VOLGOGRAD STATE TECHNICAL UNIVERSITY

Машинное обучение и нейросетевые модели

Лекция 3. Инференс в байесовских моделях с помощью MCMC

Лектор: Кравченя Павел Дмитриевич

Волгоград 2025

VOLGOGRAD STATE TECHNICAL UNIVERSITY

План лекции

1. Проблема байесовского вывода. Понятие МСМС.
2. Представление марковской цепи. Вероятности переходов.
3. Предельное распределение марковской цепи.
4. Свойства несводимости, периодичности и стационарности цепи.
5. Фундаментальная теорема марковских цепей.
6. Обратимость марковской цепи. Условия детального баланса.
7. Алгоритм Метрополиса-Гастингса. Сходимость к целевому распределению.
8. Этап «розжига» и корреляция семплов в алгоритме Метрополиса-Гастингса.
9. Ограничения алгоритма Метрополиса-Гастингса. Алгоритм Гиббса.
10. Метрики качества семплирования в цепях Монте-Карло.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Проблема байесовского вывода

- В задачах *байесовского моделирования* приходится часто решать задачу **инференса**:

$$p(\mathbf{w} \mid \mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y} \mid \mathbf{X}, \mathbf{w})p(\mathbf{w})}{\int p(\mathbf{y} \mid \mathbf{X}, \mathbf{w})p(\mathbf{w})d\mathbf{w}}$$

в которой расчет значения интеграла в знаменателе представляет собой нетривиальную задачу (т.н. *intractable* интеграл).

- *Аналитическое* интегрирование возможно только с случае сопряженных распределений.
- *Численное* интегрирование невозможно из-за *большой размерности* интегрируемой функции и её *сложного поведения*.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

- Задачи оценки таких интегралов эффективно решаются методами Монте-Карло (с применением *эстиматора Монте-Карло*):

$$\hat{F}_N = \frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)}{p(x_i)}, \qquad x_i \sim p(x).$$

- Но *возможность применения* методов Монте-Карло ограничена *возможностью* и *эффективностью* семплирования из $p(x)$.
- В случае *низких* размерностей успешно применяются различные алгоритмы семплирования (inverse transform, rejection, importance sampling).
- В случае *больших размерностей* и *сложных функций* стандартом для проведения семплирования стали **марковские цепи Монте-Карло** (*Markov chains Monte Carlo, MCMC*).

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Понятие марковской цепи. Свойство Маркова.

- **Цепь Маркова** (англ. *Markov chain*) – последовательность (процесс) *случайных событий* X_t с конечным или счётным числом исходов, в которой вероятность наступления каждого события зависит только от состояния, достигнутого в предыдущем событии.
- Данное свойство называется **свойством Маркова**:

$$\mathrm{P}(X_{t+1} \mid X_t, X_{t-1}, \dots X_0) = \mathrm{P}(X_{t+1} \mid X_t), \qquad \forall t \geq 0.$$

- Процесс в каждый момент времени находится в одном из n состояний.
- Марковские цепи называются *однородными во времени*, если вероятности перехода между состояниями *не зависят от времени*:

$$\mathrm{P}(X_{t+1} \mid X_t) = \mathrm{P}(X_1 \mid X_0), \qquad \forall t \geq 0.$$

VOLGOGRAD STATE TECHNICAL UNIVERSITY

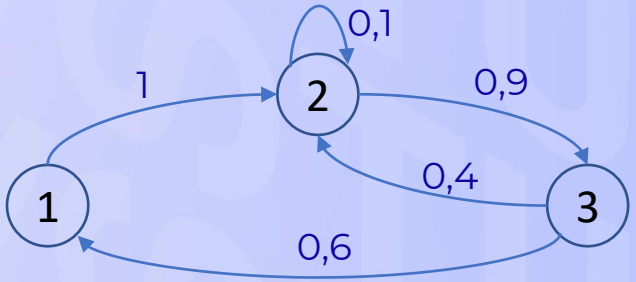
Представление марковской цепи. Вероятности переходов

- Если процесс находится в состоянии с номером i, то он перейдет в состояние j с *вероятностью* t_{ij}.
- Матрица $\mathbf{T} = [T_{ij}]$ называется матрицей переходов (англ. *Transition matrix*):

$$T_{ij} = \mathrm{P}(X_t = j \mid X_{t-1} = i), \qquad T_{ij} \geq 0, \qquad \forall i \in [0..n] \sum_{j=1}^{n} T_{ij} = 1.$$

- Марковскую цепь можно представить в виде **графа**, в котором *вершины* представляют *состояния* процесса, а *ребра* — *переходы* между состояниями с соответствующими *вероятностями*.
- Например:

$$\mathbf{T} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0{,}1 & 0{,}9 \\ 0{,}6 & 0{,}4 & 0 \end{pmatrix}$$



VOLGOGRAD STATE TECHNICAL UNIVERSITY

Состояние марковской цепи после многих временных шагов

- В каком состоянии окажутся случайные величины в цепи Маркова после многих временных шагов?
- Введем *вероятность* случайной величины оказаться в состоянии j в момент времени t: $\pi_t(j) = \mathrm{P}(X_t = j)$.
- С учетом того, что состояние X_{t-1} может быть *различным*, эта вероятность:

$$\pi_t(j) = \mathrm{P}(X_t = j) = \sum_{i=1}^{n} \mathrm{P}(X_t = j \mid X_{t-1} = i) \cdot \mathrm{P}(X_{t-1} = i) = \sum_{i=1}^{n} t_{ij}\, \pi_{t-1}(i).$$

- Пусть:

$$\boldsymbol{\pi}_t = \begin{bmatrix} \mathrm{P}(X_t = 1) \\ \mathrm{P}(X_t = 2) \\ \cdots \\ \mathrm{P}(X_t = n) \end{bmatrix}^{\mathrm{T}} \quad \Rightarrow \quad \boldsymbol{\pi}_t = \boldsymbol{\pi}_{t-1}\mathbf{T} \quad \Rightarrow \quad \boldsymbol{\pi}_t = \boldsymbol{\pi}_0\mathbf{T}^t.$$

VOLGOGRAD STATE TECHNICAL UNIVERSITY

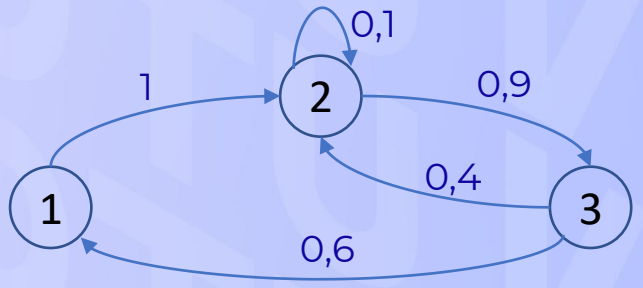
Предельное распределение марковской цепи

