クラシックな機械学習の入門

10. マルコフ連鎖モンテカルロ 法

Imputation Posterior

次元の呪い

MCMC

Metropolis-Hastings アルゴリズム

Gibbsサンプリング

by 中川裕志(東京大学)

Sampling法の必要性

- EMやVBで解析的に処理して数値計算に持ち込める場合ばかりではない。
  - VBの場合、因子化仮定による近似も問題。
- シミュレーションで解く方法は対象に係わらず使える。
- ただし、一般にシミュレーションで良い解に収束させるには時間が膨大にかかる。
  - 次元の呪いも影響してくるので、事態は良くない。
- そこで、効率の良いシミュレーション法が必要になる。

Samplingによるシミュレーションを用いる局面

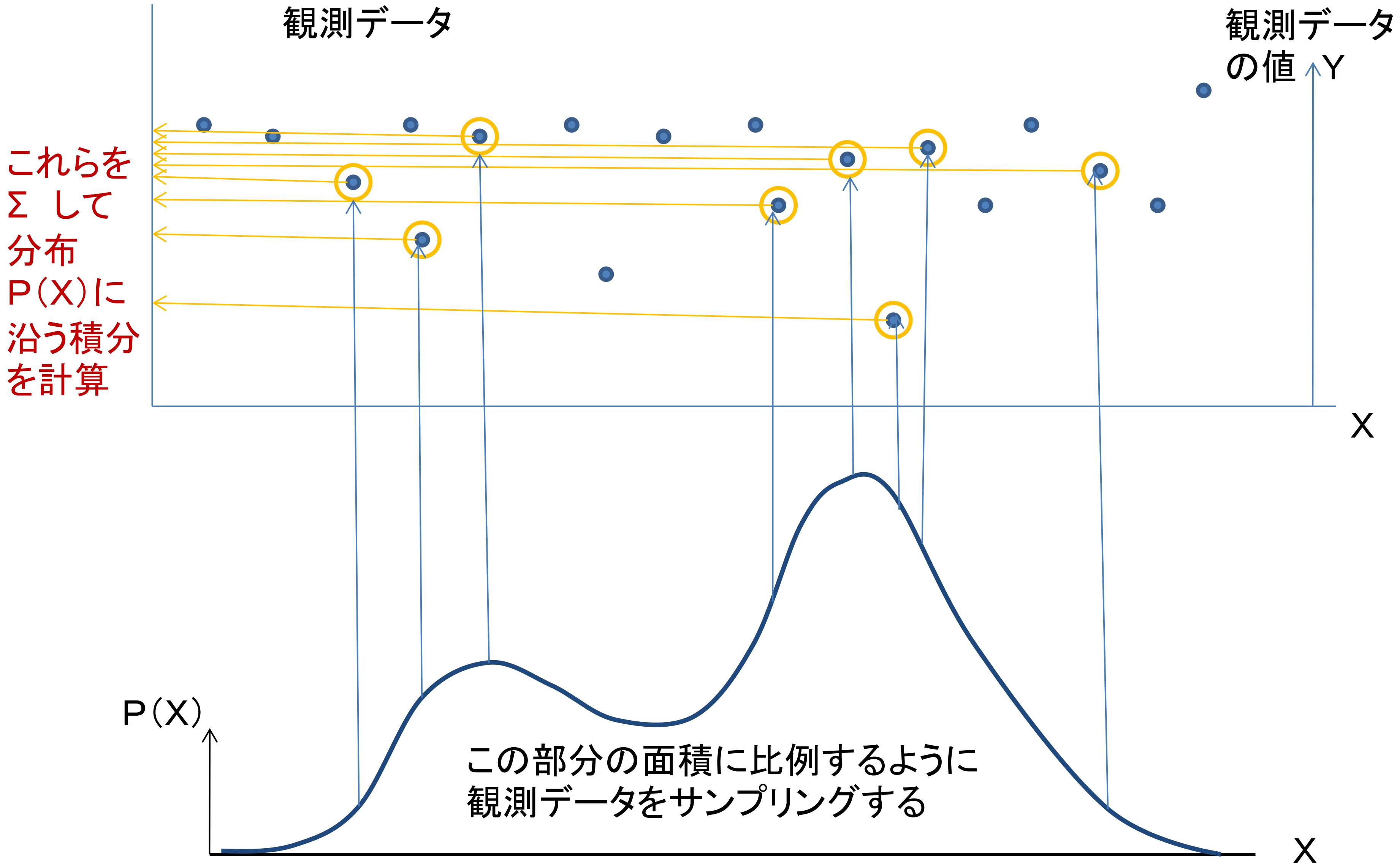
- EM,VBでは以下の更新式を解析解として求めておき、繰り返し計算を行うようになる。
- *繰り返し*
  - $P(Z,X|\,\theta^{old},M) \rightarrow \mathrm{argmax}_{\theta}\exp(\mathrm{E}[\log P(Z,X|\,\theta^{old},M)])$で$\theta$を推定
- この途中で行う期待値計算$\mathrm{E}[\log P(Z,X\,|\,\theta^{old},M)])$の中で周辺化などの積分計算の解析解を求めることが困難である場合は、その部分を期待値計算で用いる確率分布$p(Z)$にかなうようにsamplingしたデータでシミュレーションによって計算する。

EMのQ計算をsamplingで置き換える方法

- Q関数 $Q(\theta,\theta^{old})=\int p(Z\mid\theta^{old},X)\log p(Z,X\mid\theta)\mathrm{d}\theta$
  - この積分を以下のようにしてsamplingで求める
- 事後分布 $p(Z\mid\theta^{old},X)$ を用いて観測データからsamplingして$\{Z^{(l)}\}$$(l=1,2,..,L)$を得る。
- こうして得た $\{Z^{(l)}\}$を用いて、以下のようにθを求める.

