

ПЕРЕДАЧА СООБЩЕНИЙ И СЭМПЛИРОВАНИЕ

Сергей Николенко

СПбГУ — Санкт-Петербург

18 марта 2025 г.

Random facts:

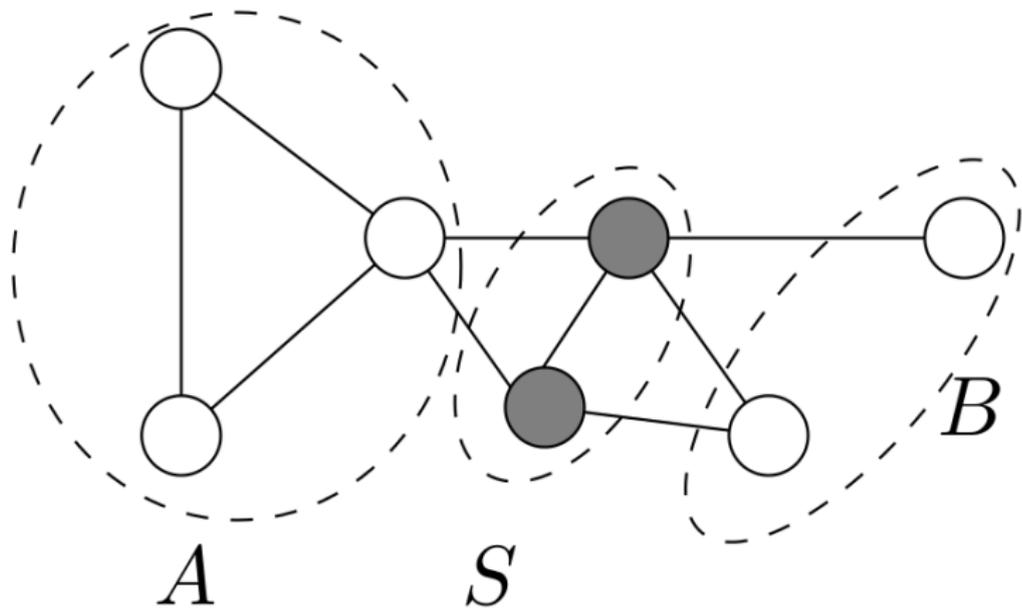


- 18 марта 1314 г. состоялся суд над великим магистром Жаком Моле и другими тамплиерами; де Моле и приора Нормандии сожгли, остальных посадили
- 18 марта 1869 г. Н.А. Меншуткин от имени Д.И. Менделеева сообщил Русскому химическому обществу об открытии соотношения между свойствами элементов и их атомными весами, более известном как периодическая система
- 18 марта 1871 г. была создана Парижская коммуна, а 18 марта 1920 г. на II конгрессе Коминтерна В.И. Ленин напомнил о предстоящем через год 50-летию и поставил задачу: «К этому времени Франция должна стать советской республикой!»
- 18 марта 1917 г. в «Вестнике Временного правительства» были опубликованы Акты об отречении Государя Императора Николая II и Великого Князя Михаила Александровича, Николай II был арестован, началось всероссийское совещание Советов рабочих и солдатских депутатов и снова начала выходить запрещённая газета «Правда»
- 18 марта 1965 г. Алексей Леонов впервые в истории человечества вышел в открытый космос, а каждый член группы «Rolling Stones» был оштрафован на 5 фунтов за публичное справление малой нужды

ДРУГИЕ ГРАФИЧЕСКИЕ МОДЕЛИ

- Можно сделать и так, чтобы условие независимости было (более) локальным.
- Для этого нужно задавать модели ненаправленными графами. В них условие совсем естественное: множество вершин X условно независимо от множества вершин Y при условии множества вершин Z , если любой путь от X к Y проходит через Z .
- В частности, очевидно,
$$p(x_i, x_j | x_{k \neq i, j}) = p(x_i | x_{k \neq i, j})p(x_j | x_{k \neq i, j})$$
 тогда и только тогда, когда x_i и x_j не соединены ребром.
- Такие модели называются *марковскими сетями* (Markov random fields).

УСЛОВНАЯ НЕЗАВИСИМОСТЬ В НЕНАПРАВЛЕННЫХ МОДЕЛЯХ



- Поэтому в ненаправленных моделях локальные распределения соответствуют кликам в графе, и факторизация получается в виде

$$p(x_1, \dots, x_k) = \frac{1}{Z} \prod \psi_C(x_C),$$

где C – максимальные клики, ψ_C – неотрицательные функции (*потенциалы*), а Z – нормировочная константа (partition function).

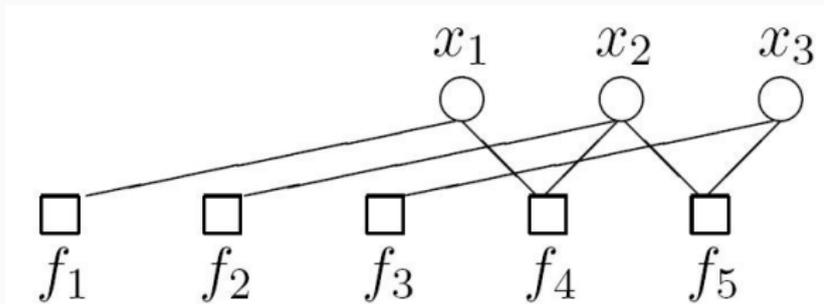
- Поскольку $\psi_C \geq 0$, их обычно представляют как экспоненты:

$$\psi_C(x_C) = \exp(-E_C(x_C)),$$

E_C – функции энергии, они суммируются в полную энергию системы (это всё похоже на статистическую физику, отсюда и терминология).