- Переходим к пределу для бесконечного числа временных шагов:

$$\boldsymbol{\pi}_\infty = \lim_{t\to\infty} \boldsymbol{\pi}_0 \mathbf{T}^t .$$

- Данный вектор вероятностей называется **предельным распределением** (англ. *Limit distribution*) Марковской цепи.
- Существует ли он? Единственный ли он?
- Например, для рассматриваемого примера:

$$\boldsymbol{\pi}_\mathrm{t} = \boldsymbol{\pi}_0 \mathbf{T}^t = \boldsymbol{\pi}_0 \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0{,}1 & 0{,}9 \\ 0{,}6 & 0{,}4 & 0 \end{pmatrix}^t = ?$$



VOLGOGRAD STATE TECHNICAL UNIVERSITY

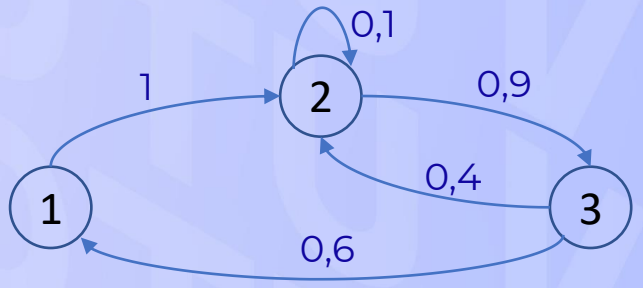
Предельное распределение марковской цепи

- Переходим к пределу для бесконечного числа временных шагов:

$$\boldsymbol{\pi}_\infty = \lim_{t\to\infty} \boldsymbol{\pi}_0 \mathbf{T}^t .$$

- Данный вектор вероятностей называется **предельным распределением** (англ. *Limit distribution*) Марковской цепи.
- Существует ли он? Единственный ли он?
- Например, для рассматриваемого примера:

$$\boldsymbol{\pi}_\mathrm{t} = \boldsymbol{\pi}_0 \mathbf{T}^t = \boldsymbol{\pi}_0 \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0{,}1 & 0{,}9 \\ 0{,}6 & 0{,}4 & 0 \end{pmatrix}^t \approx \begin{pmatrix} 0{,}221 & 0{,}409 & 0{,}368 \end{pmatrix}$$



он существует, и даже не зависит от $\boldsymbol{\pi}_0$!

- Каковы же *критерии* его *существования* и *единственности*?

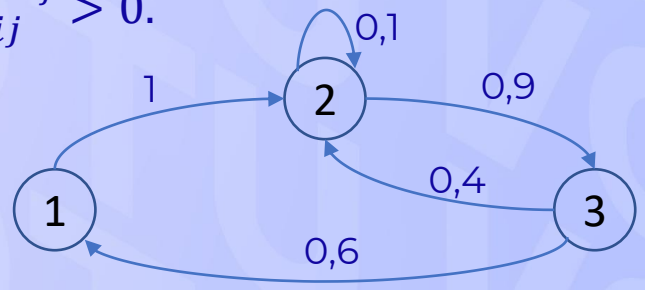
VOLGOGRAD STATE TECHNICAL UNIVERSITY

Свойство несводимости (Irreducibility)

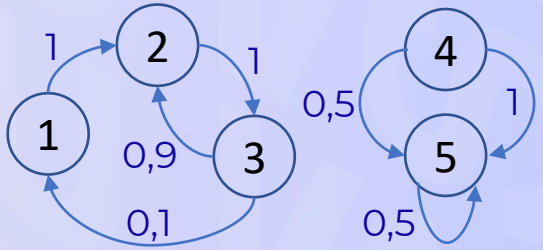
- Марковская цепь называется **несводимой** (irreducible), если каждое состояние *достижимо* из другого за конечное количество временных шагов:

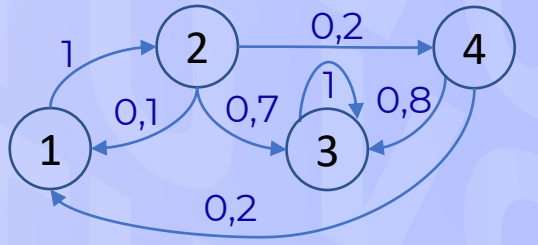
$$\forall i, j \in [1..n] \ \exists m_{ij} \in \mathbb{N} : \mathrm{P}\left(X_{t+m_{ij}} = j \,\middle|\, X_t = i\right) = p_{ij}^{m_{ij}} > 0.$$

- Рассмотренная ранее цепь Маркова является *несводимой*:



- Примеры цепей Маркова, не являющихся *несводимыми*:





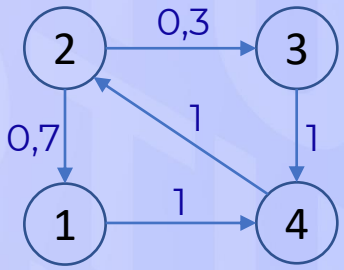
VOLGOGRAD STATE TECHNICAL UNIVERSITY

Свойство периодичности (Periodicity)

- В некоторых марковских цепях может происходить периодический возврат к какому-либо состоянию. В этом случае марковская цепь считается удовлетворяющей свойству **периодичности** (Periodicity).
- Состояние i марковской цепи имеет **период** m_i, если любой *возврат в состояние i* должен произойти за m_i временных шагов:

$$m_i = \text{НОД}\,[k \in \mathbb{N} : \mathrm{P}(X_{t+k} = i \mid X_t = i) > 0].$$

- Если период *всех состояний* марковской цепи равен *единице*, цепь называется **апериодической**.
- Пример *периодической* марковской цепи:



VOLGOGRAD STATE TECHNICAL UNIVERSITY

Свойство стационарности (Stationary). Состояние равновесия

- Вектор вероятностей $\boldsymbol{\pi}$ называется **стационарным распределением** марковской цепи с матрицей переходов $\mathbf{T}$, если справедливо равенство:

$$\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{T}.$$

- Если для некоторого временного шага c *вероятности* переходов представлены вектором $\boldsymbol{\pi}$, то $\boldsymbol{\pi}_t = \boldsymbol{\pi}$ для $\forall t > c$. В этом случае считается, что цепь Маркова достигла **устойчивого состояния** или **равновесия**.
- При стационарном распределении для любого состояния j:

$$\pi(j) = \sum_{i \neq j} \pi(i) T_{ij}\,.$$

- Данные *соотношения* называются **уравнениями глобального баланса**.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Фундаментальная теорема марковских цепей

- Если марковская цепь является *несводимой* и *апериодической*, то:
  1. $\exists! \boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_n)$ – стационарное распределение цепи;
  2. $\forall i \in [1..n] \quad \lim\limits_{t \to \infty} \mathrm{P}(X_i = i) = \pi_i$
- Т.е., цепь Маркова будет *сходиться* к единственному стационарному распределению.
- Поэтому, чтобы вместо того, чтобы семплировать непосредственно из распределения $p(x)$, можно создать *несводимую* и *апериодическую* марковскую цепь со *стационарным* распределением, совпадающим с $p(x)$, и получать реализации состояний после достижения цепью *равновесия*.
- Как гарантировать, что *именно целевое распределение* $p(x)$ является *стационарным* распределением марковской цепи?