$$Q(\theta,\theta^{old})\cong\frac{1}{L}\sum_{l=1}^{L}\log p(Z^{(l)},X\mid\theta)$$

- M-stepは、これらの結果を使って通常のEMと同じように行う。


観測データ
観測データの値 Y
これらをΣ して分布P(X)に沿う積分を計算
X
P(X)
この部分の面積に比例するように観測データをサンプリングする
X

Imputation Posterior: IP Algorithm

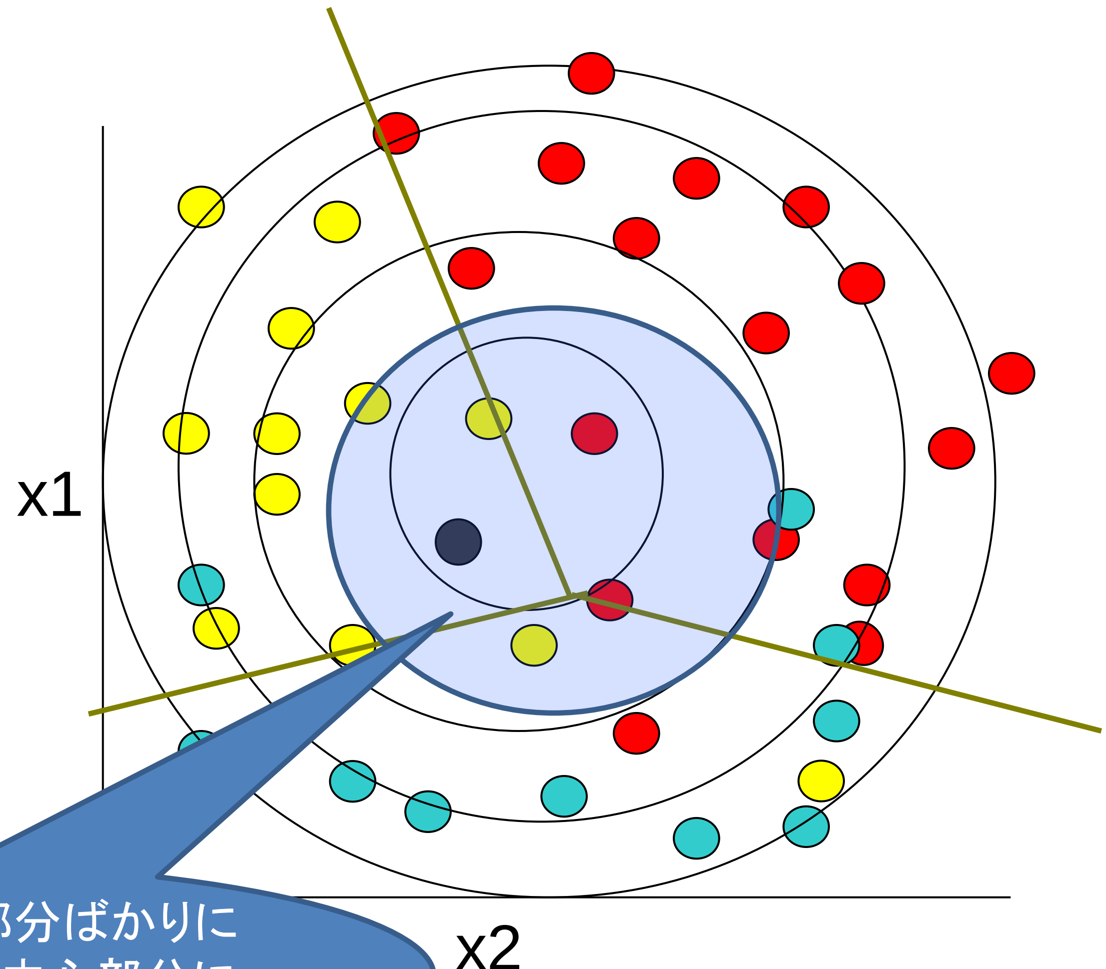
- I-step
  - $p(Z \mid X) = \int p(Z \mid \theta, X) p(\theta \mid X) \mathrm{d}\theta$ の計算にあたって、sample $\theta^{(l)}$ $l=1,2,..,L$ を$p(\theta|X)$にしたがって、取り出し、これを用いて$p(Z|\theta^{(l)},X)$によって$Z^{(l)}$を取り出して$p(Z|X)$を計算
- P-step
  - $p(\theta \mid X) = \int p(\theta \mid Z, X) p(Z \mid X) \mathrm{d}Z$ の計算にあたって、 -I-stepで得たsample｛$Z^{(l)}$｝を用いて、以下のようにθを求める.

$$p(\theta \mid X) \cong \frac{1}{L} \sum_{l=1}^{L} p(\theta \mid Z^{(l)}, X)$$

- しかし、sampleはうまくやらないと効率が悪くなってしまう.（次元の呪い）

次元の呪い(Curse of dimensinality)

