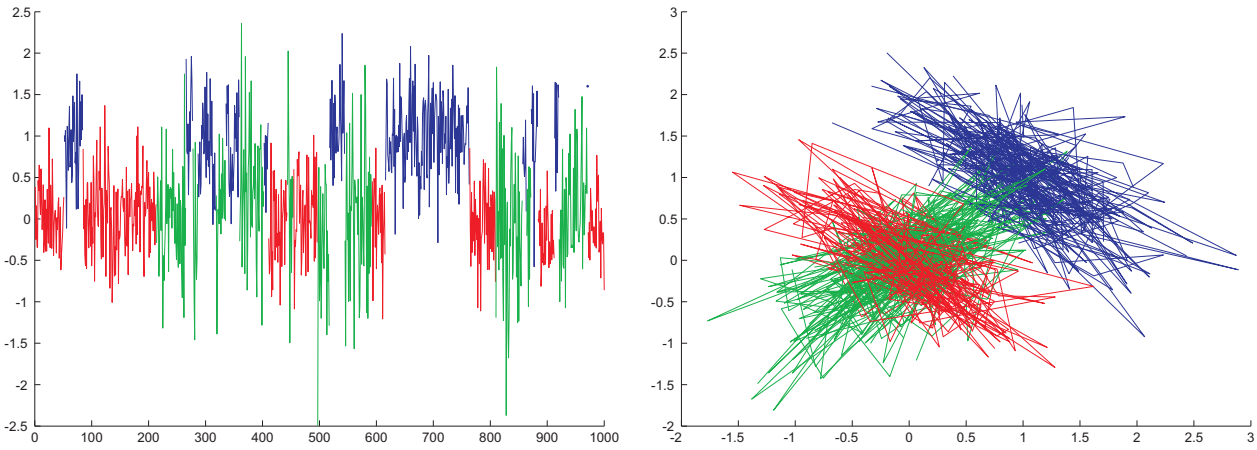
Комментарии по второму заданию по спецкурсу «Структурные методы анализа изображений и сигналов» 2009

Общая часть

Критерии выставления оценки

Основной подход при тестировании математических алгоритмов — это проверка на здравый смысл. При этом подходе алгоритму на вход предлагаются модельные примеры, на которых результат работы алгоритма имеет предсказуемую форму или обладает рядом особенностей, которые опять же можно предсказать заранее. В том случае, если полученный результат не согласуется со здравым смыслом, то это повод задуматься и продолжить тестирование алгоритма, либо скорректировать собственные представления о природе исследуемого процесса. Применявшая при проверке задания процедура тестирования алгоритмов конкретизирована далее для каждого варианта. При проверке задания оценка за задание снижалась, если допущенные в алгоритме ошибки можно легко обнаружить при тестировании на здравый смысл. В том случае, если допущенные ошибки могут себя проявить лишь при очень специальном подборе входных данных, то оценка за такие ошибки не снижалась. Также оценка за задание снижалась за откровенно неэффективный код MatLab, т.к. MatLab предоставляет эффективные средства (profiler) для тестирования кода на производительность, и обнаружить неэффективно вычисляемые операции не представляет труда.

Тестирование алгоритмов для классической СММ



(a) (b)

Рис. 1: Модельные задачи для тестирования классической СММ. Случай (a) — одномерное пространство, случай (b) — двухмерное пространство.

Для тестирования алгоритмов генерации, сегментации и обучения в классической СММ использовались две модельные задачи с хорошо отделимыми состояниями (см. рис. 1). В обоих случаях выбиралась некоторая СММ, из этой модели генерировались данные, затем на полученных данных запускалась процедура тестирования с истинными параметрами СММ, и результаты сравнивались с истинными значениями состояний. Также запускался EM-алгоритм, отслеживалось монотонное возрастание логарифма правдоподобия, и его результаты сравнивались с истинными значениями параметров СММ.