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Обратимость марковской цепи (Reversibility). Условия детального баланса

- Марковская цепь называется обратимой относительно **π**, если существует распределение вероятностей **π** по её состояниям такое, что:

$$\forall i, j \in [1..n] \qquad \pi(i)T_{ij} = \pi(j)T_{ji}$$
$$\forall x, y \in \Omega \qquad \pi(x)K(y \mid x) = \pi(y)K(x \mid y)$$

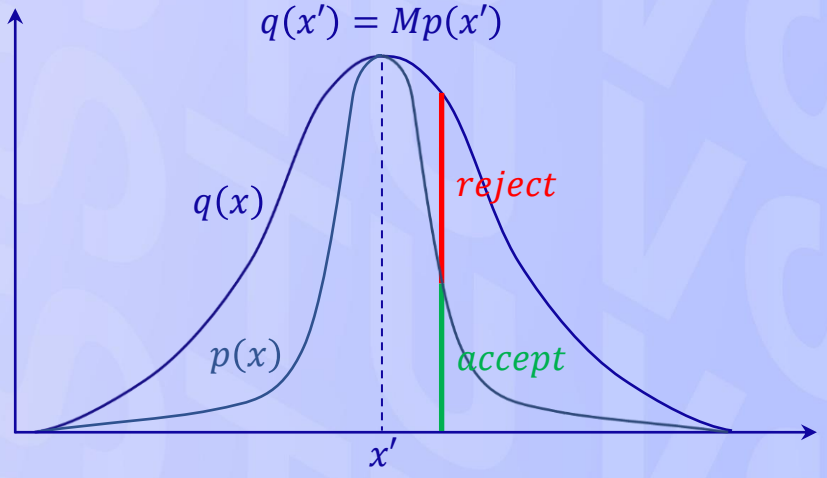
Количество «вероятностной массы», перетекающей между x и y

- Данное соотношение называется **условием детального баланса**.
- **Достаточным** (но не необходимым) **условием** того, чтобы **π** было стационарным распределением цепи Маркова с матрицей перехода **T**, является условие детального баланса.
- Поэтому, если *получится построить* неприводимую, апериодическую и обратимую относительно $p(x)$ цепь Маркова , то она будет сходиться к $p(x)$.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Подход семплирования с отклонением

- Вспомним *подход* семплирования с отклонением (rejection sampling):
  - ✓ Для *target distribution* $p(x)$ вводим *proposal distribution* $q(x)$.
  - ✓ *Масштабируем* это распределение: $Mq(x)$ – оно должно быть больше целевого в каждой точке.
  - ✓ For $i = 1..S$:
    - Семплируем $y^* \sim Mq(x)$.
    - Семплируем $u \sim \mathcal{U}(0,1)$.
    - Если $u < \frac{p(y^*)}{Mq(y^*)}$, то принимаем семпл, иначе – отвергаем.

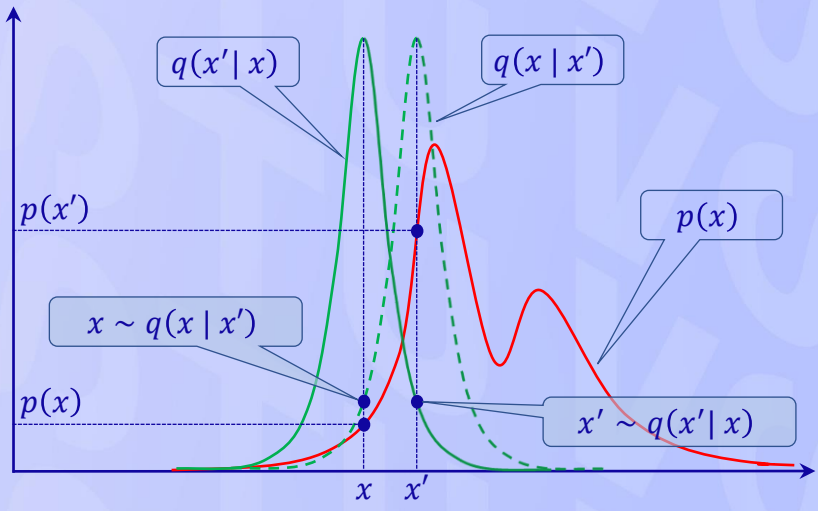


- Идея: давайте сделаем *proposal distribution* динамическим, зависящим от *предыдущего семпла*.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

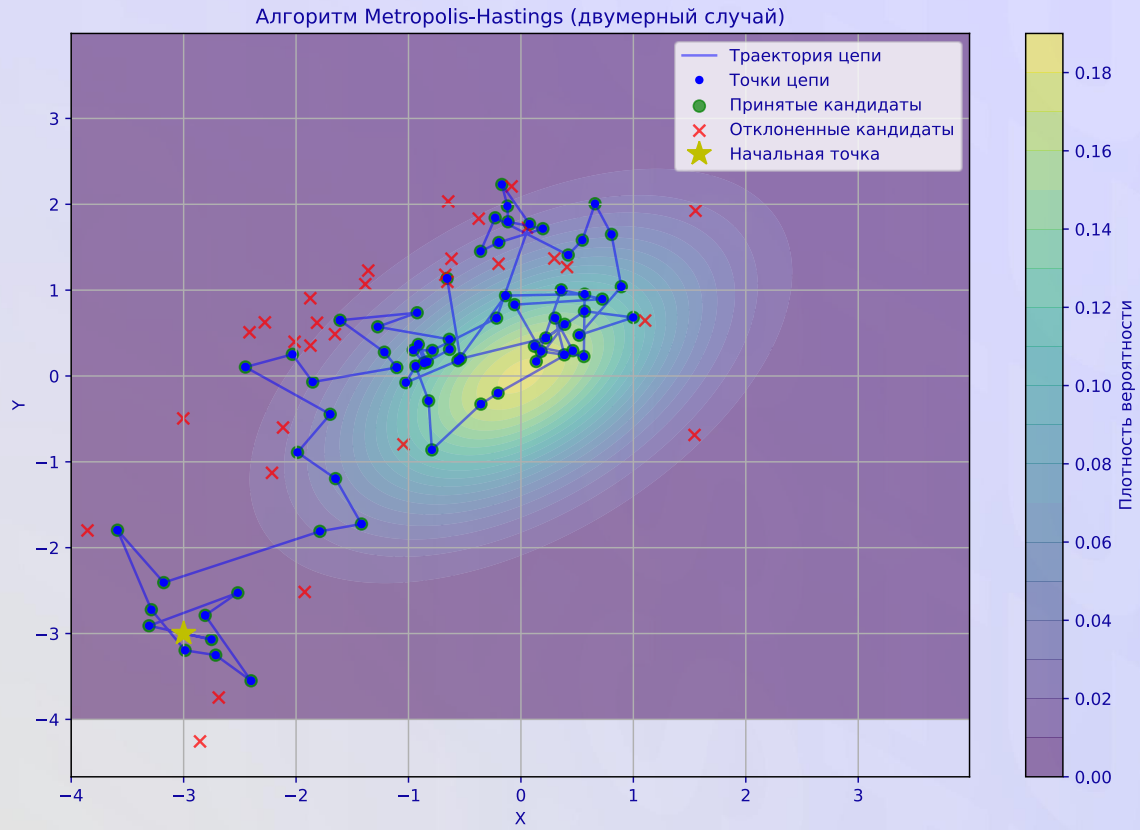
Алгоритм Метрополиса-Гастингса (Metropolis-Hastings)