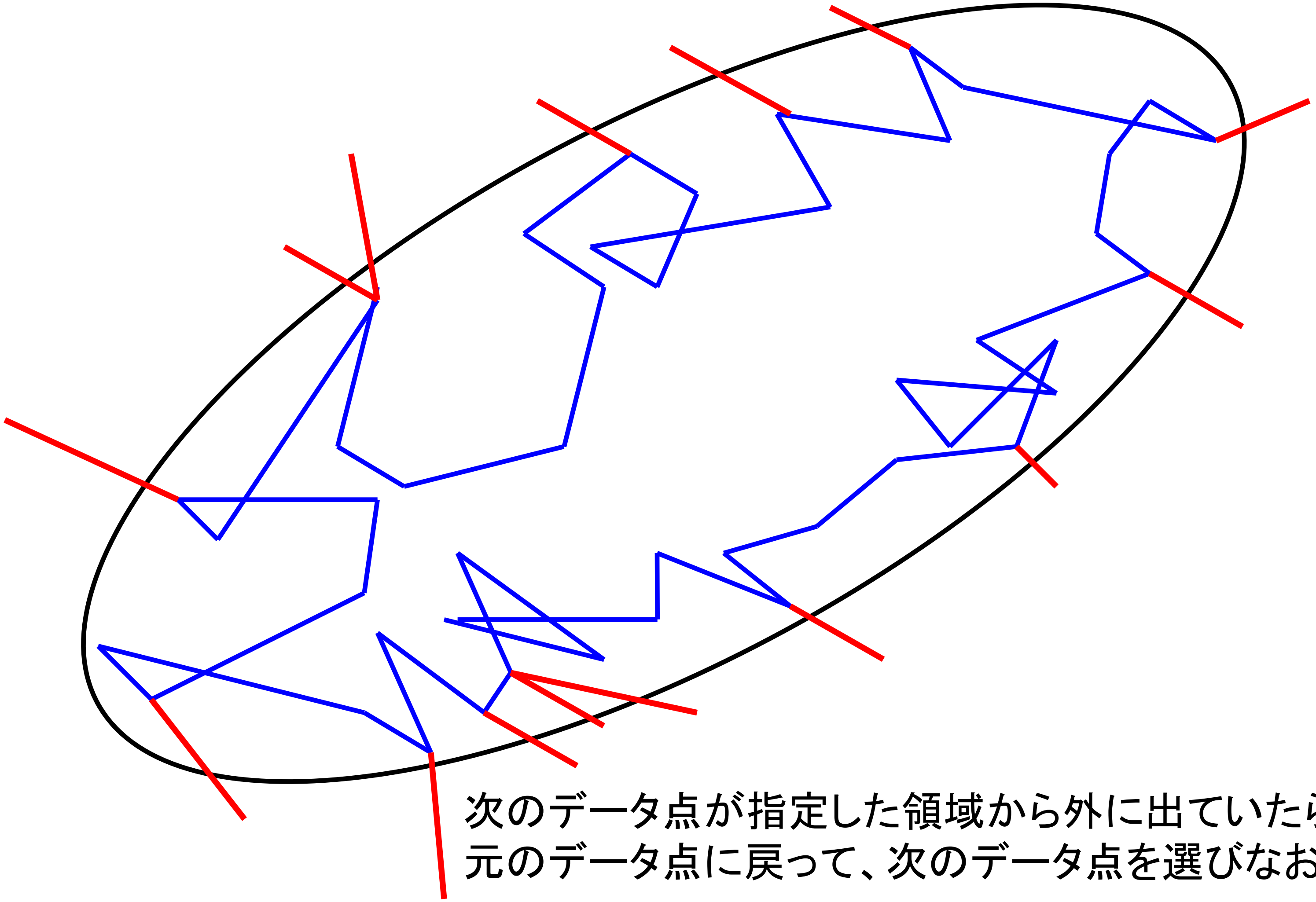
- (x1,x2)の赤、青、黄の教師データが図のように配置。
- 新規のデータ●が何色かを決める方法を教師データから学習



次元が高いと、周辺部分ばかりにサンプル点が集まり、中心部分にサンプルデータが非常に少ない。

- データを記述するための情報の種類(feature)を表す次元Dが増加するとどうなるか？
  - 各次元のfeatureの間の関係を示す係数の数がfeatureのとりうる値N(f)の積で増加。
  - 各featureの値:rによって以下のように識別するとしよう。
    - If $1\text{-}e<r<1$ then A else B
  - $1\text{-}e<r<1$という条件を満たすデータ空間の体積：K（1－$(1\text{-}e)^{D}$）のデータ空間全体:K 1^{D}に対する割合は $1-(1-e)^{D}$
  - この割合はDが大きくなると急激に1に近づく。
  - つまり、Dが大きい次元の空間では、縁の部分の体積ばかり大きくなる。よって、実測された教師データは縁の部分にばかり集まる。
  - よって、rの値の閾値が比較的小さいところになる場合は、教師データが極端に少なくなり、学習の精度が落ちる。

Markov Chain Monte Carlo: MCMC の基本アイデア



次のデータ点が指定した領域から外に出ていたら、
元のデータ点に戻って、次のデータ点を選びなおす。

近似分布の導入

- sampling で$p(Z)$の正確な評価が困難でも、$p(Z)$と比例する分布 $\tilde{p}(Z)$ なら乱数によってsamplingすることができるとしよう. 両者の関係は正規化定数Z_pで下のように書ける。

$$p(Z) = \frac{1}{Z_p}\tilde{p}(Z)$$

本当に知りたいのはこれだが、

- ただし、Z_pは未知。だから、$\tilde{p}(Z)$ のままで計算できることが望ましい。
- さらに近似分布(proposal分布）$q(z^{(t+1)}|z^{(t)})$にしたがい、$z^{(t)}$からsamplingして$z^{(t+1)}$を求める。

