, для $n \leq x \leq 2n$ выполняется условие $У$. Рассмотрим случай $\frac{3n}{4} \leq x \leq n$.

При $x = kn, 0 < k < 1$ по теореме косинусов имеем

$$\cos\alpha = k - \frac{1-k^2}{2(z-k)},\ \cos\beta = k + \frac{1-k^2}{2(z+k)},\ \cos\gamma = k - 1 + \frac{2k-k^2}{2(z+k)},$$

где $z = \dfrac{a}{n} = \dfrac{e_1+e_2}{2n} \geq \dfrac{e_1+e_2}{L} = \dfrac{e_1+e_2}{e_2-e_1} = 1 + \dfrac{2e_1}{e_2-e_1} = 1 + \dfrac{2}{\frac{e_2}{e_1}-1} \geq 1 + \dfrac{2}{2-1} = 3,$

т.е. $z \geq 3$. Следовательно,

$$k - \frac{1-k^2}{2(3-k)} < \cos\alpha < k,\ k < \cos\beta < k + \frac{1-k^2}{2(3+k)},\ k-1 < \cos\gamma < k - 1 + \frac{2k-k^2}{2(3+k)}.$$

Исходя из этих оценок, получаем

$$\sin\beta = \sqrt{1-\cos^2\beta} < \sqrt{1-k^2},\ \sin\gamma \leq 1.$$

Множество допустимых углов для $s \oplus n_1$ определяется величиной α, а для $s \oplus n_2$ - величиной $\gamma - \beta$. Для того, чтобы выполнялось условие $У$, необходимо, чтобы $\alpha < \gamma - \beta$, т.е.

$$\cos\alpha > \cos(\gamma-\beta) = \cos\gamma\cos\beta + \sin\gamma\sin\beta. \tag{5}$$

Рассмотрим случай, когда $k - 1 + \frac{2k-k^2}{2(3+k)} < 0,$ т.е. $k < \sqrt{15} - 3$. В этом случае условие (5) будет следовать из неравенства

$$k(k - 1 + \frac{2k-k^2}{2(3+k)}) + \sqrt{1-k^2} < k - \frac{1-k^2}{2(3-k)}. \tag{6}$$

Продифференцируем левую часть неравенства (6) по k. Получим

$$\frac{1}{2(3+k)^2\sqrt{1-k^2}}(2k^3(\sqrt{1-k^2}-1) + 3k^2(5\sqrt{1-k^2}-4) + \sqrt{1-k^2}(36k-18) - 18k)$$

Ясно, что при $k \in [\frac{3}{4}, 1]$:

$$2k^3(\sqrt{1-k^2}-1) < 0,$$

$$3k^2(5\sqrt{1-k^2}-4) < 3k^2(5\sqrt{1-\left(\frac{3}{4}\right)^2}-4) = 3k^2\frac{5\sqrt{7}-16}{4} < 0,$$

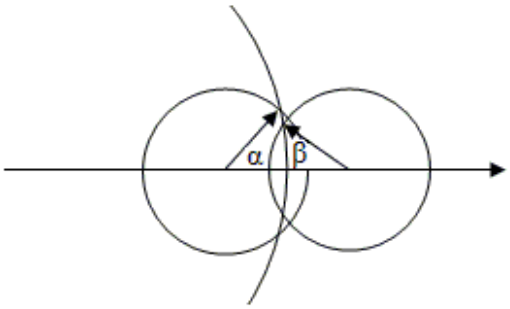
$$\sqrt{1-k^2}(2k-1) - k < \sqrt{1-\left(\frac{3}{4}\right)^2}(2k-1) - k = \frac{\sqrt{7}}{4}(2k-1) - k <$$

$$< \frac{3}{4}(2k-1) - k = \frac{1}{2}k - \frac{3}{4} < \frac{1}{2} - \frac{3}{4} < 0.$$

Следовательно, производная меньше нуля, и левая часть неравенства (6) убывает. Аналогично показывается, что правая часть возрастает. Т.к. при $k = \frac{3}{4}$ это неравенство верно, оно будет верно и при $\frac{3}{4} \leq k \leq \sqrt{15} - 3 = k_1$.

Аналогичным образом можно было бы поступить и при $\sqrt{15} - 3 \leq k \leq 1$ (правда, неравенство (6) примет немного иной вид из-за другой оценки произведения косинусов), но в таком случае вычисление производных и исследование функций очень сложно. Поступим иначе.