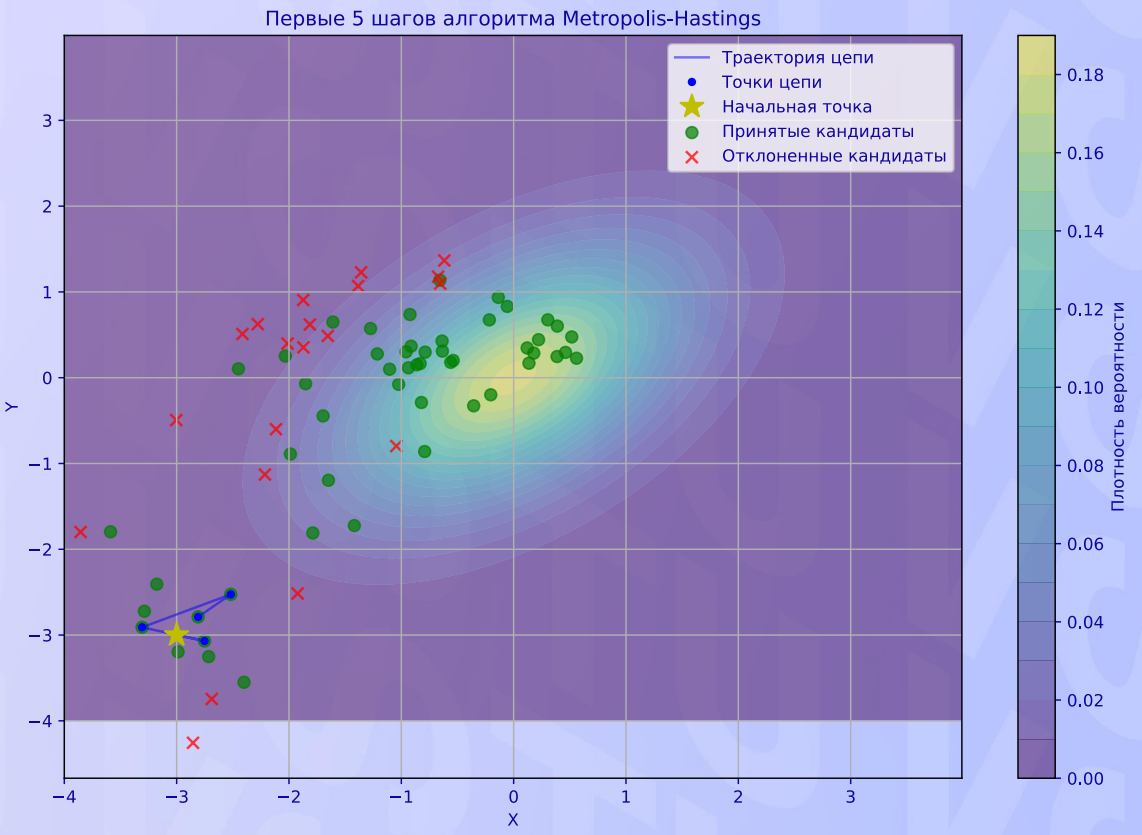
- Алгоритм впервые был опубликован *Н. Метрополисом* в 1953 году, а затем обобщён *К. Гастингсом* в 1970 году.
- Вводим *proposal distribution* $q(x|y)$.
- Инициализируем $x_0 = random$ (часто – из *априорного* распределения $p(x)$).
- For $i = 1..S$:
  - Семплируем $u \sim \mathcal{U}(0,1)$.
  - Семплируем $x^* \sim q(x^*, x_{i-1})$.
  - Вычисляем $\mathcal{A}(x^*, x_{i-1}) = \min\left\{1, \frac{p(x^*)\cdot q(x_{i-1} \mid x^*)}{p(x_{i-1})\cdot q(x^* \mid x_{i-1})}\right\}$.
  - Если $u \le \mathcal{A}(x^*, x_{i-1})$, то $\boxed{x_i = x^*}$
    иначе: $\boxed{x_i = x_{i-1}}$
- $\mathcal{A}(x^*, x_{i-1})$ называется *вероятностью* принятия.



VOLGOGRAD STATE TECHNICAL UNIVERSITY

Визуализация алгоритма Метрополиса-Гастингса





VOLGOGRAD STATE TECHNICAL UNIVERSITY

Алгоритм Метрополиса-Гастингса как марковская цепь

- *Вероятности переходов* между состояниями в алгоритме Метрополиса-Гастингса представляются ядром переходов (*transition kernel*) $K(X_t \mid X_{t-1})$.
- Алгоритм является **марковской цепью** (выполняется *свойство Маркова*).
- Цепь является **несводимой** (irreducible), поскольку на каждом шаге есть положительная *вероятность достичь* любого состояния (из возможных).
- Для некоторого состояния x^* вероятность принятия:
  - ✓ меньше единицы, значит, цепочка останется в x^* с ненулевой вероятностью. Следовательно, ее период равен 1.
  - ✓ равна единице, но в таком случае вероятность вернуться в состояние x^* из последующего отлична от единицы. Значит, период вновь равен 1.
- Таким образом, цепь является **апериодической**.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Сходимость алгоритма Метрополиса-Гастингса к целевому распределению