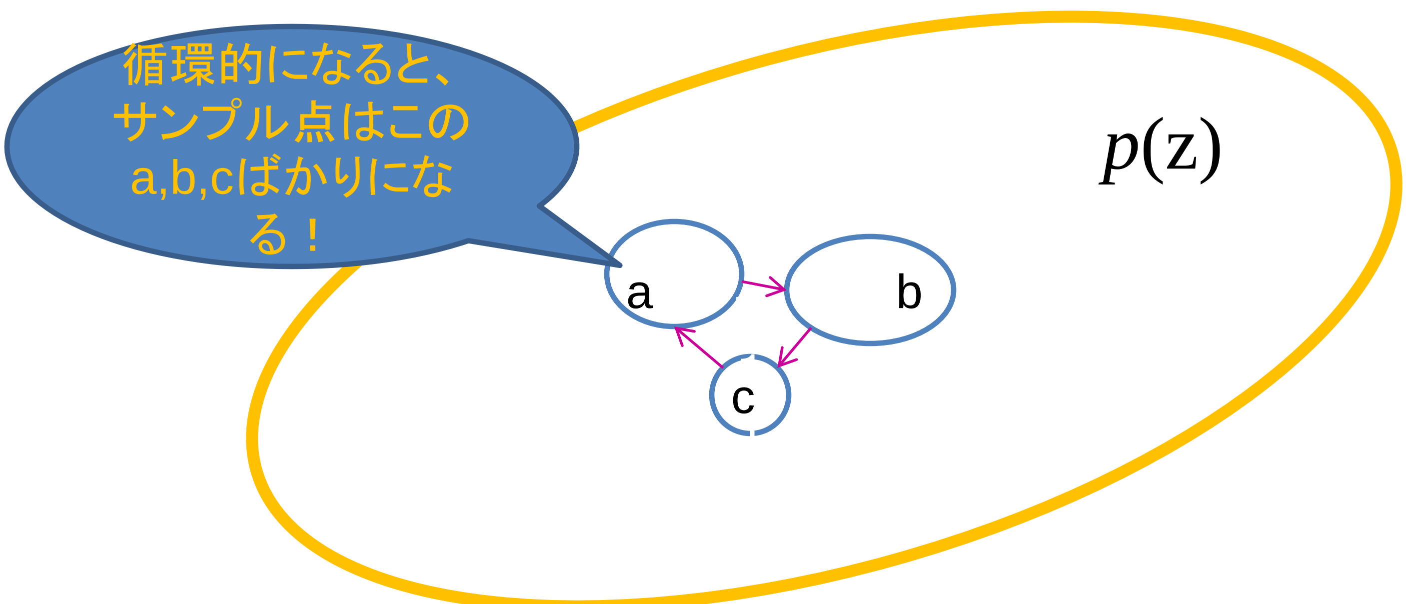
この条件付き確率（遷移確率）は、問題から求めておく

Markov連鎖

- Markov連鎖の定常性

  when $p(\mathbf{z}^{(t+1)}) = \sum_{\mathbf{z}^{(t)}} p(\mathbf{z}^{(t+1)} \mid \mathbf{z}^{(t)})\, p(\mathbf{z}^{(t)})$, then $p(\mathbf{z}^{(t+1)}) = p(\mathbf{z}^{(t)})$

- ただし、定常的であっても、一つに状態に収束しているとは言い切れない。循環的になっているかもし れない。
- 下の図のような状況はまずい。つまり、a,b,cばかりにシミュレーションでサンプルする点が集中してしまい、p(z)の示す領域全体の体積（＝確率）が正確に求まらない。

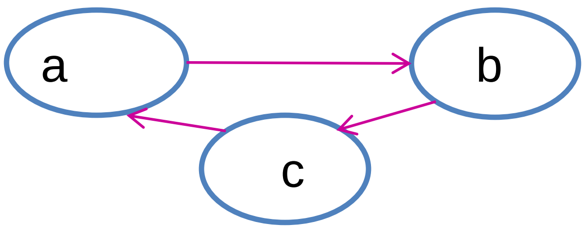


Markov連鎖

- Markov連鎖の定常性

  when $p(\mathbf{z}^{(t+1)}) = \sum_{\mathbf{z}^{(t)}} p(\mathbf{z}^{(t+1)} \mid \mathbf{z}^{(t)}) p(\mathbf{z}^{(t)})$, then $p(\mathbf{z}^{(t+1)}) = p(\mathbf{z}^{(t)})$

- ただし、定常的であっても、一つに状態に収束しているとは言い切れない。循環的になっているかもしれない。



- 循環的でない定常状態になるためには次の詳細釣り合い条件が必要。この条件から定常性もいえる。
- 詳細釣り合い条件：

  あるp^*が存在し,すべてのz, z'に対して

  $$p^*(z')p(z' \to z) = p^*(z)p(z \to z')$$

- 詳細釣り合い条件を書き換えると

$$\frac{p(z' \to z)}{p(z \to z')} = \frac{p(z)}{p(z')} = \frac{\tilde{p}(z)/Z_p}{\tilde{p}(z')/Z_p} = \frac{\tilde{p}(z)}{\tilde{p}(z')}$$

- つまり、z→z’の遷移確率を決めるには、正規化定数：Z_pは知らなくてもよい。

Metropolis　アルゴリズム

Step 1: 近似分布(proposal分布)として、$q(z^{(t+1)}|z^{(t)})$を設定しておく。

Step 2: $q(z^{(t+1)}|z^{(t)})$ を用いてsample $z^{(1)}, z^{(2)},\ldots.\ z^{(t)}$を求めておき、
$\tilde{\hat{p}}(\bullet)$が計算できる状態にした上で、以下のstep,3,4,5で生成する。

Step 3: 新規のsample　z^{new}　を生成する。

Step 4: $A(z^{new}, z^{(t)}) = \min\left(1, \dfrac{\tilde{\hat{p}}(z^{new})}{\tilde{\hat{p}}(z^{(t)})}\right)$　とする。