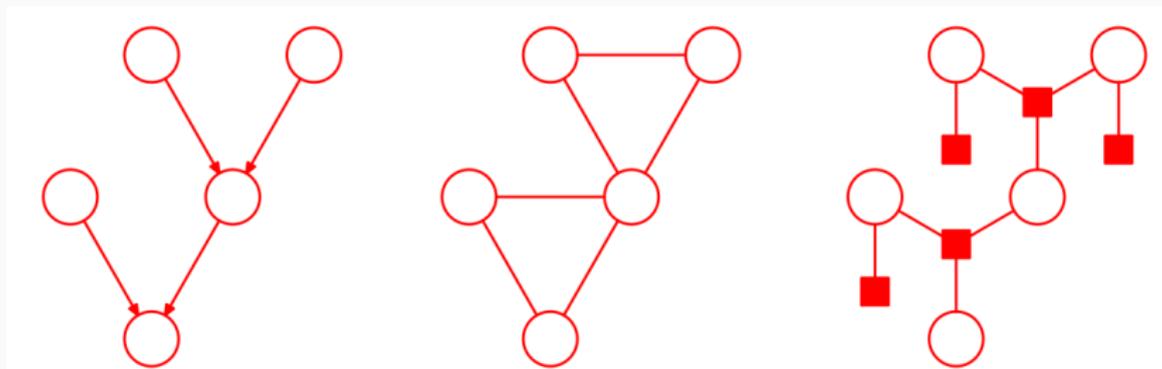
- Интересный факт: назовём *идеальной картой* (perfect map) распределения D графическую модель G , если все условные независимости, присутствующие в D , отображены в G , и наоборот (ничего лишнего). Тогда идеальные карты в виде направленных моделей существуют не у всех распределений, в виде ненаправленных тоже не у всех, и эти множества существенно различаются (бывают распределения, которые нельзя идеально выразить направленной моделью, но можно ненаправленной, и наоборот).

- Важная для вывода модификация – *фактор-граф* (можно построить и по направленной модели, и по ненаправленной).
- Фактор-граф – двудольный граф функций и переменных.
- Функция, соответствующая графу, – произведение всех входящих в него функций (т.е. то самое разложение и есть).
- Пример: $p(x_1, x_2, x_3) = f_1(x_1)f_2(x_2)f_3(x_3)f_4(x_1, x_2)f_5(x_2, x_3)$.



АЛГОРИТМ ПЕРЕДАЧИ СООБЩЕНИЙ

ТРИ ПРЕДСТАВЛЕНИЯ



- Чтобы поставить задачу в общем виде, рассмотрим функцию

$$p^*(X) = \prod_{j=1}^m f_j(X_j),$$

где $X = \{x_i\}_{i=1}^n$, $X_j \subseteq X$.

- Т.е. мы рассматриваем функцию, которая раскладывается в произведение нескольких других функций.

- Задача нормализации: найти $Z = \sum_X \prod_{j=1}^m f_j(X_j)$.
- Задача маргинализации: найти

$$p_i^*(x_i) = \sum_{k \neq i} p^*(X).$$

Также может понадобиться, например, $p_{i_1 i_2}$, но реже.

- Поиск гипотезы максимального правдоподобия:

$$\mathbf{x}^* = \arg \max_X p(X).$$

- Все эти задачи NP-трудные.
- То есть, если мир не рухнет, сложность их решения в худшем случае возрастает экспоненциально.
- Но можно решить некоторые частные случаи.

- Давайте начнём с графа в виде (ненаправленной) цепи:

$$p(x_1, \dots, x_n) = \frac{1}{Z} \psi_{1,2}(x_1, x_2) \dots \psi_{n-1,n}(x_{n-1}, x_n).$$

- Мы хотим найти

$$p(x_k) = \sum_{x_1} \dots \sum_{x_{k-1}} \sum_{x_{k+1}} \dots \sum_{x_n} p(x_1, \dots, x_n).$$

- Очевидно, тут можно много чего упростить; например, справа налево:

$$\begin{aligned} \sum_{x_n} p(x_1, \dots, x_n) &= \\ &= \frac{1}{Z} \psi_{1,2}(x_1, x_2) \dots \psi_{n-2,n-1}(x_{n-2}, x_{n-1}) \sum_{x_n} \psi_{n-1,n}(x_{n-1}, x_n). \end{aligned}$$

- Эту сумму можно вычислить отдельно и продолжать в том же духе справа налево, потом аналогично слева направо.

- В итоге процесс сойдётся на узле x_k , куда придут два «сообщения»: слева

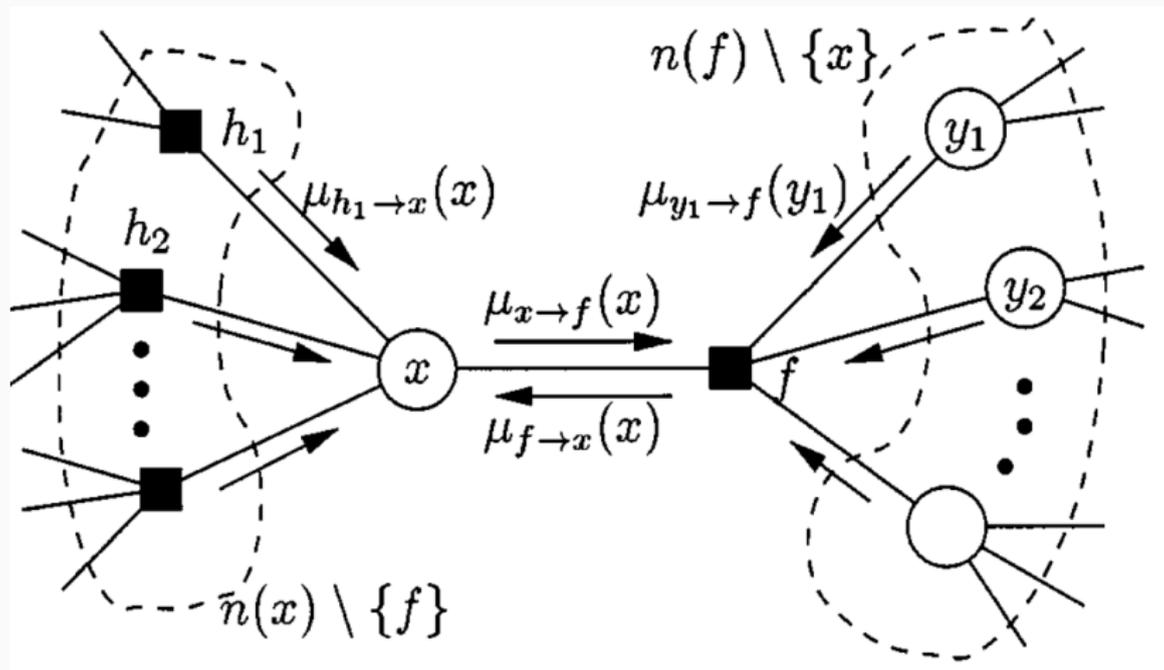
$$\mu_\alpha(x_k) = \sum_{x_{k-1}} \psi_{k-1,k}(x_{k-1}, x_k) \left[\dots \sum_{x_2} \psi_{2,3}(x_2, x_3) \left[\sum_{x_1} \psi_{1,2}(x_1, x_2) \right] \dots \right],$$