- *Несводимая апериодическая* цепь Маркова сходится к стационарному распределению. Для того, чтобы это распределение совпадало с целевым, необходимо выполнение *уравнения детального баланса* относительно $p(x)$:

$$p(x_{i-1})K(x_i \mid x_{i-1}) = p(x_i)K(x_{i-1} \mid x_i).$$

- Проверим это:
  - ✓ Случай $x_i = x_{i-1}$: $p(x_i)K(x_i \mid x_i) = p(x_i)K(x_i \mid x_i)$. Равенство *очевидно*.
  - ✓ Случай $x_i \neq x_{i-1}$. Рассмотрим поочередно *левую* и *правую* части УДБ:

$$p(x_{i-1})K(x_i \mid x_{i-1}) = p(x_{i-1})q(x_i \mid x_{i-1})\mathcal{A}(x_i, x_{i-1}) = p(x_{i-1})q(x_i \mid x_{i-1}) \cdot \min\left\{1, \frac{p(x_i)q(x_{i-1} \mid x_i)}{p(x_{i-1})q(x_i \mid x_{i-1})}\right\} =$$

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Сходимость алгоритма Метрополиса-Гастингса к целевому распределению

$$= \begin{cases} p(x_{i-1})q(x_i \mid x_{i-1}) & if\ p(x_i)q(x_{i-1} \mid x_i) > p(x_{i-1})q(x_i \mid x_{i-1}) \\ p(x_{i-1})q(x_i \mid x_{i-1}) \cdot \dfrac{p(x_i)q(x_{i-1} \mid x_i)}{p(x_{i-1})q(x_i \mid x_{i-1})} & if\ p(x_i)q(x_{i-1} \mid x_i) < p(x_{i-1})q(x_i \mid x_{i-1}) \end{cases} =$$

$$= \begin{cases} p(x_{i-1})q(x_i \mid x_{i-1}) & if\ p(x_{i-1})q(x_i \mid x_{i-1}) < p(x_i)q(x_{i-1} \mid x_i) \\ p(x_i)q(x_{i-1} \mid x_i) & if\ p(x_i)q(x_{i-1} \mid x_i) < p(x_{i-1})q(x_i \mid x_{i-1}) \end{cases} =$$

$$= \boxed{\min\{p(x_{i-1})q(x_i \mid x_{i-1}), p(x_i)q(x_{i-1} \mid x_i)\}}$$

- Аналогично рассмотрим правую часть УДБ:

$$p(x_i)K(x_{i-1} \mid x_i) = p(x_i)q(x_{i-1} \mid x_i)\mathcal{A}(x_{i-1}, x_i) = p(x_i)q(x_{i-1} \mid x_i) \cdot \min\left\{1, \frac{p(x_{i-1})q(x_i \mid x_{i-1})}{p(x_i)q(x_{i-1} \mid x_i)}\right\} =$$

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Сходимость алгоритма Метрополиса-Гастингса к целевому распределению

$$= \begin{cases} p(x_i)q(x_{i-1} \mid x_i) & if\ p(x_{i-1})q(x_i \mid x_{i-1}) > p(x_i)q(x_{i-1} \mid x_i) \\ p(x_i)q(x_{i-1} \mid x_i) \cdot \dfrac{p(x_{i-1})q(x_i \mid x_{i-1})}{p(x_i)q(x_{i-1} \mid x_i)} & if\ p(x_{i-1})q(x_i \mid x_{i-1}) < p(x_i)q(x_{i-1} \mid x_i) \end{cases} =$$

$$= \begin{cases} p(x_i)q(x_{i-1} \mid x_i) & if\ p(x_i)q(x_{i-1} \mid x_i) < p(x_{i-1})q(x_i \mid x_{i-1}) \\ p(x_{i-1})q(x_i \mid x_{i-1}) & if\ p(x_{i-1})q(x_i \mid x_{i-1}) < p(x_i)q(x_{i-1} \mid x_i) \end{cases} =$$

$$= \boxed{\min\{p(x_i)q(x_{i-1} \mid x_i), p(x_{i-1})q(x_i \mid x_{i-1})\}}$$

- Равенства совпадают, что свидетельствует о выполнении уравнения детального баланса и для случая $x_i \neq x_{i-1}$.
- Следовательно, цепь Маркова, порождаемая алгоритмом Метрополиса-Гастингса, **сходится к целевому распределению** $p(x)$.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Обработка нормирующей константы в алгоритме Метрополиса-Гастингса

- Для расчета вероятности принятия нужно уметь вычислять $p(x)$, однако, она определена с *точностью до неизвестной константы*:

$$p(\boldsymbol{w}|\boldsymbol{\mathcal{D}}) = \frac{p(\boldsymbol{\mathcal{D}}|\boldsymbol{w})p(\boldsymbol{w})}{\int p(\boldsymbol{\mathcal{D}}|\boldsymbol{w})p(\boldsymbol{x})} = \frac{1}{Z} \cdot p(\boldsymbol{\mathcal{D}}|\boldsymbol{w})p(\boldsymbol{\mathcal{D}})$$

- Ситуацию спасает то, что при расчете вероятности принятия семплов требуется считать не само значение $p(x)$, а отношение:

$$\frac{p(\mathbf{w}^*|\boldsymbol{\mathcal{D}})}{p(\mathbf{w}_{i-1}|\boldsymbol{\mathcal{D}})} = \frac{\frac{p(\boldsymbol{\mathcal{D}}|\mathbf{w}^*)p(\mathbf{w}^*)}{p(\boldsymbol{\mathcal{D}})}}{\frac{p(\boldsymbol{\mathcal{D}}|\mathbf{w}_{i-1})p(\mathbf{w}_{i-1})}{p(\boldsymbol{\mathcal{D}})}} = \frac{p(\boldsymbol{\mathcal{D}}|\mathbf{w}^*)p(\mathbf{w}^*)}{p(\boldsymbol{\mathcal{D}}|\mathbf{w}_{i-1})p(\mathbf{w}_{i-1})} = \frac{p(\mathbf{w}^*, \boldsymbol{\mathcal{D}})}{p(\mathbf{w}_{i-1}, \boldsymbol{\mathcal{D}})}.$$

- А само отношение **не зависит** от *неизвестной нормирующей константы*.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Этап «розжига» алгоритма Метрополиса-Гастингса

- Алгоритм Метрополиса-Гастингса начинает свою работу с произвольно сгенерированного значения x, и ему требуется время, чтобы прийти к *стационарному распределению*.
- Особенно ярко эта ситуация проявляется в случае расположения начального значения x *в маловероятной области* целевого распределения.
- До этого момента семплы, генерируемые алгоритмом, *отличаются от целевого распределения* и должны быть *отброшены*.
- Этап работы алгоритма *до выхода на стационарное распределение* носит название «Burn-in» режима («розжиг» семплера).
- Длительность «Burn-in» режима зависит от *особенностей целевого распределения*. В процессе моделирования её часто подбирают *вручную*.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Корреляция семплов в алгоритме Метрополиса-Гастингса