В первом случае использовалась СММ с параметрами $d = 1, K = 3$, $\mu_1 = 0, \mu_2 = 0, \mu_3 = 0$, $\sigma_1^2 = 0.1, \sigma_2^2 = 0.5, \sigma_3^2 = 0.1$, матрица A близка к единичной. Во втором случае $d = 2, K = 3$, $\boldsymbol{\mu}_1 = [0, 0], \boldsymbol{\mu}_2 = [0, 0], \boldsymbol{\mu}_3 = [1, 1]$, в матрицах Σ_i одно собственное значение в 5 раз больше другого, а собственные вектора повернуты на угол 45°, матрица A близка к единичной, причем переходы между красным и синим состояниями запрещены (см. рис. 1,b).

Первый вариант

Теория

Пусть $p_i(\tau)$ — априорная вероятность нахождения в состоянии i ровно τ моментов времени. Обозначим границы сегментов в сигнале через s_i так, что на участке $[s_{i-1} + 1, s_i]$ сигнал находится в состоянии t_i, $s_0 = 0, s_N = N$. Тогда совместное распределение вероятности можно записать следующим образом:

$$p(X, T, S) = p(t_1)p_{t_1}(s_1) \prod_{n=1}^{s_1} p(\boldsymbol{x}_n|t_1) \prod_{i=2}^{M} p(t_{i-1}, t_i)p_{t_i}(s_i - s_{i-1}) \prod_{n=s_{i-1}+1}^{s_i} p(\boldsymbol{x}_n|t_i)$$

Введем функцию Беллмана $V_n(k)$ как стоимость оптимальной сегментации до момента времени n, заканчивающейся в состоянии k, причем в следующий момент времени произойдет переход в другое состояние. Пусть также известно, что априорные распределения $p_i(\tau) = 0$ при $\tau < a$ и $\tau > b$. Тогда алгоритм Витерби для данного случая будет выглядеть следующим образом:

Проход вперед:

$$V_a(k) = \log \pi_k + \log p_k(a) + \sum_{j=1}^{a} \log p(\boldsymbol{x}_j|\boldsymbol{\mu}_k, \Sigma_k)$$

$$V_n(k) = \max[f_n(k), \max_{i \neq k} g_n(i, k)]$$

$$f_n(k) = \log \pi_k + \log p_k(n) + \sum_{j=1}^{n} \log p(\boldsymbol{x}_j|\boldsymbol{\mu}_k, \Sigma_k)$$

$$g_n(i, k) = \max_{m=n-b,\ldots,n-a} \left[V_m(i) + \log A(i, k) + \log p_k(n - m) + \sum_{j=m+1}^{n} \log p(\boldsymbol{x}_j|\boldsymbol{\mu}_k, \Sigma_k) \right]$$

$$S_n(k) = \begin{cases} \arg\max\limits_{i \neq k} g_n(i, k) & \text{, если } f_n(k) < \max\limits_{i \neq k} g_n(i, k) \\ \text{не определено} & \text{, иначе} \end{cases}$$

$$M_n(k) = \begin{cases} \arg \max\limits_{m=n-b,\ldots,n-a} \left[V_m(S_n(k)) + \log p_k(n - m) + \sum_{j=m+1}^{n} \log p(\boldsymbol{x}_j|\boldsymbol{\mu}_k, \Sigma_k) \right] & \text{, если } f_n(k) < \max\limits_{i \neq k} g_n(i, k) \\ 0 & \text{, иначе} \end{cases}$$

Проход назад:

$$\begin{aligned} n_{curr} &= N \\ s_{curr} &= \arg \max_{j=1,\ldots,K} V_N(j) \\ n_{next} &= M_{n_{curr}}(s_{curr}) \\ s_{next} &= S_{n_{curr}}(s_{curr}) \\ k^*(n_{next} + 1, \ldots, n_{curr}) &= s_{curr} \end{aligned}$$

Тестирование сегментации с заданным априорным распределением на длину сегмента

Пусть ограничения на геометрическое априорное распределение на длину сегмента задается параметрами a и b. Рассмотрим модельную задачу, представленную на рис. 1,a. При тестировании с параметрами $a = 1, b = +\infty$ результат сегментации должен примерно соответствовать истинным значениям состояний. При тестировании с параметрами $a = 10, b = 20$ должен наблюдаться эффект разделения длинного состояния на ограниченные сегменты, причем вынужденные перескоки происходят в ближайшее состояние, и длина такого перескока порядка 10 (см. рис. 2,a). При тестировании с параметрами $a = 1, b = 20$ (см. рис. 2,b) должен наблюдаться аналогичный эффект, только длина вынужденного состояния порядка 1.