また　$u \in (0,1)$　をあらかじめ決めた閾値とする。

Step 5: if $A(z^{new},z^{(t)}) > u$ then $z^{(t+1)}=z^{new}$

else $z^{(t)}$からz^{new}を生成してstep 3へ戻る

$z^{(t)}$を捨てずにとっておき、$q(z^{(t+1)}|z^{(t)})$を用いて再度のsample z^{new}の生成で役立てるところがポイント

Metropolisアルゴリズムの味噌

Step 4: $A(z^{new}, z^{(t)}) = \min\left(1, \frac{\tilde{p}(z^{new})}{\tilde{p}(z^{(t)})}\right)$ とする。

また。 $u \in (0,1)$ をあらかじめ決めた閾値とする。

Step 5: if $A(z^{new},z^{(t)}) > u$ then $z^{(t+1)}=z^{new}$

else $z^{(t)}$からz^{new}を生成してstep 3へ戻る

- つまり、$\tilde{p}\left(z^{new}\right) > \tilde{p}\left(z^{(t)}\right) \cdot u$ であること、すなわち新たな $\mathbf{z}^{new}$ が前の値 $\mathbf{z}^{(t)}$よりある程度以上、狙った分布に近い場合のみ採用されることになる。
- $\frac{\tilde{p}(z^{new})}{\tilde{p}(z^{(t)})}$ だから、正規化定数Z_pを知らなくてもよいところが味噌

- 対称性 $q(z^a|z^b)=q(z^b|z^a)$ の条件は $t \to \infty$のとき、samplingされた $\mathbf{z}$ の分布が$p(\mathbf{z})$に近づくことを保証する。

Metropolis-Hastings アルゴリズム

Step 1: 近似分布(proposal分布)として、$q(z^{(t+1)}|z^{(t)})$を設定しておく。

Step 2: $q(z^{(t+1)}|z^{(t)})$ を用いてsample $z^{(1)}, z^{(2)},\dots.\ z^{(t)}$を求めて、
$\tilde{\hat{p}}(\bullet)$ が計算できる状態にしたうえで、以下のstep3,4,5で生成する。

Step 3: 新規のsample z^{new} を生成する。

Step 4:
$$A_k(z^{new}, z^{(t)}) = \min\left(1, \frac{\tilde{\hat{p}}(z^{new})q_k\left(z^{(t)} \mid z^{new}\right)}{\tilde{\hat{p}}(z^{(t)})q_k\left(z^{new} \mid z^{(t)}\right)}\right)$$
とする。

ただしkは検討すべき状態遷移の集合上のインデックス。Step5の条件を満たすものを選ぶ。(**Z^{new}に行ったきり戻ってこられないようなもの、つまりminの第2項の分子が小さいものは使わない**)

また $u \in (0,1)$ をあらかじめ決めた閾値とする。

Step 5: if $A(z^{new},z^{(t)}) > u$ then $z^{(t+1)}=z^{new}$

else $z^{(t)}$からz^{new}を生成してstep 3へ戻る

Gibbs Sampling

- $\mathbf{z}=(z_1 z_2,\ldots z_M)$からなるベクトルの場合, $\mathbf{z}$の全要素を一度に生成して、条件$A(\mathbf{z}^{new},\mathbf{z}^{(t)})>u$を満たすかをチェックするのは効率が悪い。
- そこで、z_j^{new}を$z_1^{(t)},\ldots, z_{j-1}^{(t)}$が与えられた条件でsamplingする方法が有効。

- Gibbs　Samplingでは、z_jのsampling　と評価を、条件$A(\mathbf{z}^{new},\mathbf{z}^{(t)})>u$のチェックの代わりに、それ以前の結果を利用して行う。

Gibbs Sampling

1. $z_1 z_2, \ldots z_M$の初期化
2. 以下を$t=1$からTまで繰り返す
   1. Sample $z_1^{(t+1)} \leftarrow z_1^{new} : p(z_1^{new} | z_2^{(t)}, z_3^{(t)}, \ldots, z_M^{(t)})$
   2. Sample $z_2^{(t+1)} \leftarrow z_2^{new} : p(z_2^{new} | z_1^{(t+1)}, z_3^{(t)}, \ldots, z_M^{(t)})$
   3. Sample $z_3^{(t+1)} \leftarrow z_3^{new} : p(z_3^{new} | .., z_2^{(t+1)}, z_4^{(t)}, \ldots, z_M^{(t)})$