При $x = n$ $(k = 1)$ имеем $\alpha = \beta = 0, \gamma - \beta = \gamma = \gamma_0$. Далее, очевидно, что при уменьшении x γ и β увеличиваются; пусть $\gamma = \gamma_0 + \psi$. Следовательно, $\gamma - \beta = \gamma_0 + \psi - \beta$. Из геометрических соображений (см.рисунок) всегда $\alpha > \beta$, т.е. $-\beta > -\alpha$. Значит, $\gamma - \beta > \gamma_0 + \psi - \alpha$.



Таким образом, если мы докажем, что $\gamma_0 + \psi - \alpha > \alpha$, т.е. $\alpha < \frac{1}{2}(\gamma_0 + \psi)$, а это неравенство следует из $\alpha < \frac{1}{2}\gamma_0$, то получим, что $\gamma - \beta > \alpha$, т.е. условие $У$ выполняется.

Рассмотрим, при каких k выполняется неравенство

$$\cos\alpha > \cos\frac{\gamma_0}{2} = \sqrt{\frac{1+\cos\gamma_0}{2}} \tag{7}$$

Подставляя $k = 1$ в вышеприведенные формулы для косинусов, получаем, что

$$\cos\gamma_0 = \frac{1}{2(z+1)} < \frac{1}{8}.$$

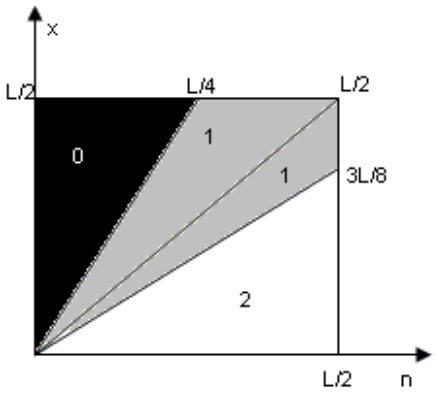
Опять же используя те же формулы, получаем, что для (7) нужно доказать неравенство $k - \frac{1-k^2}{2(3-k)} > \frac{3}{4}$, которое равносильно квадратичному неравенству

$$2k^2 - 15k + 11 < 0$$

т.е. $k > \frac{15-\sqrt{137}}{4} = k_2$. Т.к. $k_2 < k_1$, то мы доказали, что при $\frac{3}{4} \le k \le 1$ условие $У$ выполняется, а вместе с тем и условие теоремы ■

Замечание. *Мера на множестве допустимых значений (s, n), где метод 1) работает лучше, больше соответствующей меры для метода 2).*

►



Мера на квадрате (s, n), $0 \le s \le L/2$, $0 \le n \le L/2$ (см. рисунок: 0 - где мера допустимых параметров равна нулю, т.е. оба метода не работают, 1 - где выполняется условие $У$, т.е. метод 1) работает лучше, и 2 - где лучше работает второй метод) где условие $У$ выполняется, есть

$$\mu_1 > \frac{1}{2}\frac{L}{4}\frac{L}{2} + \frac{1}{2}\frac{L}{8}\frac{L}{2} = \frac{1}{4}\frac{3L^2}{8}.$$

Условие $У$ не выполняется на мере

$$\mu_2 < \frac{1}{2}\frac{3L}{8}\frac{L}{2} = \frac{1}{4}\frac{3L^2}{8}.$$

Значит, $\mu_1 > \mu_2$. ■

4 Введение в Марковские случайные поля

4.1 Необходимые сведения и основные определения

Пусть $A = \{1, 2, ..., a\}$ и $B = \{1, 2, ...b\}, a, b < \infty$ - два конечных множества. Пусть $S = \{1, 2, ..., N\}$ - множество индексов.

Пусть $R = \{R_i, i \in S\}$ - многомерная случайная величина, т.ч. каждая компонента R_j, являющаяся одномерной случайной величиной, принимает значение r_j и определена в своем вероятностном пространстве. Для простоты будем считать, что $\forall j$ R_j дискретны, определены на одном вероятностном пространстве и множество значений - конечно.

Также для удобства представления можно представлять, что множество индексов S задает множество точек на плоскости. Соответственно, рассматриваем реализацию многомерной случайной величины R в этих точках. Введенная т.о. случайная величина R называется **случайным полем (Random Field)**.

Совместное событие $(R_1 = r_1, ..., R_N = r_N)$, или кратко $R = r$, где $r = \{r_1, ..., r_N\}$, назовем **конфигурацией** R.