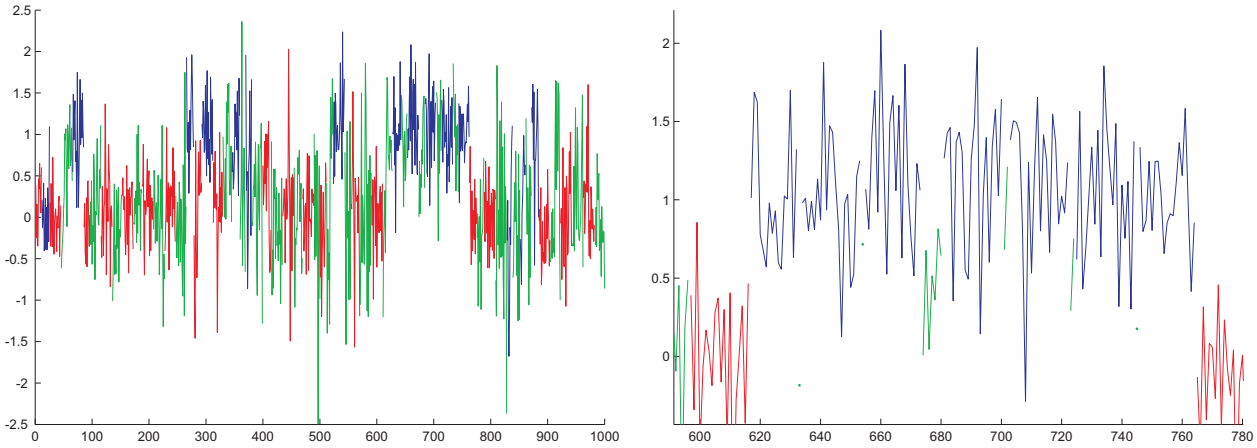


Рис. 2: Результаты сегментации с ограничением на длину сегмента. Случай (a) соответствует ограничению $[10, 20]$, случай (b) — $[1, 20]$.

Второй вариант

Теория

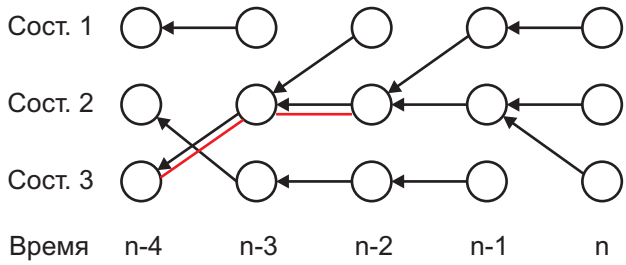


Рис. 3: Иллюстрация механизма принятия решения при он-лайн сегментации

Алгоритм Витерби в классическом варианте представляется следующим образом:

Проход вперед:

$$
\begin{aligned}
V_1(k) &= \log \pi_k + \log p(\boldsymbol{x}_1|\boldsymbol{\mu}_k, \Sigma_k) \\
V_n(k) &= \max_{j=1,\dots,K}(V_{n-1}(j) + \log A_{jk}) + \log p(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \Sigma_k),\ n = 2, \dots, N \\
S_n(k) &= \arg\max_{j=1,\dots,K}(V_{n-1}(j) + \log A_{jk}),\ n = 2, \dots, N
\end{aligned}
$$

Проход назад:

$$
\begin{aligned}
k^*(N) &= \arg\max_{j=1,\dots,K} V_N(j) \\
k^*(n) &= S_{n+1}(k^*(n+1)), n = N-1, \dots, 1
\end{aligned}
$$

Пусть при проходе вперед в момент времени n найдется момент времени в прошлом $n\prime$ такой, что все оптимальные траектории проходят через одно и то же состояние. В частности, если все $S_n(\cdot)$ совпадают между собой, т.е. $S_n(1) = S_n(2) = \dots = S_n(K)$, то $n\prime = n - 1$. Однако, существование $n\prime$ возможно не только для таких ситуаций. Для примера на рис. 3 $n\prime = n - 2$. Такие моменты времени n будем называть моментами принятия решений, соответствующие им $n\prime$ — границами принятия решений, а разницы $n - n\prime$ — глубиной принятия решений.

