# Вычисление функции правдоподобия для смеси гауссовских распределений

*Г.А. Ситкарев, <sitkarev@unixkomi.ru>*

Сыктывкарский Государственный Университет

## 1. Функция правдоподобия

Функция правдоподобия встречается в различных приложениях теории вероятности. Во множестве практических случаев, стоит задача найти её максимум, что связано с вычислением её производной. В таком случае, как правило, пользуются её логарифмом, ибо логарифм функции — монотонно возрастает, а потому её максимум будет достигнут в той же точке, что и у самой функции.

В нашем случае, практический интерес состоит в вычислении функции правдоподобия для смеси из многомерных гауссовских распределений, в связи с этим, её составляющие будут рассматриваться по отдельности, начиная с функции плотности.

Формула функции плотности многомерного гауссовского распределения выглядит так:

$$\eta(\mathbf{x} \mid \mu, \Sigma) = \frac{1}{(2\pi)^{M/2} \left|\Sigma\right|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})}. \tag{1}$$

Здесь:

- $\mathbf{x}$ вектор $(x_1, x_2, x_3, \ldots, x_M)$, размерности $M$;
- $\Sigma$ ковариационная матрица а $|\Sigma|$ — её детерминант;
- $\boldsymbol{\mu}$ вектор средних значений $\mathbf{x}$.

Обозначение ‘‘$\eta(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma)$’’ указывает на то, что функция получает вектор $\mathbf{x}$, как аргумент, и параметризуется значениями $\Sigma$ и $\boldsymbol{\mu}$.

Плотность вероятности для смеси из $K$ гауссовских распределений, вес каждой из которых задан $\omega_k$, задаётся так:

$$f_{\mathbf{x}}(\mathbf{x}) = \sum_{k=1}^{K} \omega_k \, \eta(\mathbf{x} \mid \boldsymbol{\mu}_k, \Sigma_k). \tag{2}$$

Так как функция $f_{\mathbf{x}}(\mathbf{x})$ есть плотность вероятности, её интеграл должен быть равен единице, очевидно, что сумма всех весов $\sum \omega$ тоже должна равняться единице, и все $\omega_k > 0$.

Функция правдоподобия рассчитывается всегда для конечного набора значений $\mathbf{x}_n$, как произведение их плотностей вероятности:

$$\mathscr{L} = f_{\mathbf{x}}(\mathbf{x}_1) f_{\mathbf{x}}(\mathbf{x}_2) \cdots f_{\mathbf{x}}(\mathbf{x}_N) = \prod_{n=1}^{N} f_{\mathbf{x}}(\mathbf{x}_n).$$

Функция правдоподобия для смеси из $K$ гауссовских распределений для всего набора значений $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$ тогда:

$$\mathscr{L} = \prod_{n=1}^{N} \sum_{k=1}^{K} \omega_k \eta(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \Sigma_k). \tag{3}$$

В практическом смысле, в ходе выполнения какого-то алгоритма, нас чаще всего интересует как изменялась функция правдоподобности, а не её абсолютное значение. Поэтому вместо вычисления $\mathscr{L}$ можно взять её логарифм, при этом $\log \mathscr{L}$ превратит произведение в сумму:

$$\begin{aligned}
\log \mathscr{L} &= \log \left[ \prod_{n=1}^{N} \sum_{k=1}^{K} \omega_k \eta(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \Sigma_k) \right] \\
&= \log \sum_{k=1}^{K} \omega_k \eta(\mathbf{x}_1 \mid \boldsymbol{\mu}_k, \Sigma_k) + \cdots + \log \sum_{k=1}^{K} \omega_k \eta(\mathbf{x}_N \mid \boldsymbol{\mu}_k, \Sigma_k) \\
&= \sum_{n=1}^{N} \log \sum_{k=1}^{K} \omega_k \eta(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \Sigma_k).
\end{aligned}$$

Существует один важный практический аспект. При подсчёте сумм плотностей, часто значения плотностей оказываются настолько малыми, что в операциях с плавающей точкой происходит исчезновение порядка (floating point underflow). В арифметике IEEE-754 исчезновение порядка приводит к тому что число денормализуется и плавно, а не резко, приближается к нулю. Большинство сопроцессоров операции с денормализованными числами выполняют через микропрограмму, а не аппаратно, что обычно медленнее на порядок. Потому целесообразно и при вычислении $\eta(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma)$ переходить к логарифмированию:

$$\begin{aligned}
\log \eta(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma) &= -\frac{M}{2} \log(2\pi) - \frac{1}{2} \log |\Sigma| - \frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \\
&= -\frac{1}{2} \left( M \log(2\pi) + \log |\Sigma| + (\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right).
\end{aligned}$$

После логарифмирования, значения должны быть восстановлены экспоненцированием.