справа

$$\mu_\beta(x_k) = \sum_{x_{k+1}} \psi_{k,k+1}(x_k, x_{k+1}) \left[\dots \left[\sum_{x_n} \psi_{n-1,n}(x_{n-1}, x_n) \right] \dots \right].$$

- Каждую частичную сумму можно рассматривать как «сообщение» от узла к своему соседу, причём это сообщение – функция от соседа.

- Чтобы обобщить, удобно рассмотреть опять фактор-граф.
- Предположим, что фактор-граф – дерево (если не дерево, так просто не работает).
- Алгоритм передачи сообщений решает задачу маргинализации для функции вида $p(x_1, \dots, x_n) = \prod_s f_s(X_s)$, заданной в виде фактор-графа.
- Передаём сообщения по направлению к нужному узлу от переменных к функциям и наоборот.



- Чтобы найти $p(x_k)$, запишем

$p(x_1, \dots, x_n) = \prod_{s \in \neq(x_k)} F_s(x_k, X_s)$, где X_s – переменные из поддерева с корнем в f_s . Тогда

$$\begin{aligned} p(x_k) &= \sum_{x_{i \neq k}} p(x_1, \dots, x_n) = \prod_{s \in \neq(x_k)} \left[\sum_{X_s} F_s(x_k, X_s) \right] = \\ &= \prod_{s \in \neq(x_k)} \mu_{f_s \rightarrow x_k}(x_k), \end{aligned}$$

где $\mu_{f_s \rightarrow x_k}(x_k)$ – сообщения от соседних функций к переменной x_k .

- Чтобы найти $\mu_{f_s \rightarrow x_k}(x_k)$, заметим, что $F_s(x_k, X_s)$ тоже можно разложить по соответствующему подграфу:

$$F_s(x_k, X_s) = f_s(x_k, Y_s) \prod_{y \in Y_s} G_y(y, X_{s,y}),$$

где Y_s – переменные, непосредственно связанные с f_s (кроме x_k), $X_{s,y}$ – соответствующие поддеревья.

- Итого получаем

$$\begin{aligned} \mu_{f_s \rightarrow x_k}(x_k) &= \sum_{Y_s} f_s(x_k, Y_s) \prod_{y \in Y_s} \left(\sum_{X_{s,y}} G_y(y, X_{s,y}) \right) = \\ &= \sum_{Y_s} f_s(x_k, Y_s) \prod_{y \in Y_s} \mu_{y \rightarrow f_s}(y). \end{aligned}$$

- Можно аналогично подсчитать, что

$$\mu_{y \rightarrow f_s}(y) = \prod_{f \in \neq(y) f_s} \mu_{f \rightarrow y}(y).$$

- Итак, получился простой и понятный алгоритм:
 - как только узел получил сообщения от всех соседей, кроме одного, он сам начинает передавать сообщение в этого соседа;
 - сообщение по ребру между функцией и переменной является функцией от этой переменной;
 - узел-переменная x передаёт сообщение

$$\mu_{x \rightarrow f}(x) = \prod_{g \in \text{neigh}(x) \setminus \{f\}} \mu_{g \rightarrow x}(x);$$

- узел-функция $f(x, Y)$ передаёт сообщение

$$\mu_{f \rightarrow x}(x) = \sum_{y \in Y} f(x, Y) \prod_{y \in Y} \mu_{y \rightarrow f}(y);$$

- начальные сообщения в листьях $\mu_{x \rightarrow f}(x) = 1, \mu_{f \rightarrow x}(x) = f(x)$.

- Когда сообщения придут из всех соседей в какую-то переменную x_k , можно будет подсчитать

$$p(x_k) = \prod_{f \in \bar{\neq}(x_k)} \mu_{f \rightarrow x_k}(x_k).$$

- Когда сообщения придут из всех соседей в какой-то фактор $f_s(X_s)$, можно будет подсчитать совместное распределение

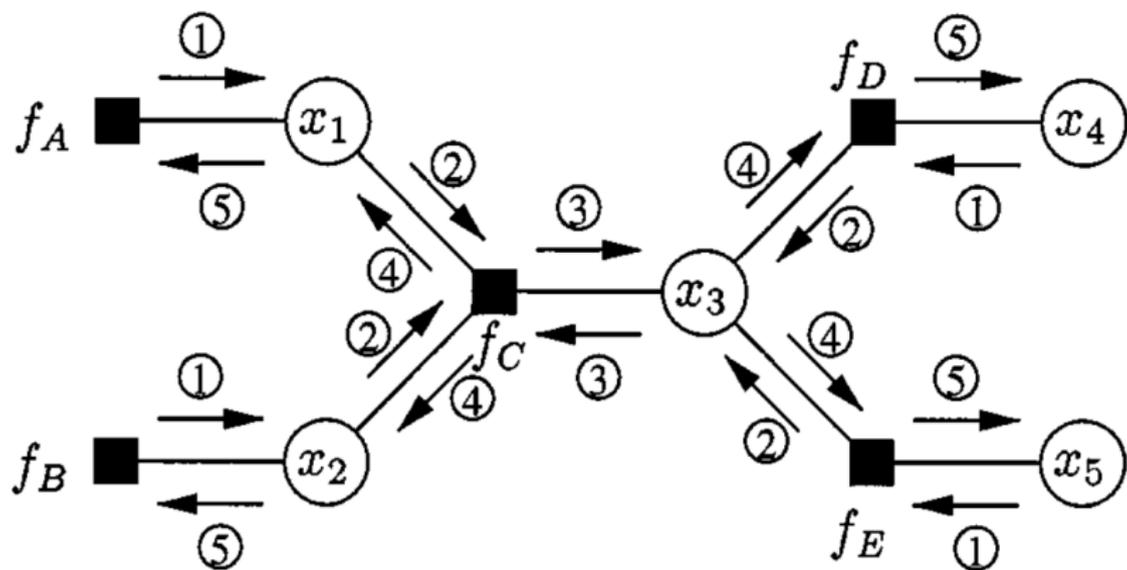
$$p(X_s) = f_s(X_s) \prod_{y \in \bar{\neq}(f_s)} \mu_{y \rightarrow f_s}(y).$$

- За два прохода (по каждому ребру туда и обратно) можно будет подсчитать маргиналы во всех узлах.