Пусть X - случайное поле со значениями на множестве A, т.е. $\forall i$ $X_i \in A$. Если x - какая-то конкретная конфигурация X, то χ - множество всех возможных конфигураций:

$$\chi = \{x = (x_1, ..., x_N) |\ x_i \in A\ \forall i \in S\}.$$

Введем понятие **системы соседства**: это множество $\partial = \{\partial i, i \in S\}$, где ∂i - множество элементов из S, называемое **шаблоном соседства для элемента i**, т.ч.:

$$\begin{cases} i \notin \partial i, \\ i \in \partial j \Leftrightarrow j \in \partial i. \end{cases}$$

Определение. Случайное поле X будем называть **марковским случайным полем (Markov Random Field, MRF)** в соответствии с системой соседства ∂ тогда и только тогда, когда $\forall i$:

$$\begin{cases} P(X = x) > 0\ \forall x \in \chi, \\ P(X_i = x_i |\ X_j = x_j,\ j \in S \backslash \{i\}) = P(X_i = x_i |\ X_j = x_j,\ j \in \partial i). \end{cases}$$

Определение. Клика c - множество элементов из S, т.ч. $\forall s, r \in c \Rightarrow r \in \partial s$. Заметим, что любое подмножество клики - также клика.

Пусть x_c - набор значений X_i, где $i \in c$. Введем *потенциальную функцию* $V_c(x_c)$ как любую функцию от x_c.

Определение. Дискретное распределение называется **распределением Гиббса**, если

$$\mathbf{P}(X = x) = \frac{1}{Z} \exp\left(-\sum_{c \in C} V_c(x_c)\right), \tag{1}$$

где C - множество всех клик, а Z - нормирующая константа, т.ч.:

$$Z = \sum_{x \in \chi} \exp\left(-\sum_{c \in C} V_c(x_c)\right). \tag{2}$$

Наиболее важной теоремой, связывающей марковские случайные поля и распределение Гиббса, является следующая

Теорема. *(Hammersley-Clifford).* X - марковское случайное поле $\Leftrightarrow$ $\mathbf{P}(X = x)$ - распределение Гиббса.

Т.о., мы имеем возможность вычислять вероятность конфигурации для любого марковского случайного поля по формулам (1), (2).

Определение. Скрытое марковское случайное поле (Hidden Markov Random Field, HMRF) - пара случайный полей (X, Y), т.ч. выполняются следующие условия:

1. $X = \{X_i, i \in S\}$ - так называемое "скрытое" (или, другими словами, ненаблюдаемое) марковское поле со значениями в A.

2. $Y = \{Y_i, i \in S\}$ - наблюдаемое (вовсе не обязательно марковское) случайное поле со значениями в B. Важно, что $\forall i \in S, \forall d \in A$ известны условные распределения $\mathbf{P}(Y_i|\ X_j = d), \forall j \in S$.

3. Для любой конфигурации $x \in \chi$ случайные величины Y_i условно независимы, т.е.

$$\mathbf{P}(Y|\ X = x) = \prod_{i \in S} \mathbf{P}(Y_i|\ X_i = x_i).$$

Т.о., получим

$$\mathbf{P}(Y = y, X = x) = \mathbf{P}(Y = y|\ X = x)\mathbf{P}(X = x) = \mathbf{P}(X = x) \prod_{i \in S} \mathbf{P}(Y_i = y_i|\ X_i = x_i).$$

Совместное распределение (X_i, Y_i) с шаблоном соседства ∂i определяется как

$$\mathbf{P}(X_i = x_i, Y_i = y_i|\ X_j = x_j, j \in \partial i) = \mathbf{P}(Y_i = y_i|\ X_i = x_i)\mathbf{P}(X_i = x_i|\ X_j = x_j, j \in \partial i),$$

а распределение Y_i с шаблоном соседства ∂i определяется как

$$\mathbf{P}(Y_i = y_i|\ X_j = x_j, j \in \partial i) = \sum_{d \in A} \mathbf{P}(Y_i = y_i, X_i = d|\ X_j = x_j, j \in \partial i) =$$

$$= \sum_{d \in A} \mathbf{P}(Y_i = y_i|\ X_i = d)\mathbf{P}(X_i = d|\ X_j = x_j, j \in \partial i).$$

4.2 Взаимосвязи между марковскими цепями\ скрытыми марковскими моделями и случайными марковскими полями\ скрытыми случайными марковскими полями

Рассмотрим марковскую цепь с состояниями $x_0, x_1, ..., x_n$, функционирующую в дискретном времени. Построим многомерную случайную величину $X = \{X_i, i = \overline{0..n}\}$ по этой марковской цепи следующим образом: $\mathbf{P}(X_i = x_i)$ - это вероятность того, что в момент времени i мы находились в состоянии x_i. Назовем X *порожденной* случайной величиной.