## 2. Формула log-sum-exp

В формуле для смеси гауссовских распределений каждая компонента плотности имеет вес $\omega_k$. Будем обозначать логарифм плотности распределения с весом $\omega$ как $\gamma$:

$$\gamma = \log \eta(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma) + \log \omega$$

Формула для расчёта $\log \mathscr{L}$ тогда примет вид:

$$\log \mathscr{L} = \sum_{n=1}^{N} \log \left( \sum_{k=1}^{K} \exp(\gamma_k) \right).$$

Значение $exp(\gamma_k)$ может опять таки оказаться настолько малым, что это приведёт к исчезновению порядка. Для того чтобы избежать этого, при вычислении логарифма суммы сначала находится самое большое $\gamma_{\max}$, а затем применяется формула

$$\log \left( \sum_{k=1}^{K} \exp(\gamma_k) \right) = \gamma_{\max} + \log \left( \sum_{k=1}^{K} \exp(\gamma_k - \gamma_{\max}) \right),$$

которая гарантирует, что хотя бы одно значение не вызовет исчезновения порядка, а остальные

значения в таком случае всё равно не окажут влияния на сумму в силу своей малости. Такой трюк известен под названием «*формула log-sum-exp*».

### 3. Эффективное вычисление функции плотности

Функция плотности гауссовского распределения (1) содержит скаляр

$$(\mathbf{x} - \boldsymbol{\mu})^T \, \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) = \theta^T \, \Sigma^{-1} \theta.$$

Конечно, можно вычислить это значение, как говорится, «в лоб», найдя обратную ковариационную матрицу и выполнив две операции умножения «матрица-вектор». Тем не менее, существует способ сделать это проще и компактнее, что не маловажно для систем, обладающих скромными ресурсами. Свойства ковариационной матрицы позволяют осуществить упрощение, потому что эта матрица *симметричная* и, в большинстве практических случаев, *положительно-определённая*, а значит её можно факторизовать, пользуясь *разложением Холецкого*, на две треугольные матрицы

$$\Sigma = \mathbf{C}\mathbf{C}^T,$$

где:

- $\mathbf{C}$ нижняя треугольная матрица со строго положительными диагональными элементами;
- $\mathbf{C}^T$ верхняя треугольная матрица.

Любую симметричную положительно-определённую матрицу можно представить в таком виде. Взяв нижнюю треугольную матрицу $\mathbf{C}$, мы можем очень быстро, через прямую подстановку, найти вектор $u$ такой, что будет выполняться равенство

$$\mathbf{C}u = \theta.$$

Если теперь заменить вектор $\theta$ на $\mathbf{C}u$, получим:

$$\theta^T \, \Sigma^{-1} \theta = (\mathbf{C}u)^T \, (\mathbf{C}\mathbf{C}^T)^{-1} \, \mathbf{C}u$$

$$= u^T \mathbf{C}^T \mathbf{C}^{-T} \mathbf{C}^{-1} \mathbf{C} u$$

$$= u^T u = u \cdot u.$$

Алгоритм разложения Холецкого достаточно прост и эффективен в реализации на ЭВМ, и, в том случае если размерность вектора $\mathbf{x}$ невелика и известна заранее, все циклы алгоритма, вычисляющие коэффициенты матриц разложения, можно развёрнуть.

#### 3.1. Численный пример для GNU Octave

В заключении, приведём пример вычислений, на котором можно убедиться в работоспособности предложенного выше метода.

```
# input data
C = [ 1, 0, 0; 2, 3, 0; 4, 5, 6];
x = [ 0.24; 0.55; 0.29 ];
M = C*C';

# compute using literal formula
ans1 = x'*M^-1*x;

# prove that L equals C if we factor it using Cholesky
L = chol(M, 'lower');

# compute using Cholesky lower matrix inverse
u = (L^-1)*x;
ans2 = u'*u;

# compute using forward substitution
n = length(x);
z = zeros(n, 1);
for i=1:n
    z(i) = (x(i) - (L(i, :) * z)) / L(i,i);
end
ans3 = z'*z;

printf("ans1 = %f, ans2 =% f, ans3 = %f0, ans1, ans2, ans3);
```