- Это называется алгоритм sum-product, потому что сообщение вычисляется как

$$\mu_{f \rightarrow x}(x) = \sum_{y \in Y} f(x, Y) \prod_{y \in Y} \mu_{y \rightarrow f}(y).$$

- Задача максимизации $\arg \max_x p(x_1, \dots, x_n)$ решается так же, но алгоритмом max-sum: сумма заменяется на максимум, а произведение на сумму.



ТАК ЧТО ЖЕ ДЕЛАТЬ С БАЙЕСОВСКОЙ СЕТЬЮ?

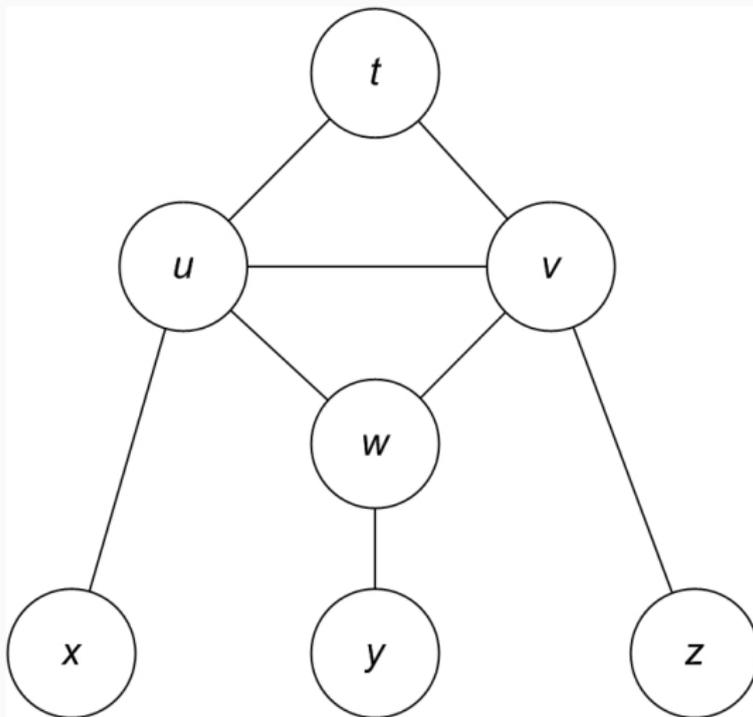
Для модели не в виде фактор-графа надо просто представить её в виде фактор-графа тем или иным способом.

Для байесовской сети это может означать, что надо сначала сделать морализацию, а потом добавить факторы в явном виде.

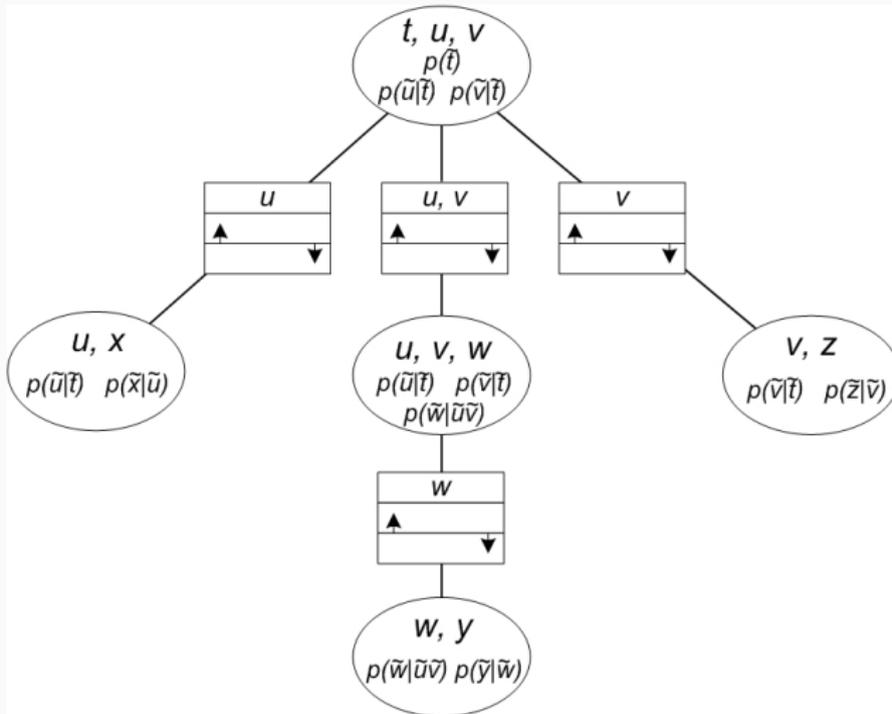
ТАК ЧТО ЖЕ ДЕЛАТЬ С БАЙЕСОВСКОЙ СЕТЬЮ?



ТАК ЧТО ЖЕ ДЕЛАТЬ С БАЙЕСОВСКОЙ СЕТЬЮ?



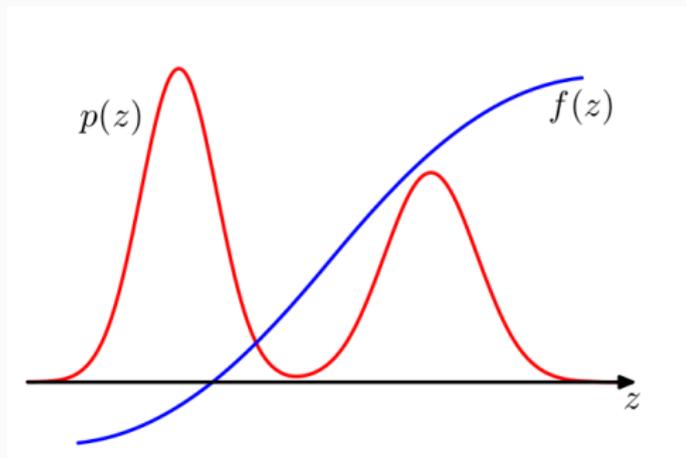
ТАК ЧТО ЖЕ ДЕЛАТЬ С БАЙЕСОВСКОЙ СЕТЬЮ?