   …

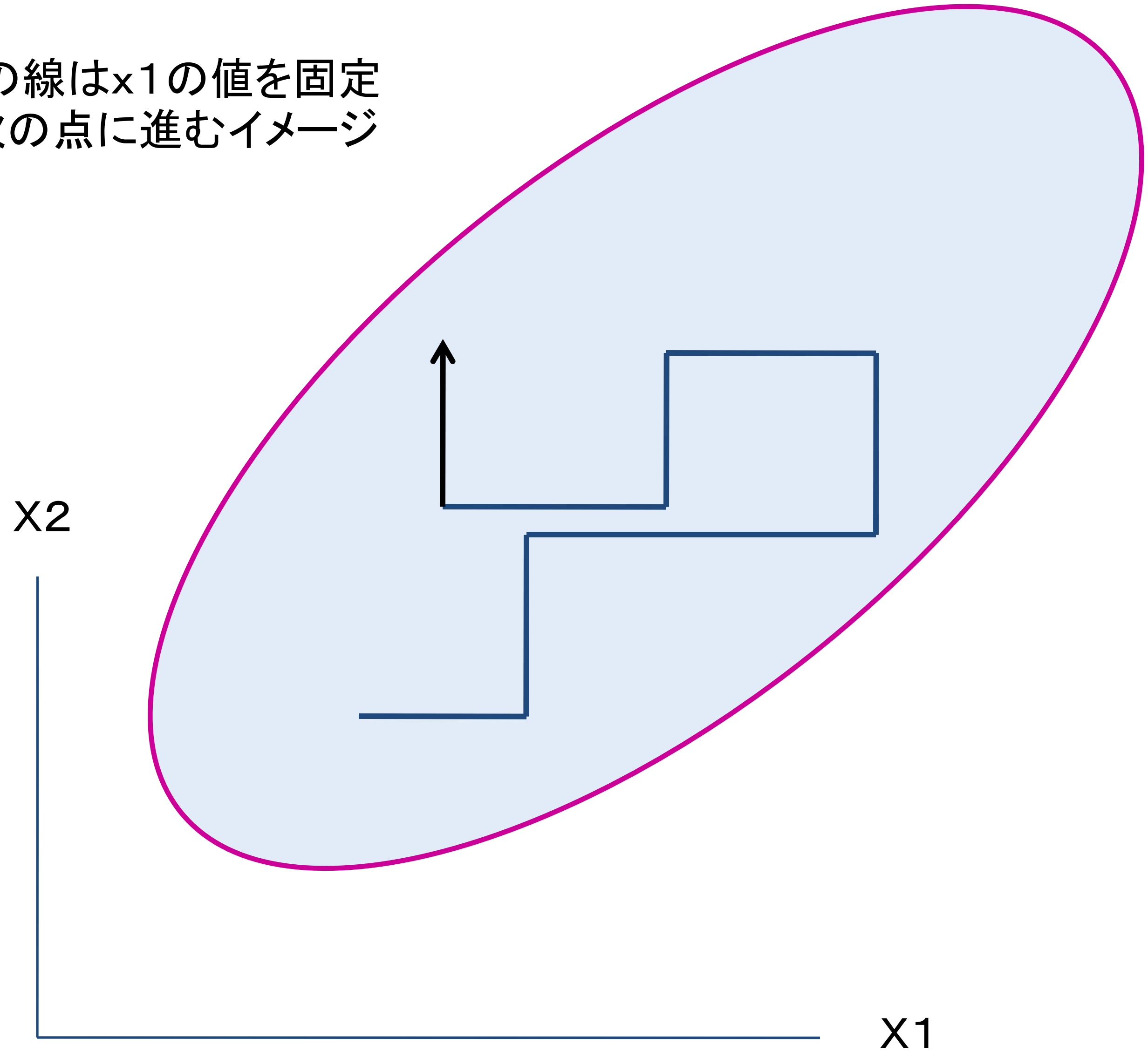
   j. Sample $z_j^{(t+1)} \leftarrow z_j^{new} : p(z_j^{new} | .., z_{j-1}^{(t+1)}, z_{j+1}^{(t)}, \ldots, z_M^{(t)})$

   …

   M. Sample $z_M^{(t+1)} \leftarrow z_M^{new} : p(z_M^{new} | .., z_{M-1}^{(t+1)})$

水平の線はx2の値を固定
して次の点に進むイメージ

垂直の線はx1の値を固定
して次の点に進むイメージ



z_kを更新するときには$\mathbf{z}_{\backslash k}$は固定されていることに注意

$$\Rightarrow \boxed{\quad z^{(t)}{}_{\backslash k} = z^{new}{}_{\backslash k}}$$

$$q_k\left(z^{new} \mid z^{(t)}\right) = p(z^{new}{}_k \mid z^{(t)}{}_{\backslash k})$$

$$q_k\left(z^{(t)} \mid z^{new}\right) = p(z^{(t)}{}_k \mid z^{new}{}_{\backslash k})$$

$$p(z^{new}) = p(z^{new}{}_k \mid z^{(t)}{}_{\backslash k})\, p\left(z^{(t)}{}_{\backslash k}\right)$$

$$p(z^{(t)}) = p(z^{(t)}{}_k \mid z^{new}{}_{\backslash k})\, p\left(z^{new}{}_{\backslash k}\right)$$

これを利用してMetroplis– Hasting法を書き換える

$$A_k(z^{new}, z^{(t)}) = \min\left(1, \frac{\tilde{\hat{p}}(z^{new})\, q_k\left(z^{(t)} \mid z^{new}\right)}{\tilde{\hat{p}}(z^{(t)})\, q_k\left(z^{new} \mid z^{(t)}\right)}\right)$$