- В процессе семплирования *в соответствии с алгоритмом* Метрополиса-Гастингса возникает проблема корреляции соседних семплов.
- В соответствии *с заданным ядром перехода* $K(x_i \mid x_{i-1})$ текущий семпл зависит от предыдущего.
- Однако, все оценки в методах Монте-Карло выводятся в предположении независимости семплов друг от друга. Наличие такой зависимости снижает точность их работы.
- Для *смягчения зависимости* формируются R семплов (*lag*), но используется только последний из них. При достаточно *больших* R семплы можно считать *относительно независимыми*.
- Значение R зависит от распределений *target* и *proposal*.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Ограничения алгоритма Метрополиса-Гастингса

- Алгоритм предполагает случайное семплирование с использованием *proposal distribution*, информация о поведении *целевого распределения* не учитывается.
- В силу случайного «блуждания» при реализации случайного семплирования вероятность *значительно удалиться* от текущего положения незначительна, что снижает «*разведочную силу*» алгоритма.
- Требуется правильно подобрать *proposal distribution* для эффективной работы алгоритма. Часто выбирают $q(x_i \mid x_{i-1}) = \mathcal{N}(x_{i-1}, \sigma^2)$.
- При этом, если дисперсия σ^2 *маленькая*, то доля принятия семплов *большая*, но «разведка» целевого распределения *очень медленная*. И *наоборот*.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

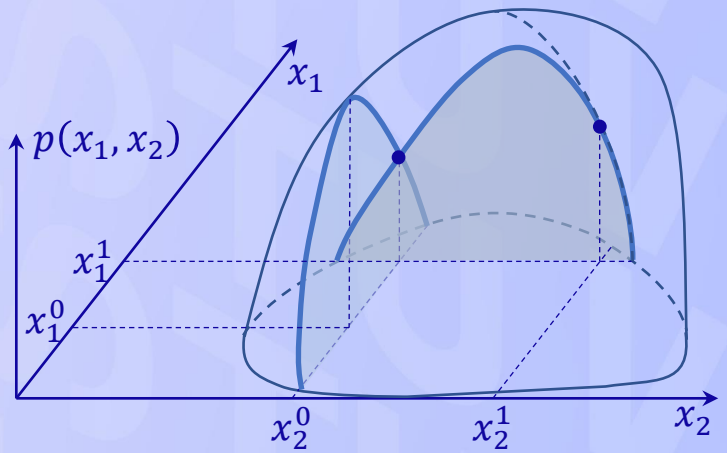
Важный частный случай алгоритма Метрополиса-Гастингса

- Алгоритм Метрополиса-Гастингса *положил начало* *большому количеству алгоритмов*, которые унаследовали от него важные особенности.
- Одним из его частных случаев является алгоритм семплирования Гиббса, который предполагает выбор в качестве *proposal условных распределений переменных* модели по другим переменным.
- В этом случае, *вероятность принятия всегда равна 1 (нет отклонений)*, что повышает эффективность.
- Алгоритм Гиббса прост в реализации и быстрее сходится для *простых условных распределений* и параметров со *слабой корреляцией*.
- *Основное требование*: Условные распределения *известны* и из них *легко семплировать*.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Алгоритм семплирования Гиббса

- Алгоритм семплирования Гиббса применяется в случаях, когда *прямое семплирование* из совместного распределения $p(\mathbf{x})$ затруднено, но вероятностная модель позволяет *семплировать* из *одномерных условных распределений*: $p(x_i \mid \{x_{-i}\})$, где $\{x_{-i}\} = \{x_j : x_j \neq x_i \ \forall x_j \in \mathbf{x}\}$.
- Инициализируем $\mathbf{x}^0 = random\ (x_1^0, x_2^0, \dots, x_m^0)$.
- For $t = 0..S$ получаем семплы:
  - $x_1^{t+1} \sim p(x_1 \mid x_2^t, x_3^t, \dots, x_m^t)$;
  - $x_2^{t+1} \sim p(x_2 \mid x_1^{t+1}, x_3^t, \dots, x_m^t)$;
  - ...
  - $x_k^{t+1} \sim p\left(x_k \mid x_1^{t+1}, x_2^{t+1}, \dots, x_{k-1}^{t+1}, x_{k+1}^t, \dots, x_m^t\right)$;
  - $x_m^{t+1} \sim p(x_m \mid x_1^{t+1}, x_2^{t+1}, \dots, x_{m-1}^{t+1})$.



VOLGOGRAD STATE TECHNICAL UNIVERSITY

Сравнение алгоритмов семплирования Метрополиса-Гастингса и Гиббса

Реальные практические проблемы	Алгоритм Метрополиса-Гастингса	Алгоритм Гиббса
Условные распределения неизвестны	Подходит, так как не требует условных распределений	Не подходит, так как требует знания условных распределений
Сильная корреляция между переменными	Может справиться, если *proposal* хорошо настроен	Плохо работает, так как цепь медленно перемешивается
Мультимодальное распределение	Может исследовать разные моды при подходящей функции предложения	Часто застревает в одной моде
Высокая размерность	Может быть медленным, но гибкость позволяет адаптироваться	Может быть неэффективным из-за сложности условных распределений
Простые условные распределения	Может быть избыточным, но работает	Предпочтителен, так как эффективен и прост в реализации

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Применение нескольких цепей Монте-Карло

- На практике часто запускают *несколько независимых цепей* (*chains*) с разными начальными точками. Это делается по следующим причинам:
  - ✓ *Проверка сходимости*. Если все цепи сходятся к одному и тому же распределению (например, их выборки имеют схожие статистики), это свидетельствует о том, что МСМС достиг целевого распределения. Если же цепи не сходятся, это может указывать на проблемы, такие как плохая сходимость, мультимодальность распределения или застревание в локальных максимумах.
  - ✓ *Ускорение вычислений*. Запуск нескольких цепей параллельно позволяет быстрее исследовать пространство параметров и получить больше выборок за меньшее время. Цепи должны быть независимыми, чтобы избежать ложной уверенности в сходимости.
  - ✓ *Диагностика проблем*. Сравнение цепей помогает выявить проблемы с алгоритмом.
- Процесс семплирования *с несколькими цепями* обычно проводится так:
  - ✓ запускаются несколько цепей МСМС с разными начальными значениями;
  - ✓ каждая цепь генерирует выборки, которые постепенно приближаются к целевому апостериорному распределению;
  - ✓ после периода прогрева анализируются семплы из всех цепей и проверяется сходимость.