ПОСТАНОВКА ЗАДАЧИ СЭМПЛИРОВАНИЯ

- Пусть у нас есть некоторое вероятностное распределение.
- Как с ним работать? Как, например, его симулировать?
- Мы не всегда можем приблизить (как по методу Лапласа) распределение каким-нибудь известным так, чтобы всё посчитать в явном виде.
- Например, в кластеризации: мультимодальное распределение с кучей параметров, что с ним делать?

- Однако часто задача не в том, чтобы что-то сделать с самой плотностью, а в том, чтобы считать ожидания функций:



- Пусть имеется некое распределение $p(x)$.
- Задача 1: научиться генерировать сэмплы $\{x^{(r)}\}_{r=1}^R$ по $p(x)$.
- Задача 2: научиться оценивать ожидания функций по распределению $p(x)$, т.е. научиться оценивать интегралы вида

$$E_p[f] = \int p(x)f(x)dx.$$

- Мы будем обычно предполагать, что x — это вектор из \mathbb{R}^n с компонентами x_n , но иногда будем рассматривать дискретные множества значений.
- Функции f — это, например, моменты случайных величин, зависящих от x .
- Например, если $t(x)$ — случайная величина, то её среднее — это $E_p[t(x)] (\int p(x)t(x)dx)$, а её дисперсия равна $E_p[t^2] - (E_p[t])^2$.
- И мы предполагаем, что явно вычислить не получается — слишком сложная функция p .

- Мы будем заниматься только сэмплингом, потому что задача оценки ожиданий функций легко решится, если мы научимся делать сэмплинг.
- Как она решится?

- Мы будем заниматься только сэмплингом, потому что задача оценки ожиданий функций легко решится, если мы научимся делать сэмплинг.
- Как она решится?
- Нужно взять сэмплы $\{x^{(r)}\}_{r=1}^R$ и подсчитать

$$\hat{f} = \frac{1}{R} \sum_r f(x^{(r)}).$$

- Ожидание \hat{f} равно $E_p[f]$, а дисперсия убывает обратно пропорционально R .

- Пример применения: вспомним, где мы часто вычисляем ожидания – в алгоритме EM, на E-шаге:

$$Q(\theta, \theta^{\text{old}}) = \int p(\mathbf{Z} | \mathbf{X}, \theta^{\text{old}}) \ln p(\mathbf{Z}, \mathbf{X} | \theta) d\mathbf{Z}.$$

- Давайте приблизим:

$$Q(\theta, \theta^{\text{old}}) \approx \frac{1}{R} \sum_{r=1}^R \ln p(\mathbf{Z}^{(r)}, \mathbf{X} | \theta) d\mathbf{Z}.$$

- А потом будем это приближение оптимизировать; получится *Monte Carlo EM*.
- Пример ещё проще: байесовские предсказания – это ожидания известных функций по сложному апостериорному распределению, и посчитать их руками обычно сложно.

СЭМПЛИРОВАНИЕ ИЗВЕСТНЫХ ФУНКЦИЙ

СЭМПЛИРОВАНИЕ ИЗВЕСТНОЙ ПЛОТНОСТИ

- Как сэмплировать из известного распределения? Ну скажем, нормального?
- Предположим, что умеем сэмплировать $z \in [0, 1]$ равномерно, `rand`.
- Какое будет распределение $p(y)$ у $y = f(z)$?

СЭМПЛИРОВАНИЕ ИЗВЕСТНОЙ ПЛОТНОСТИ

- Как сэмплировать из известного распределения? Ну скажем, нормального?
- Предположим, что умеем сэмплировать $z \in [0, 1]$ равномерно, `rand`.
- Какое будет распределение $p(y)$ у $y = f(z)$?

$$p(y) = p(z) \left| \frac{dz}{dy} \right|, \text{ и тут } p(z) = 1.$$

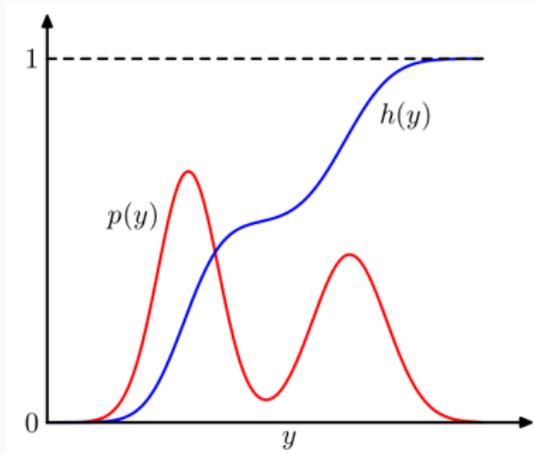
- Нам надо выбрать $f(z)$ так, чтобы получилось заданное $p(y)$. Это что за $f(z)$ будет?

СЭМПЛИРОВАНИЕ ИЗВЕСТНОЙ ПЛОТНОСТИ

- Надо выбрать

$$z = h(y) = \int_{-\infty}^y p(\hat{y}) d\hat{y},$$

неопределённый интеграл от $p(y)$.



- Т.е. надо как-то найти $h^{-1}(z)$.

- Например, что будет для экспоненциального распределения

$$p(y) = \lambda e^{-\lambda y}, \text{ где } y \in [0, \infty)?$$

- Например, что будет для экспоненциального распределения

$$p(y) = \lambda e^{-\lambda y}, \text{ где } y \in [0, \infty)?$$

- Будет интеграл от нуля, $h(y) = 1 - e^{-\lambda y}$, и $y = \frac{1}{\lambda} \ln(1 - z)$.
- То же и с многомерными распределениями, только теперь якобиан

$$p(y_1, \dots, y_M) = p(z_1, \dots, z_M) \left| \frac{d(z_1, \dots, z_M)}{d(y_1, \dots, y_M)} \right|.$$

- А как гауссиан сэмплировать?