Один из возможных способов реализации алгоритма он-лайн сегментации — вычислять дополнительно в каждый момент времени величину

$$M_n(k) = |\{j : S_n(j) = k\}|,$$

т.е. количество стрелок, входящих в каждую вершину на предыдущем слое. Тогда, если в момент времени n оказывается, что $M_n(j) = 0$ для какого-то j, то осуществляется проход назад по оптимальной траектории из состояния j. На очередном шаге назад соответствующая величина $M_{n_1}(j_1)$ уменьшается на единицу. Если при этом она становится равной нулю, то процесс продолжается дальше. Теперь, если в какой-то момент времени $n\prime$ окажется, что все $M_{n\prime}(j)$, кроме одного, равны нулю, то $n\prime$ — искомый момент времени в прошлом для очередной границы он-лайн сегментации.

Тестирование алгоритма он-лайн сегментации

Для тестирования алгоритма он-лайн сегментации в данном случае возможно применение подхода «грубой силы» (по аналогии с задачами дискретной оптимизации, когда возможно перебрать все пространство аргументов и гарантированно найти строгий оптимум функции). Здесь для каждого момента времени n вычисляются оптимальные траектории для всех состояний j: $k_j^*(1), \ldots, k_j^*(n) = j$. Затем у найденных оптимальных траекторий находится общая часть, т.е. $n\prime : k_1^*(i) = k_2^*(i) = \cdots = k_K^*(i)\ \forall\ i = 1, \ldots, n\prime$.