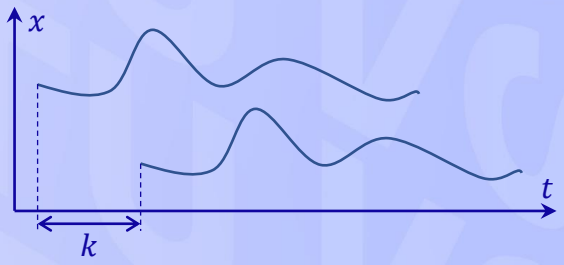
VOLGOGRAD STATE TECHNICAL UNIVERSITY

Способы оценки качества семплирования

- К основным *диагностическим метрикам*, характеризующим качество семплов из MCMC, относятся:

1. **Эффективный размер семпла** (*Effective Sample Size, ESS*).
   - ✓ Показывает, *сколько независимых семплов*, полученных из апостериорного распределения, эквивалентно данной цепочке скоррелированных семплов.
   - ✓ MCMC генерирует *скоррелированные семплы* (каждый следующий семпл зависит от предыдущего). Их информационная ценность *ниже*, чем у независимых семплов.
   - ✓ ESS *корректирует* общее число семплов с учётом *автокорреляции*:

$$ESS = \frac{N}{1 + 2 \cdot \sum_{k=1}^{\infty} \rho_k}, \qquad \rho_k = \frac{\mathrm{Cov}(x_t, x_{t+k})}{Dx_t}.$$



   Здесь N – общее число семплов, ρ_k – ковариация между значениями параметра на итерациях, отстоящих друг от друга на лаг k.
   - ✓ Для *надёжного семплирования* рекомендуется, чтобы *ESS* был не менее 400–500 для каждого параметра.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

2. **Статистика Гельмана-Рубина** (*Gelman-Rubin Statistic, R-hat*).

- ✓ Оценивает *сходимость цепочек* MCMC, сравнивая дисперсию внутри цепочек (*within-chain variance*) с дисперсией между цепочками (*between-chain variance*).

$$\hat{R} = \sqrt{\frac{\hat{V}}{W}}, \qquad V = \frac{N-1}{N}W + \frac{B}{N}, \qquad W = \frac{1}{M}\sum_{m=1}^{M} s_m^2, \qquad B = \frac{N}{M-1}\sum_{m=1}^{M}(\bar{x}_m - \bar{x}).$$

  Здесь N – общее число семплов, W – средняя внутрицепочечная дисперсия, s_m^2 – дисперсия в цепочке m, B – межцепочечная дисперсия, $\hat{x}_m$ – среднее значение параметра в цепочке m, $\hat{x}$ – среднее значение параметра по всем цепочкам.

- ✓ Для вычисления $\hat{R}$ обычно запускается *несколько цепочек MCMC* с разными начальными значениями. Но если цепь одна, её можно разбить на части и анализировать как различные цепочки.
- ✓ *Интерпретация* значения статистики *Гельмана-Рубина*:
  - $0{,}95 < \hat{R} < 1{,}05$: цепочки *хорошо сошлись* к одному и тому же распределению.
  - $\hat{R} \geq 1{,}05$: *плохая сходимость* цепочек, результаты *ненадёжны*.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Способы оценки качества семплирования

- Метрики ESS и $\hat{R}$ *взаимосвязаны*:
  - ✓ Если $\hat{R}$ высокий, то он часто сопровождается низким ESS, так как цепочки не сошлись;
  - ✓ Если $\hat{R}$ близок к единице, но ESS низкий, то это может указывать на *высокую автокорреляцию* внутри цепочек, несмотря на сходимость.
- Обе метрики *важны для оценки качества* MCMC.
- Для улучшения метрик (и качества семплирования) следует *настроить алгоритм* MCMC.
- Данные метрики обычно применяются *совместно с визуализацией других метрик*.

```
Sample: 100%| 2500/2500 [00:14, 170.33it/s, step size=8.73e-01, acc. prob=0.907]

Pyro Summary:
                mean       std    median      5.0%     95.0%     n_eff     r_hat
        mu      2.02      0.05      2.03      1.93      2.10   2105.25      1.00
     sigma      0.50      0.04      0.50      0.44      0.56   1252.09      1.00

Number of divergences: 0
```

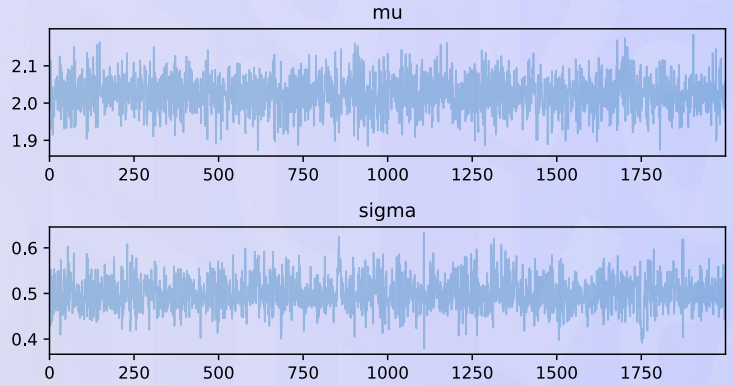
```
Sample: 100%| 2500/2500 [00:11, 211.15it/s, step size=1.39e-01, acc. prob=0.689]
Pyro Summary:
                mean       std    median      5.0%     95.0%     n_eff     r_hat
       mu1      1.78      0.35      1.88      1.21      2.26     15.22      1.13
       mu2      2.05      0.04      2.04      1.98      2.12      8.00      1.17
    sigma1      0.50      0.21      0.45      0.17      0.78     28.10      1.03
    sigma2      0.49      0.06      0.48      0.42      0.59      3.97      1.44
      w[0]      0.28      0.23      0.18      0.04      0.71      4.12      1.35
      w[1]      0.72      0.23      0.82      0.29      0.96      4.12      1.35

Number of divergences: 0
```

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Способы оценки семплов MCMC: траектории семплирования