- А как гауссиан сэмплировать?
- Преобразование Бокса-Мюллера: сначала сгенерируем (z_1, z_2) равномерно из единичного круга, потом посчитаем

$$y_1 = z_1 \sqrt{\frac{-2 \ln r^2}{r^2}}, \quad y_2 = z_2 \sqrt{\frac{-2 \ln r^2}{r^2}}, \quad \text{где } r^2 = z_1^2 + z_2^2.$$

- Тогда совместное распределение

$$p(y_1, y_2) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}y_1^2} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}y_2^2}.$$

- Всё понятно?..

ВЫБОРКА С ОТКЛОНЕНИЕМ

Что же сложного в сэмплинге?

- Мы предполагаем, что дана функция $p^*(x)$, которая отличается от $p(x)$ только нормировочной константой $Z = \int p^*(x)dx$: $p(x) = p^*(x)/Z$.
- Почему трудно делать сэмплинг?
- Во-первых, мы обычно не знаем Z ; но это не главное.
- Главное — обычно правильные сэмплы p^* часто попадают туда, где p^* велика. А как определить, где она велика, не вычисляя её *везде*?

- Простейшая идея: давайте дискретизируем пространство, вычислим p^* на каждом участке (пусть она гладкая), потом будем брать дискретные сэмплы, зная все вероятности (это нетрудно).
- Сколько же будет дискретных участков?
- Главная проблема — обычно велика размерность x .
Например, если разделить каждую ось на 20 участков, то участков будет 20^n ; а n в реальных задачах может достигать нескольких тысяч...
- Иными словами, такой подход никак не работает.

ПРИМЕР: СКОЛЬКО В ОЗЕРЕ НЕФТИ?

- Перед вами — участок, под которым залежи нефти (да хоть подземное озеро нефти).
- Вам нужно определить, сколько её тут.
- Вы можете проводить замер в каждой конкретной точке, чтобы определить глубину слоя в этой точке.
- Проблема в том, что значительная часть общего объёма нефти может быть сосредоточена в глубоких, но узких каньонах.
- И это только размерность два. :)

- Может быть, всё-таки получится решить хотя бы вторую задачу?
- Давайте брать сэмплы $\{x^{(r)}\}_{r=1}^R$ *равномерно* из всего пространства, затем вычислять там p^* и нормализовать посредством $Z_R = \sum_{r=1}^R p^*(x^{(r)})$.
- Тогда \hat{f} можно будет оценить как

$$\hat{f} = \frac{1}{Z_R} \sum_{r=1}^R f(x^{(r)}) p^*(x^{(r)}).$$

- В чём проблема?

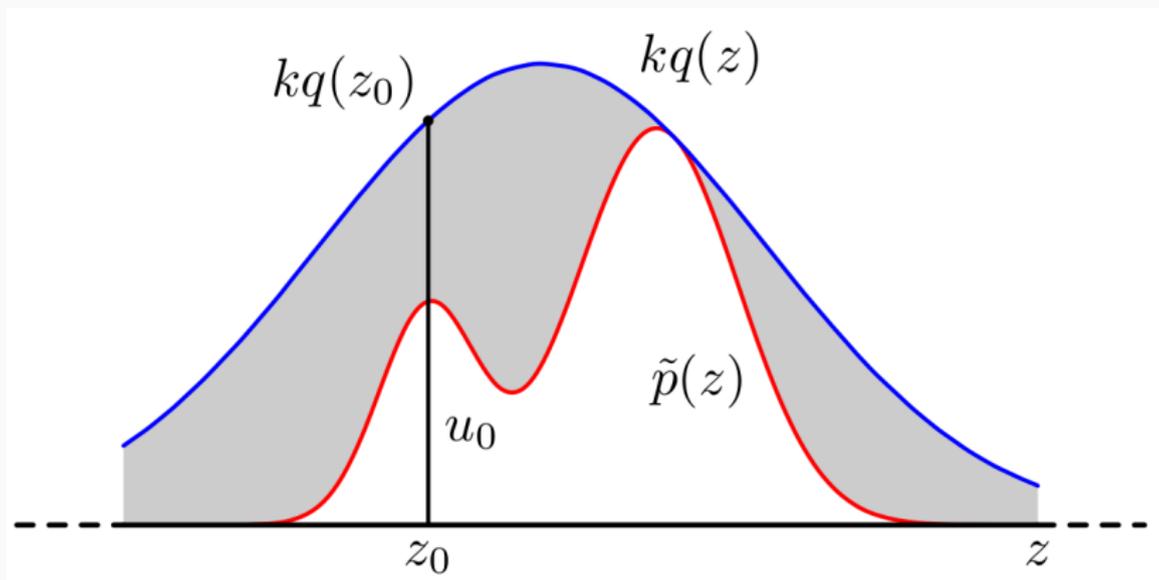
- Да в том же самом.
- Обычно значительная часть p^* сосредоточена в очень небольшой части пространства.
- Вероятность попасть в неё за R равномерно выбранных сэмплов тоже экспоненциально мала (например, если по каждой оси вероятность попасть $1/2$, и всё независимо, то получится вероятность 2^{-n}).
- Так что даже вторую задачу решить не получится.

- Но что-то всё-таки делать надо.
- Выборка с отклонением — rejection sampling.
- Наше предположение теперь в том, что у нас есть q^* , которое мы можем сэмплировать и про которое мы знаем константу c , такую, что