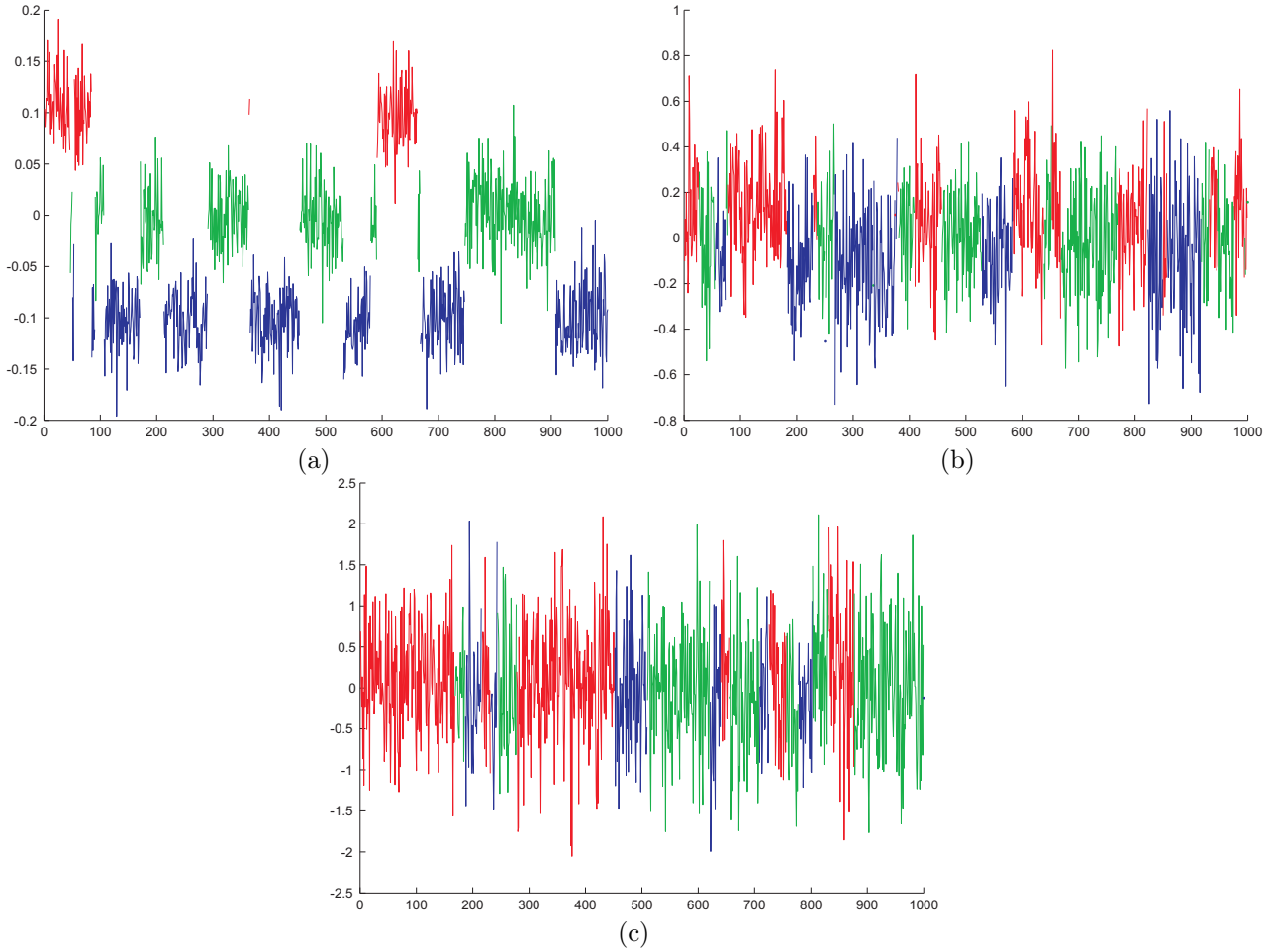


Рис. 4: Задачи для тестирования он-лайн сегментации. Случай (a) соответствует хорошо отделимым состояниям, в случае (b) состояния менее различимы, случай (c) соответствует практически полному смешению состояний.

В качестве задач для тестирования алгоритма он-лайн сегментации использовались следующие (см. рис. 4). Имеется одномерный сигнал, сгенерированный из СММ с тремя состояниями. В первой задаче все состояния хорошо отделимы друг от друга (см. рис. 4,a), например, $\mu_1 = -0.1,\ \mu_2 = 0,\ \mu_3 = 0.1, \sigma_1 = \sigma_2 = \sigma_3 = 0.001$. Затем в последующих задачах состояния становятся все более похожими друг на друга путем сокращения расстояния между центрами гауссиан, либо увеличением всех дисперсий (см. рис. 4,b и c). Каждый раз параметры СММ для он-лайн тестирования совпадают с параметрами СММ, из которой генерируется сигнал. При тестировании на указанных задачах у корректного алгоритма он-лайн сегментации должны наблюдаться следующие эффекты. В начале сегментация сигнала практически совпадает с истинными значениями. При этом количество моментов принятия решений практически совпадает с длиной сигнала, а глубина равна единице. Затем по мере смешения состояний отклонение от истинных значений при сегментации увеличивается, количество моментов принятия

решений сокращается, а расстояние между ними увеличивается. При этом глубина принятия решений в некоторых ситуациях начинает значительно отличаться от единицы (см. рис. 5). В пределе, когда состояния становятся практически неотличимыми, момент принятия решения всего один — это конец сигнала. Кроме того, если провести аналогичный эксперимент с двумя состояниями, то глубина принятия решений во всех случаях должна равняться единице. Действительно, в случае двух состояний любой случай $M_n(j) = 0$ для какого-то j означает, что для второго состояния k $M_n(k) = 2$, т.е. все оптимальные траектории в момент $n-1$ проходят через одно и тоже состояние k, а значит $n\prime = n-1$.

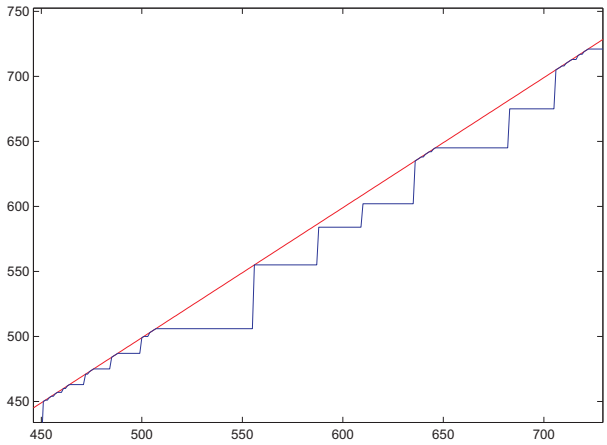


Рис. 5: Пример он-лайн сегментации. При осям отложены моменты времени. На графиках приведены границы принятия решений для каждого момента времени. Красный график соответствует он-лайн сегментации полностью без задержек, синий график показывает он-лайн сегментацию с плохо отличимыми состояниями. Здесь наблюдается как увеличение расстояний между соседними моментами принятия решений (горизонтальные отрезки), так и увеличение глубины принятия решений (разница между красным и синим графиком в момент вертикального скачка).