Утверждение 1. Порожденная случайная величина - марковское случайное поле.

Доказательство.

Зададим шаблон соседства для i : $\partial i = \{i - 1, i + 1\}$. Любая клика c будет иметь вид $c = \{i - 1, i\}$. Тогда вероятность реализации порожденной случайной величины X (т.е. прохода по марковской цепи по соответствующим состояниям) будет иметь вид

$$\mathbf{P}(X = x) = \mathbf{P}(X_0 = x_0) \prod_{i=1}^{n} \mathbf{P}(X_i = x_i|\ X_{i-1} = x_{i-1}) =$$

$$= \mathbf{P}(X_0 = x_0) \exp\left(\sum_{i=1}^{n} \log \mathbf{P}(X_i = x_i|\ X_{i-1} = x_{i-1}) \right),$$

т.е. будет иметь вид распределения Гиббса с потенциальными функциями

$$V(x_i, x_{i-1}) = -\log \mathbf{P}(X_i = x_i |\ X_{i-1} = x_{i-1}).$$

Конец доказательства.

Утверждение 2. Любое марковское случайное поле с линейной структурой - порожденная случайная величина.

Доказательство.

Имеем шаблон соседства для i : $\partial i = \{i-1, i+1\}$. Любая клика c имеет вид $c = \{i-1, i\}$. Вероятность конфигурации x имеет форму распределения Гиббса:

$$\mathbf{P}(X = x) = \mathbf{P}(X_0 = x_0) \exp\left(-\sum_{i=1}^{n} V(x_i, x_{i-1})\right).$$

Исходя из определения случайного марковского поля, можно записать вероятность нахождения в данном состоянии в зависимости от нахождений во всех иных состояниях:

$$\mathbf{P}(X_i = x_i |\ X_j = x_j, j \neq i) = \mathbf{P}(X_i = x_i |\ X_{i-1} = x_{i-1}, X_{i+1} = x_{i+1}).$$

Зависимость от состояний с меньшими номерами выражается формулой

$$\mathbf{P}(X_i = x_i |\ X_j = x_j, j < i) = \mathbf{P}(X_i = x_i |\ X_{i-1} = x_{i-1}).$$

Т.о., зависимость от предыстории и от всей последовательности состояний у линейного марковского случайного поля такие же, как и у порожденной от марковской цепи случайной величины.

Конец доказательства.

Заметим, что вычисление вероятностей перехода по марковскому случайному полю не является тривиальной задачей и в данной работе не рассматривается.

Теперь рассмотрим скрытую марковскую модель и ее связь с марковским случайным полем и скрытым марковским случайным полем.

Рассмотрим скрытую марковскую модель, функционирующую в дискретном времени. Пусть при некотором ее функционировании при проходе через состояния $x_0, x_1, ..., x_n$ на выход подавались буквы $y_0, y_1, ..., y_n$ соответственно. Рассмотрим многомерную случайную величину

$$Z = (Z_1, Z_2, ..., Z_{2n+2}) = (X_0, Y_0, X_1, Y_1, ..., X_n, Y_n),$$

причем вероятность $\mathbf{P}(X_i = x_i |\ X_{i-1} = x_{i-1})$ - это вероятность перехода $a(x_{i-1}, x_i)$ из состояния x_{i-1} в состояние x_i, а вероятность $\mathbf{P}(Y_i = y_i |\ X_i = x_i)$ - это вероятность выдачи $b(y_i, x_i)$ буквы y_i в состоянии x_i. Назовем такую многомерную случайную величину *скрытой порожденной* случайной величиной.

Утверждение 3. 1. Скрытая порожденная случайная вличина - это случайное марковское поле специального вида: причем условимся называть случайные величины X_i скрытыми, а Y_i - наблюдаемыми.

2. Скрытая порожденная случайная вличина - это скрытое случайное марковское поле линейной структуры.

Доказательство.