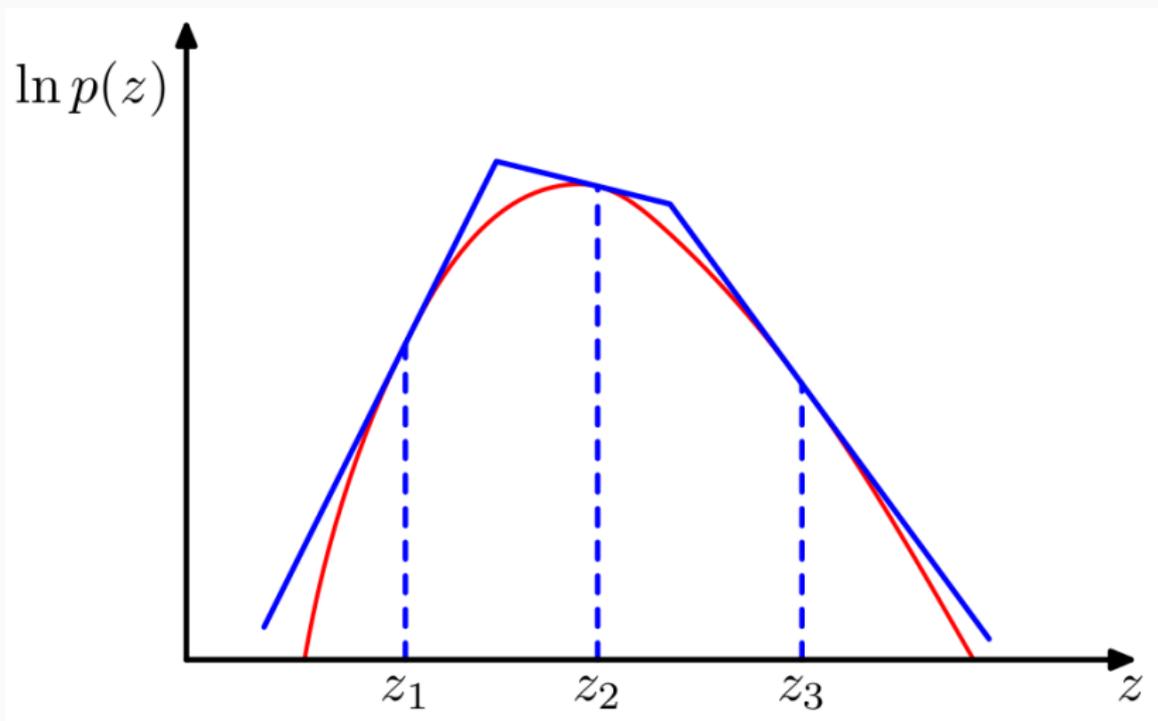
$$\forall x \quad cq^*(x) > p^*(x).$$

- Тогда мы сумеем сэмплировать p .

- Взять сэмпл x по распределению $q^*(x)$.
- Выбрать случайное число u равномерно из интервала $[0, cq^*(x)]$.
- Вычислить $p^*(x)$. Если $u > p^*(x)$, x отклоняется (отсюда и название), иначе добавляется в сэмплы.



- Алгоритм работает, потому что выбирает точки $[x, u]$ равномерно из области под графиком $p^*(x)$, а это и значит, что получатся сэмплы p^* .
- Вариант – *адаптивная выборка*: если мы можем точнее определить $q(x)$, например построить её как многогранник, касающийся выпуклой (как правило, лог-выпуклой – и многогранник в логарифмическом пространстве) плотности распределения.



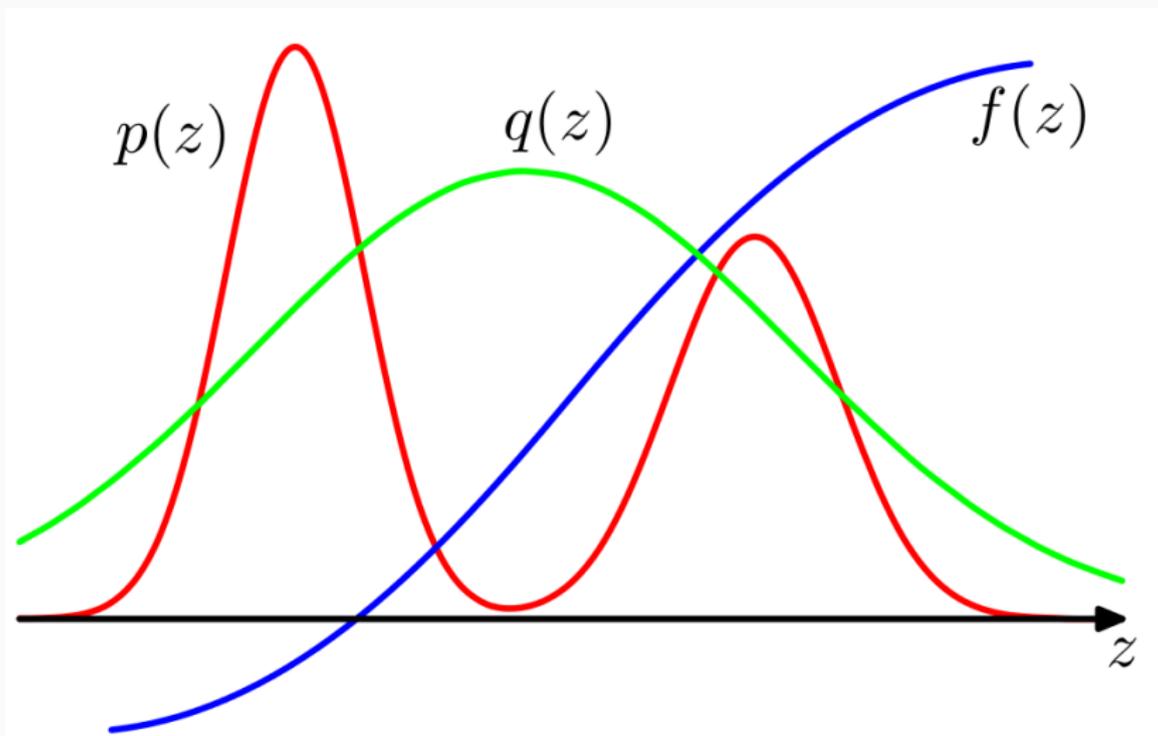
СЭМПЛИРОВАНИЕ ПО ЗНАЧИМОСТИ

- Вариант выборки с отклонением можно применить к направленным графическим моделям.
- Сэмплировать без evidence – тривиально.
- Сэмплировать с evidence можно так: сделаем сэмпл, если наблюдаемые переменные не сошлись, выкинем.
- Для ненаправленных не так просто, да и для направленных не сработает, если наблюдаемых много.

- Как и у предыдущего алгоритма, у выборки с отклонением начинаются проблемы в больших размерностях.
- Суть проблемы та же, что в предыдущем случае, а выражается она в том, что c будет очень большим (экспоненциальным от n), и почти все сэмплы будут отвергаться.