Третий вариант

Теория

Авторегрессионная скрытая марковская модель (AP CMM) задается следующим совместным распределением:

$$p(T, X|\Theta) = p(\boldsymbol{t}_1) \prod_{n=2}^{N} p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1}) \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{t}_n, \boldsymbol{x}_{n-1}, \ldots, \boldsymbol{x}_{n-M})$$

Здесь

$p(\boldsymbol{t}_1) = \prod_{j=1}^{K} \pi_j^{t_{1j}}$ — априорные вероятности состояний

$p(\boldsymbol{t}_n|\boldsymbol{t}_{n-1}) = \prod_{i=1}^{K} \prod_{j=1}^{K} A_{ij}^{t_{n-1,i} t_{nj}}$ — вероятности перехода между состояниями

$p(\boldsymbol{x}_n|\boldsymbol{t}_n, \boldsymbol{x}_{n-1}, \ldots, \boldsymbol{x}_{n-M}) = \prod_{j=1}^{K} \left(\mathcal{N}\left(\boldsymbol{x}_n|\boldsymbol{\mu}_j + \sum_{k=1}^{M} c_{jk}\boldsymbol{x}_{n-k}, \Sigma\right)\right)^{t_{nj}}$ — унарные вероятности

При анализе AP CMM необходимо для каждого $\boldsymbol{x}_n$ знать значения M предыдущих наблюдений $\boldsymbol{x}_{n-1}, \ldots, \boldsymbol{x}_{n-M}$. Здесь возможны два варианта: либо ввести искусственные наблюдения в начало сигнала $\boldsymbol{x}_0 = \boldsymbol{x}_{-1} = \cdots = \boldsymbol{x}_{-M+1} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}_n$, равные среднему значению наблюдаемого сигнала, либо проводить анализ сигнала для моментов времени $M+1, \ldots, N$.

Введем обозначение для непосредственной предыстории наблюдения $\boldsymbol{x}_n$:

$$Y_n = [\boldsymbol{x}_{n-1}, \ldots, \boldsymbol{x}_{n-M}] = \begin{pmatrix} x_{n-1}(1) & \cdots & x_{n-M}(1) \\ x_{n-1}(2) & \cdots & x_{n-M}(2) \\ \vdots & \ddots & \vdots \\ x_{n-1}(d) & \cdots & x_{n-M}(d) \end{pmatrix}$$

Тогда мат.ожидание распределения $p(\boldsymbol{x}_n|\boldsymbol{t}_n, \boldsymbol{x}_{n-1}, \ldots, \boldsymbol{x}_{n-M})$ при $t_{nj} = 1$ можно записать как $\boldsymbol{\mu}_j + Y_n\boldsymbol{c}_j$, где $\boldsymbol{c}_j$ — набор коэффициентов авторегрессии для j-ого состояния.

Алгоритм Витерби, а также E-шаг EM-алгоритма для AP CMM полностью повторяют алгоритмы для классической CMM с той лишь разницей, что вместо $p(\boldsymbol{x}_n|\boldsymbol{t}_n)$ подставляется значение $p(\boldsymbol{x}_n|\boldsymbol{t}_n, \boldsymbol{x}_{n-1}, \ldots, \boldsymbol{x}_{n-M})$. На M-шаге EM-алгоритма возникают следующие формулы пересчета значений параметров:

$$\boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} \gamma_{nj}(\boldsymbol{x}_n - Y_n \boldsymbol{c}_j)}{\sum_{n=1}^{N} \gamma_{nj}}$$

$$\Sigma = \frac{1}{N} \sum_{n=1}^{N} \sum_{j=1}^{K} \gamma_{nj}(\boldsymbol{x}_n - \boldsymbol{\mu}_j - Y_n \boldsymbol{c}_j)(\boldsymbol{x}_n - \boldsymbol{\mu}_j - Y_n \boldsymbol{c}_j)^T$$

$$\boldsymbol{c}_j = \left(\sum_{n=1}^{N} \gamma_{nj} Y_n^T \Sigma^{-1} Y_n \right)^{-1} \left(\sum_{n=1}^{N} \gamma_{nj} Y_n^T \Sigma^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu}_j) \right)$$