1. Для скрытой порожденной случайной величины клики, содержащие номер $2k+1$: $c_1 = \{2k-1, 2k+1\}, c_2 = \{2k+1, 2k+2\}, c_3 = \{2k+1, 2k+3\}$ (по сути, клики c_1 и c_3

идентичны); клика, содержащая номер $2k+2$ - это c_2. Вероятность реализации скрытой порожденной случайной величины Z есть

$$\mathbf{P}(Z=z)=\mathbf{P}(X=x,Y=y)=\mathbf{P}(X_0=x_0)\mathbf{P}(Y_0=y_0|\ X_0=x_0)\times$$

$$\times\prod_{i=1}^{n}\mathbf{P}(X_i=x_i|\ X_{i-1}=x_{i-1})\mathbf{P}(Y_i=y_i|\ X_i=x_i)=$$

$$=\mathbf{P}(X_0=x_0)\exp\left(\log\mathbf{P}(Y_0=y_0|\ X_0=x_0)+\right.$$

$$\left.+\sum_{i=1}^{n}\left(\log\mathbf{P}(X_i=x_i|\ X_{i-1}=x_{i-1})+\log\mathbf{P}(Y_i=y_i|\ X_i=x_i)\right)\right)=$$

$$=\mathbf{P}(X_0=x_0)\exp\left(-\sum_{i=1}^{n}V_{c_1}(x_i,x_{i-1})-\sum_{i=0}^{n}V_{c_2}(y_i,x_i)\right),$$

т.е. вероятность реализации задается распределением Гиббса с потенциальными функциями

$$V_{c_1}(x_i,x_{i-1})=-\log\mathbf{P}(X_i=x_i|\ X_{i-1}=x_{i-1})=-\log a(x_{i-1},x_i)$$

и

$$V_{c_2}(y_i,x_i)=-\log\mathbf{P}(Y_i=y_i|\ X_i=x_i)=-\log b(y_i,x_i).$$

Т.о., по теореме Хаммерсли-Клиффорда скрытая порожденная случайная величина является случайным марковским полем.

2. Для доказательства утверждения установим, что все три пункта в определении скрытого случайного марковского поля выполняются.

Доказательство того, что X - случайное марковское поле с потенциальными функциями вида $V_{c_1}(x_i,x_{i-1})=-\log\mathbf{P}(X_i=x_i|\ X_{i-1}=x_{i-1})=-\log a(x_{i-1},x_i)$ проходит совершенно аналогично пункту 1 данного утверждения. Условные вероятности $\mathbf{P}(Y_i=y_i|\ X_i=x_i)$ также даны и равны $b(y_i,x_i)$. И, наконец, по определению скрытой марковской модели Y_i условно независимы, т.к. наблюдаемая буква зависит только от скрытого под ней состояния:

$$\mathbf{P}(Y=y|\ X=x)=\prod_{i=0}^{n}\mathbf{P}(Y_i=y_i|\ X_i=x_i).$$

Т.о., скрытая порожденная случайная величина является скрытым марковским случайным полем линейной структуры.

Конец доказательства.

Утверждение 4. Любое скрытое марковское случайное поле с линейной структурой задает некоторую скрытую порожденную случайную величину.

Доказательство.

Имеется скрытое марковское случайное поле с наблюдаемым случайным полем $Y=\{Y_0,Y_1,...,Y_n\}$ и скрытым марковским случайным полем $X=\{X_0,X_1,...,X_n\}$.
Рассмотрим многомерную случайную величину

$$Z=(Z_1,Z_2,...,Z_{2n+2})=(X_0,Y_0,X_1,Y_1,...,X_n,Y_n).$$

Клики, содержащие номер $2k+1$: $c_1=\{2k-1,2k+1\},c_2=\{2k+1,2k+3\}$ (по существу эти клики идентичны).

Исходя из определения скрытого случайного марковского поля, можно записать вероятность нахождения в данном состоянии в зависимости от нахождений во всех иных состояниях:

$$\mathbf{P}(X_i = x_i|\ X_j = x_j, j \neq i) = \mathbf{P}(X_i = x_i|\ X_{i-1} = x_{i-1}, X_{i+1} = x_{i+1}).$$

Зависимость от состояний с меньшими номерами выражается формулой

$$\mathbf{P}(X_i = x_i|\ X_j = x_j, j < i) = \mathbf{P}(X_i = x_i|\ X_{i-1} = x_{i-1}).$$

Т.о., случайная величина $X = \{X_0, X_1, ..., X_n\}$ - порожденная случайная величина (как и должно быть, т.к. она должна быть порождена от ненаблюдаемого марковского процесса). А случайная величина Z является скрытой порожденной случайной величиной, т.к. по определению скрытого случайного марковского поля для любой конфигурации $x \in \chi$ случайные величины Y_i условно независимы:

$$\mathbf{P}(Y|\ X = x) = \prod_{i \in S} \mathbf{P}(Y_i|\ X_i = x_i),$$

т.е. наблюдаемая буква зависит только от скрытого под ней состояния.

Конец доказательства.