- Выборка по значимости — importance sampling.
- Мы решаем только вторую задачу, а не первую.
- То есть нам нужно брать сэмплы, при этом желательно попадая в зоны, где функция p^* имеет большие значения.

- Предположим, что у нас есть какое-то другое распределение вероятностей q (точнее, q^*), попроще, и мы умеем брать его сэмплы.
- Тогда алгоритм такой: сначала взять выборку по q^* , а затем перевзвесить её так, чтобы получилась всё-таки выборка по p^* .



- Мы хотим

$$\begin{aligned} E[f] &= \int f(\mathbf{x})p(\mathbf{x})d\mathbf{x} = \int f(\mathbf{x})\frac{p(\mathbf{x})}{q(\mathbf{x})}q(\mathbf{x})d\mathbf{x} = \\ &\approx \frac{1}{L} \sum_r \frac{p(\mathbf{x}^{(r)})}{q(\mathbf{x}^{(r)})} f(\mathbf{x}^{(r)}). \end{aligned}$$

- $w_r = p(\mathbf{x}^{(r)})/q(\mathbf{x}^{(r)})$ – веса, с которыми входят сэмплы, но все сэмплы остаются в множестве.

- Если у нас не p и q , а p^* и q^* , и $p = \frac{1}{Z_p} p^*$, $q = \frac{1}{Z_q} q^*$, то

$$\begin{aligned} E[f] &= \int f(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} = \frac{Z_q}{Z_p} \int f(\mathbf{x}) \frac{p^*(\mathbf{x})}{q^*(\mathbf{x})} q(\mathbf{x}) d\mathbf{x} \approx \\ &\approx \frac{Z_q}{Z_p} \frac{1}{R} \sum_{r=1}^R \frac{p^*(\mathbf{x}^{(r)})}{q^*(\mathbf{x}^{(r)})} f(\mathbf{x}^{(r)}), \end{aligned}$$

и Z_q/Z_p можно оценить из тех же сэмплов:

$$\frac{Z_p}{Z_q} = \frac{1}{Z_q} \int p^*(\mathbf{x}) d\mathbf{x} = \int \frac{p^*(\mathbf{x})}{q^*(\mathbf{x})} q(\mathbf{x}) d\mathbf{x} \approx \frac{1}{R} \sum_{r=1}^R \frac{p^*(\mathbf{x}^{(r)})}{q^*(\mathbf{x}^{(r)})}.$$

• Получаем такой алгоритм:

1. Взять сэмплы $\{x^{(r)}\}_{r=1}^R$ по распределению q^* .
2. Рассчитать веса

$$w_r = \frac{p^*(\mathbf{x}^{(r)})/q^*(\mathbf{x}^{(r)})}{\sum_m p^*(\mathbf{x}^{(m)})/q^*(\mathbf{x}^{(m)})}.$$

3. Оценить функцию по формуле

$$\mathbb{E}[f] \approx \frac{1}{R} \sum_{r=1}^R w_r f(\mathbf{x}^{(r)}).$$

- Зачем нужно q ? Чем это лучше равномерного распределения?

- Зачем нужно q ? Чем это лучше равномерного распределения?
- Проще говоря, распределение q должно помочь выбрать те участки, на которых имеет смысл сэмплить r .
- Если q хорошее, то может помочь, а если плохое, может только навредить.
- Но есть и более фундаментальные проблемы.

- Во-первых, сэмплер q не должен быть слишком узким.
- Например, если сэмплер гауссиановский с небольшой вариацией, то пики r далеко от центра q вообще никто не заметит.

- Во-вторых, может случиться, что все сэмплы будут напрочь убиты небольшим количеством сэмплов с огромными весами. Это плохо.
- Чтобы показать, как это бывает, давайте перейдём в многомерный случай.

- Пусть есть равномерное распределение r на единичном шаре и сэмплер q — произведение гауссианов с центром в нуле:

$$p(x) = \frac{1}{(2\pi\sigma^2)^{N/2}} e^{-\frac{1}{2\sigma^2} \sum_i x_i^2}.$$

Упражнение. Найдите среднее и дисперсию расстояния $r^2 = \sum_i x_i^2$ точки, взятой по этому распределению.

- Ответ на упражнение: расстояние будет $N\sigma^2 \pm \sqrt{2N}\sigma^2$ (распределение будет похоже на гауссовское).
- Значит, почти все сэмплы лежат в «типичном множестве», кольце расстоянием около $\sigma\sqrt{N}$ от нуля.

- Тогда большинство сэмплов q будут лежать в интервале

$$\frac{1}{(2\pi\sigma^2)^{n/2}} 2^{-\frac{N}{2} \pm \frac{\sqrt{2N}}{2}},$$

и ненулевые веса будут иметь значения порядка

$$(2\pi\sigma^2)^{n/2} 2^{\frac{N}{2} \pm \frac{\sqrt{2N}}{2}}.$$

- Это значит, что максимальный вес будет относиться к среднему примерно как $2^{\sqrt{2N}}$, а это очень много.

- Варианты выборки по значимости для направленных графических моделей:
 - uniform sampling – фиксируем evidence, выбираем остальные равномерно, вес у сэмпла получается просто $p(\mathbf{x})$, потому что он автоматически сходится с evidence;
 - likelihood weighted sampling – фиксируем evidence, выбираем остальные от родителей к детям из условного распределения $p(x_i \mid \text{pa}(x_i))$, где $\text{pa}(x_i)$ уже зафиксированы; вес тогда будет

$$r(\mathbf{x}) = \prod_{x_i \notin E} \frac{p(x_i \mid \text{pa}(x_i))}{p(x_i \mid \text{pa}(x_i))} \prod_{x_i \in E} \frac{p(x_i \mid \text{pa}(x_i))}{1} = \prod_{x_i \in E} p(x_i \mid \text{pa}(x_i)).$$

- Если размерность большая, то у выборки по значимости есть две большие проблемы.
- Во-первых, чтобы получить разумные сэмплы, нужно уже заранее выбрать q так, чтобы оно хорошо аппроксимировало p .
- Во-вторых, даже если их получить, часто может так случиться, что веса у некоторых сэмплов будут слишком велики.
- В общем, для случая многих размерностей это не очень хороший метод.

СПАСИБО!

Спасибо за внимание!