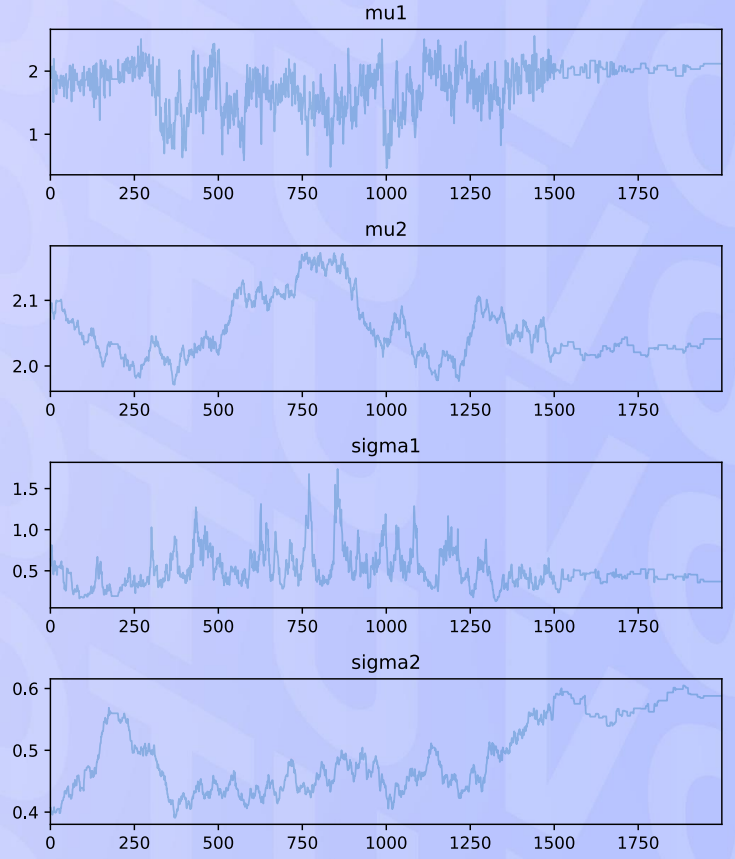
- **Траектории семплирования** (*samples traces*) показывают сам *процесс семплирования* и *степень сходимости* к стационарному распределению.
- Характеристики *хорошей траектории*:
  - ✓ Траектория выглядит как «*белый шум*» без явных трендов, автокорреляции или «застреваний».
  - ✓ Семплы хорошо перемешиваются, цепочка быстро исследует всё пространство параметров.
  - ✓ Если используется несколько цепочек, они перекрываются и выглядят одинаково.
  - ✓ Нет длинных участков, где цепочка «застревает» в одном значении.



VOLGOGRAD STATE TECHNICAL UNIVERSITY

Способы оценки семплов MCMC: траектории семплирования

- Характеристики *плохой траектории*:
  - ✓ Застревание: цепочка может долго оставаться в одной области (например, около одной моды), не переключаясь на другие моды. Это выглядит как длинные горизонтальные участки на трассе.
  - ✓ Плохое перемешивание: Траектория не похожа на «*белый шум*», а имеет тренды, автокорреляцию или резкие скачки.
  - ✓ Если используется несколько цепочек, они могут не перекрываться, а «застревать» в разных модах, что указывает на отсутствие сходимости.
  - ✓ Возможны редкие переходы между модами, но они происходят медленно и неравномерно.
- Цепочка *плохо исследует пространство параметров*. Нужна *настройка алгоритма* MCMC.

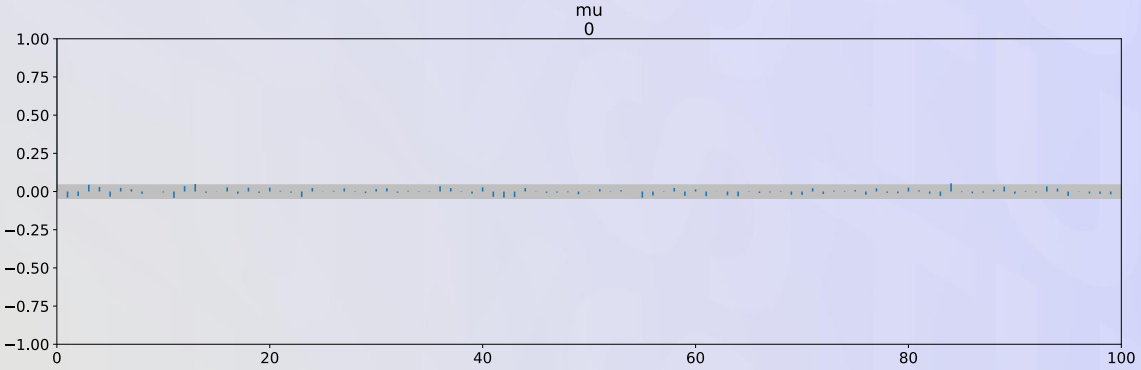


VOLGOGRAD STATE TECHNICAL UNIVERSITY

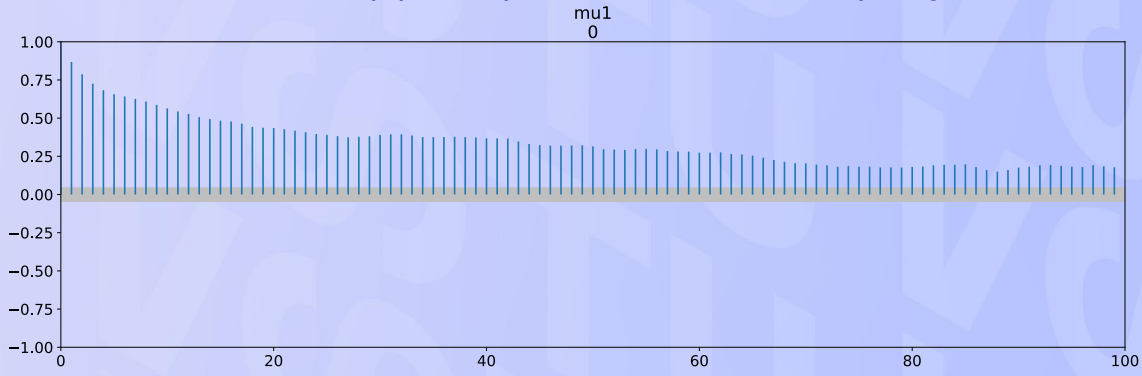
Способы оценки семплов MCMC: гистограмма автокорреляции

- **Гистограмма автокорреляционной функции** для семплов (*autocorr plot*) показывает *автокорреляцию последовательности семплов* для различных значений смещения; полезна в качестве средства *диагностики сходимости*.
- Высокая автокорреляция сообщает о том, что алгоритм MCMC *работает неэффективно*, и нужно либо увеличить количество шагов разогрева (*burn-in*), либо настроить параметры алгоритма (шаг или количество шагов).

Семплы *нескоррелированы*, **хороший** результат



Семплы *скоррелированы*, **плохой** результат



VOLGOGRAD STATE TECHNICAL UNIVERSITY

Демонстрация практических примеров

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Заключение

1. Вспомнили проблему байесовского вывода, выяснили способы её решения.
2. Познакомились с марковскими цепями Монте-Карло и рассмотрели их применимость к решению задачи байесовского вывода.
3. Узнали основные свойства цепей, сформулировали фундаментальную теорему марковских цепей и ее практическое следствие.
4. Определили требования к сходимости марковской цепи к определенному распределению, поговорили про обратимость цепей Маркова.
5. Рассмотрели алгоритм Метрополиса-Гастингса и на практических примерах разобрали механизм его работы. Познакомились с алгоритмом Гиббса.
6. Выявили особенности и ограничения алгоритма Метрополиса-Гастингса.
7. Разобрали метрики контроля качества семплирования алгоритмов МСМС.

VOLGOGRAD STATE TECHNICAL UNIVERSITY

Спасибо за внимание!

Волгоград 2025