Легко видеть, что формулы пересчета для $\boldsymbol{\mu}_j$ и Σ переходят в классические при $\boldsymbol{c}_j = \mathbf{0}$. Также заметим, что в данном случае вычисление $\boldsymbol{\mu}_j$ зависит от $\boldsymbol{c}_j$, Σ зависит от $\boldsymbol{\mu}_j$ и $\boldsymbol{c}_j$, а $\boldsymbol{c}_j$ в свою очередь зависит от $\boldsymbol{\mu}_j$ и Σ. Поэтому здесь на M-шаге необходимо проводить вычисления итерационно до стабилизации значений $\boldsymbol{\mu}_j$, $\boldsymbol{c}_j$ и Σ. В противном случае нет гарантии, что значение правдоподобия на каждом шаге EM-алгоритма будет увеличиваться.

Тестирование AP CMM

Рис. 6: Модельные данные для тестирования AP CMM

Для тестирования AP CMM применялся тот же способ, что и при тестировании классической CMM. Выбиралась модель AP CMM, генерирующая хорошо отделимые состояния (см. рис. 6). Для этой модели проверялась близость сегментации к истинной при известных значениях параметров AP CMM, а также близость значений параметров к истинным для EM-алгоритма.

Имеет смысл выбирать такие модели AP CMM, в которых авторегрессионный процесс для каждого состояния является стационарным. В противном случае, в частности, дисперсия сигнала может стремится к бесконечности при увеличении длины сигнала. Пусть авторегрессионный процесс задается следующим образом:

$$x_n = \sum_{i=1}^{M} c_i x_{n-i} + \mu_0 + \xi_n.$$

Здесь ξ_n — независимый нормальный шум с нулевым мат.ожиданием и фиксированной дисперсией. Из теории известно, что процесс авторегрессии является стационарным, если у многочлена $\lambda^M - \sum_{i=1}^{M} c_i \lambda^{M-i}$ все корни λ_j (включая комплексные) по модулю строго меньше 1. Например, при $M = 2$ условие стационарности эквивалентно следующему:

$$\begin{aligned} -2 &< c_1 < 2, \\ -1 &< c_2 < 1 - |c_1|. \end{aligned}$$

Существует тесная связь между процессом авторегрессии и обыкновенным линейным дифференциальным уравнением. Рассмотрим эту связь на примере. Пусть имеется линейное дифференциальное уравнение вида:

$$f''(x) + 2\alpha f'(x) + (\alpha^2 + k^2)f(x) = 0.$$

Решением этого уравнения будут функции, соответствующие затухающим колебаниям заданной частоты:

$$f(x) = A_1 \exp(-\alpha x)\cos(kx) + A_2 \exp(-\alpha x)\sin(kx).$$

Здесь $A_{1,2}$ — произвольные константы.

Рассмотрим разностный аналог дифференциального уравнения, т.е. заменим первую и вторую производные на их разностные оценки $f_{n+1} - f_n$ и $f_{n+1} - 2f_n + f_{n-1}$ соответственно. Тогда

$$f_{n+1} - 2f_n + f_{n-1} + 2\alpha(f_{n+1} - f_n) + (\alpha^2 + k^2)f_n = 0.$$

Эквивалентно получаем процесс авторегрессии:

$$f_{n+1} = \frac{2 + 2\alpha - \alpha^2 - k^2}{1 + 2\alpha} f_n - \frac{1}{1 + 2\alpha} f_{n-1}.$$

Очевидно, что подобный переход от линейного дифференциального уравнения к процессу авторегрессии возможен для любого M. Таким образом, в случае отсутствия шума, авторегрессионный процесс порождает функцию, которая является решением соответствующего ему линейного дифференциального уравнения. Благодаря этому свойству, мы можем строить модельные AP CMM, в которых одно состояние имеет форму заданного шаблона.