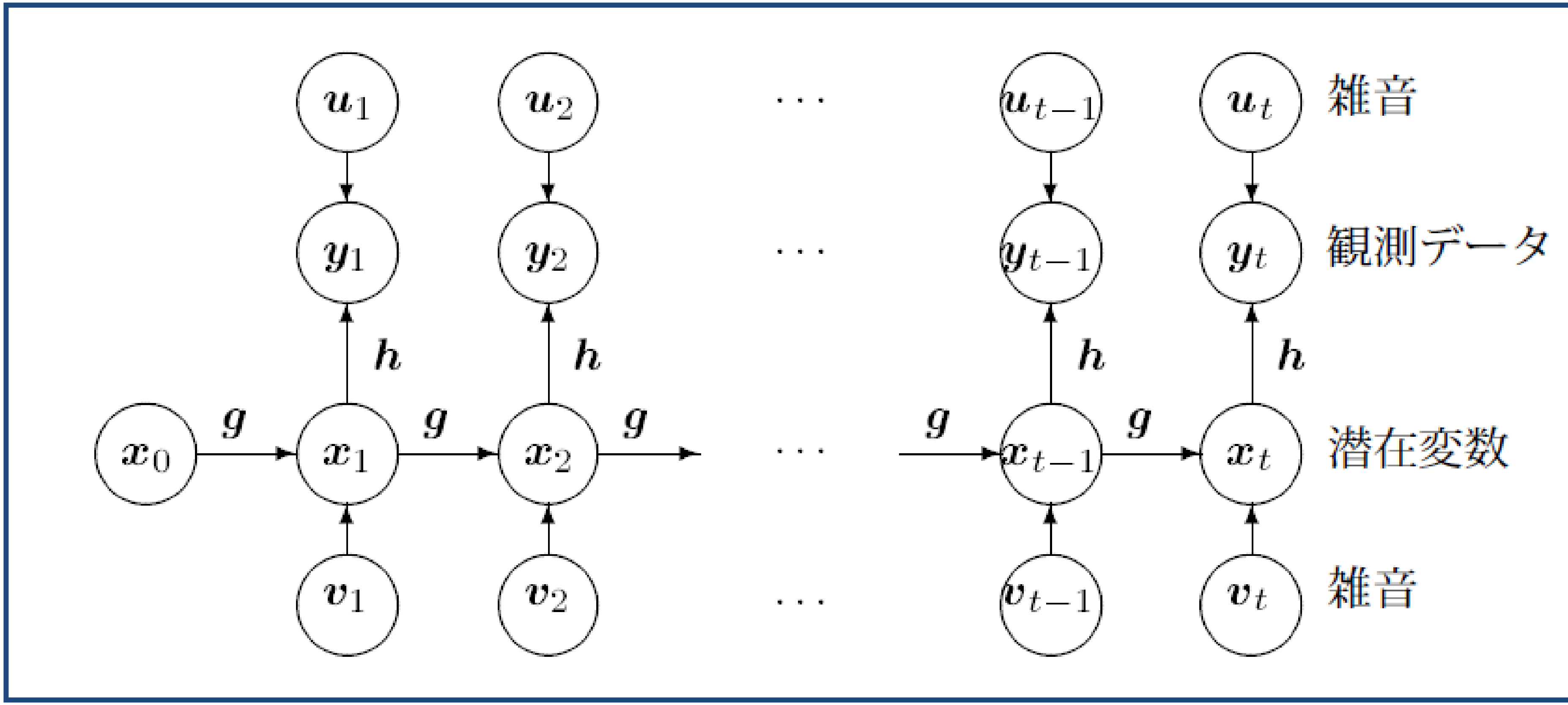
$$= \min\left(1, \frac{p(z^{new}{}_k \mid z^{(t)}{}_{\backslash k})\, p\left(z^{(t)}{}_{\backslash k}\right) q_k\left(z^{(t)} \mid z^{new}\right)}{p(z^{(t)}{}_k \mid z^{new}{}_{\backslash k})\, p\left(z^{new}{}_{\backslash k}\right) q_k\left(z^{new} \mid z^{(t)}\right)}\right)$$

$$= \min\left(1, \frac{\cancel{p(z^{new}{}_k \mid z^{(t)}{}_{\backslash k})}\, p\left(\cancel{z^{(t)}{}_{\backslash k}}\right) \cancel{p(z^{(t)}{}_k \mid z^{new}{}_{\backslash k})}}{\cancel{p(z^{(t)}{}_k \mid z^{new}{}_{\backslash k})}\, p\left(\cancel{z^{new}{}_{\backslash k}}\right) \cancel{p(z^{new}{}_k \mid z^{(t)}{}_{\backslash k})}}\right) = 1$$

よって、新規のデータは常に採用される$\Rightarrow$ Gibbs　Sampling

粒子フィルタ

- 内部状態x_tが　$x_{t+1} = g(x_t) + v_t$(ガウス雑音）という更新式で変化していく
- 観測されるのは$y_t = h(x_t) + u_t v_t$(ガウス雑音）
- t毎の観測値を用いてx_tを推定、$E_t[f(x_t)]$を計算するメカニズム
  - カルマンフィルタによく似ている。



$$\boldsymbol{x}_t = \boldsymbol{g}(\boldsymbol{x}_{t-1}) + \boldsymbol{v}_t \qquad \boldsymbol{y}_t = \boldsymbol{h}(\boldsymbol{x}_t) + \boldsymbol{u}_t$$

$$p(\boldsymbol{v}_t) = \mathcal{N}(0, \Sigma_v) \qquad p(\boldsymbol{u}_t) = \mathcal{N}(0, \Sigma_u)$$

$$p(\boldsymbol{y}_t \mid \boldsymbol{x}_t) = \frac{1}{\sqrt{(2\pi)^K det(\Sigma_u)}} \exp\left(-\frac{1}{2}(\boldsymbol{y}_t - \boldsymbol{x}_t)^\top \Sigma_u^{-1} (\boldsymbol{y}_t - \boldsymbol{x}_t)\right)$$

我々が知りたいのは潜在変数$\boldsymbol{x}$の推定値，およびそれに対する何らかの関数の値$f(\boldsymbol{x}_t)$たとえば$\boldsymbol{x}$のモーメントである．これを得るために Monte Carlo 法を$\boldsymbol{x}_t$に対して適用する．

すなわち，条件付き確率

$$p(\boldsymbol{x}_t \mid \boldsymbol{y}_{t-1}, \ldots, \boldsymbol{y}_1) \tag{8.36}$$

に従うN個のランダムなサンプル$\{\boldsymbol{x}_t^{(n)} \mid n = 1, \ldots, N\}$を生成する．このとき，$\boldsymbol{x}_t$に対する何らかの関数の値$f(\boldsymbol{x}_t)$，例えばモーメントを次式で得る．

$$\int f(\boldsymbol{x}_t) p(\boldsymbol{x}_t \mid \boldsymbol{y}_{t-1}, \ldots, \boldsymbol{y}_1) \mathrm{d}\boldsymbol{x}_t = \frac{1}{N} \sum_{n=1}^{N} f(\boldsymbol{x}_t^{(n)}) \tag{8.37}$$

一例として，期待値$E[\boldsymbol{x}_t]$は下式となる．

$$\int \boldsymbol{x}_t p(\boldsymbol{x}_t \mid \boldsymbol{y}_{t-1}, \ldots, \boldsymbol{y}_1) \mathrm{d}\boldsymbol{x}_t = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_t^{(n)} \tag{8.38}$$

この計算をするために必要なのは，$\{\boldsymbol{x}_t^{(n)} \mid n = 1, \ldots, N\}$を生成する以下のアルゴリズムである．

step 0: 初期化 $\boldsymbol{x}_0$ の N 個のランダムな初期値 $\{\boldsymbol{x}_0^{(n)} \mid n = 1, \ldots, N\}$ を生成する．もし，事前知識として $\boldsymbol{x}_0$ の近似的確率密度関数 $q(\boldsymbol{x}_0)$ が分かっていたなら，$q(\boldsymbol{x}_0)$ を用いてこれらの初期値を生成する．

step 1: t の繰り返し $t = 1, \ldots, T$ に対して step 2 から step 5 を繰り返す

step 2: n の繰り返し ． $n = 1, \ldots, N$ に対して step 2-1 から step 2-4 を繰り返す

step 2-1 $p(\boldsymbol{v}_t)$ に従う乱数 $\boldsymbol{v}_t^{(n)}$ を生成

step 2-2 $\hat{\boldsymbol{x}}_t^{(n)} = \boldsymbol{g}(\boldsymbol{x}_{t-1}^{(n)}) + \boldsymbol{v}_t^{(n)}$

step 2-3 $p(\boldsymbol{u}_t)$ に従う乱数 $\boldsymbol{u}_t^{(n)}$ を生成

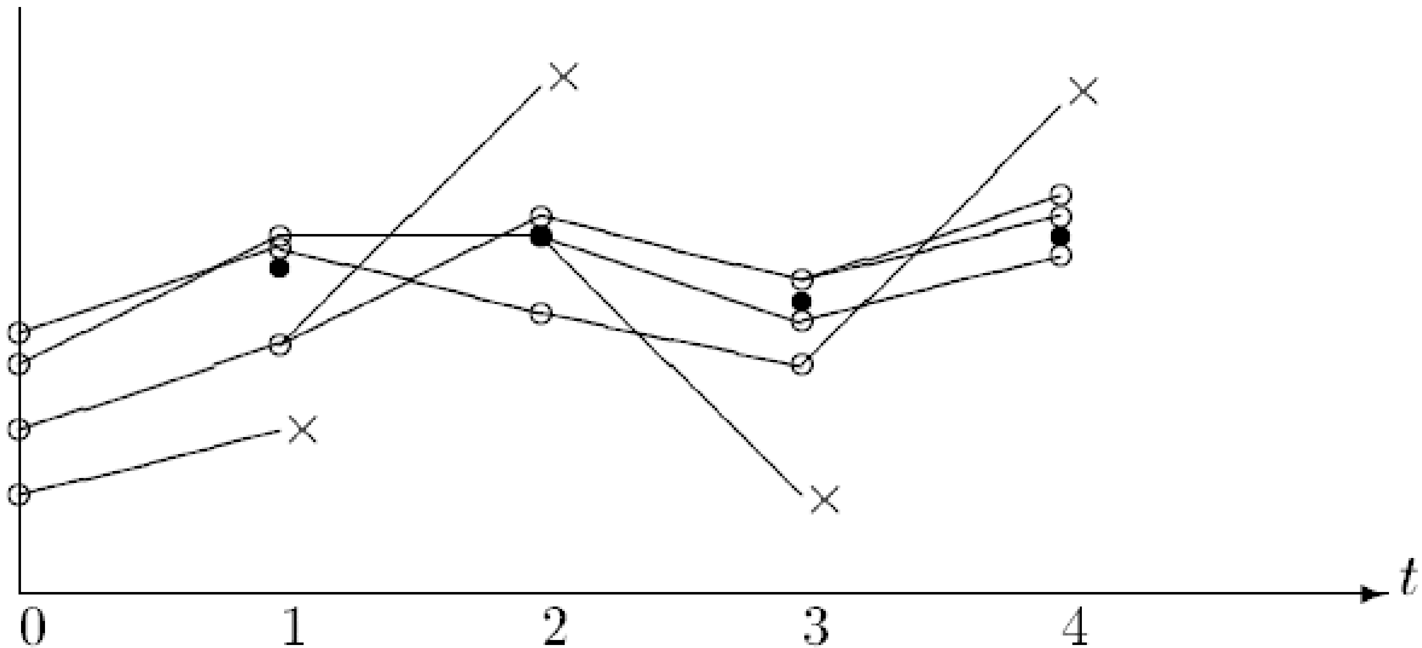
step 2-4 $p(\boldsymbol{y}_t \mid \hat{\boldsymbol{x}}_t^{(n)} + \boldsymbol{u}_t^{(n)})$ を計算

step 3

$$w_t^{(n)} = \frac{p(\boldsymbol{y}_t \mid \hat{\boldsymbol{x}}_t^{(n)} + \boldsymbol{u}_t^{(n)})}{\sum_{n=1}^{N} p(\boldsymbol{y}_t \mid \hat{\boldsymbol{x}}_t^{(n)} + \boldsymbol{u}_t^{(n)})}$$

step 4: 再サンプリング $\{\hat{\boldsymbol{x}}_t^{(n)} \mid n = 1, \ldots, N\}$ から $w_t^{(n)}$ に比例する回数で N 個を復元抽出し，これを $\{\boldsymbol{x}_t^{(n)} \mid n = 1, \ldots, N\}$ とする

step 5 目的の関数 f に対して $E[f(\boldsymbol{x}_t)] = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_t^{(n)}$ を求める



● 観測データ ○ 粒子すなわちサンプル $\boldsymbol{x}_t$
× 破棄されたサンプル $\boldsymbol{x}_t$