Machine Learning for Signal Processing
Predicting and Estimation from Time Series: Part 3

Bhiksha Raj

The Problem

- Given a "state-space" system, where the system travels according to a state-evolution relation

$$s_t = f(s_{t-1}, \varepsilon_t)$$

- And we only receive a sequence of observations that relate stochastically to the state

$$o_t = g(s_t, \gamma_t)$$

- We must determine the sequence of underlying states

- Many applications: Control (e.g. autonomous vehicles, robots) and prediction (e.g. stock market)

Approach

- Given the best guest $\hat{s}_{t-1}$ for the state at time $t-1$, *predict the state* $\bar{s}_t$ at time t

$$\bar{s}_t = f(\hat{s}_{t-1}, \varepsilon_t)$$

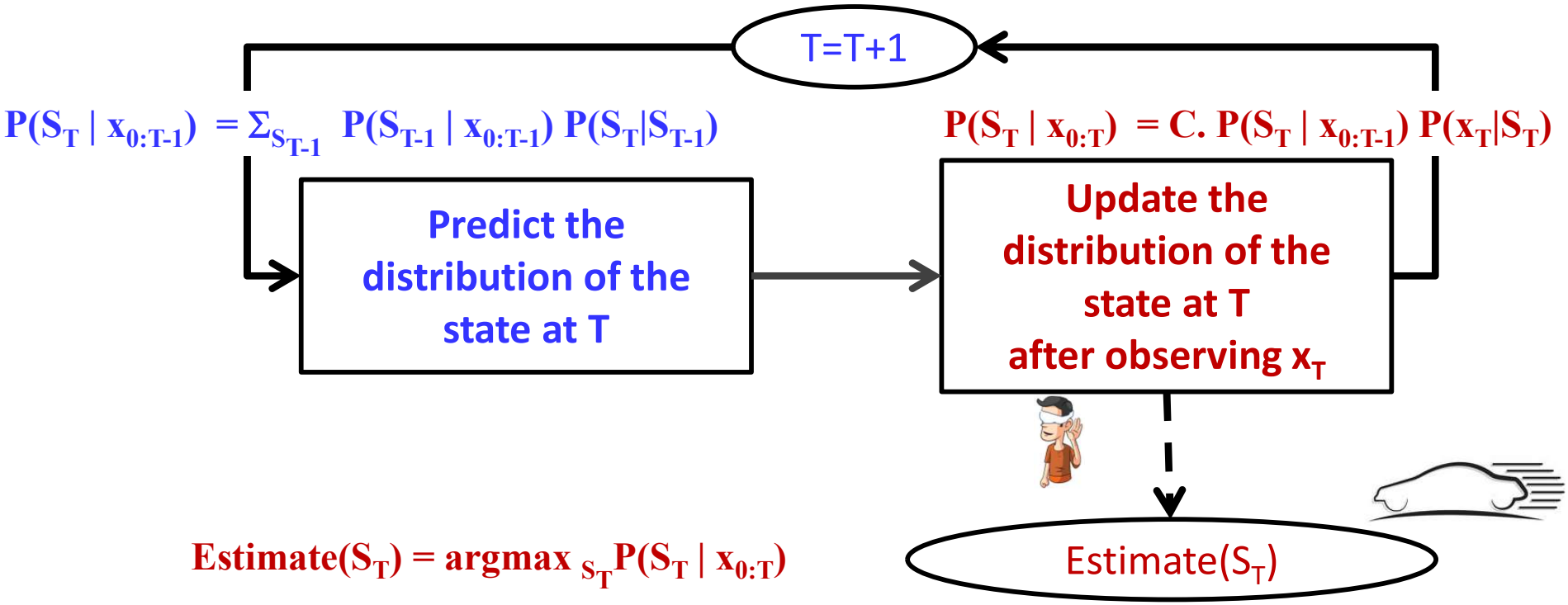
- Using the predicted state $\bar{s}_t$, predict the *observation* $\bar{o}_t$

$$\bar{o}_t = g(\bar{s}_t, \gamma_t)$$

- Make an actual observation o_t
- Update your best guess for the state at time t by

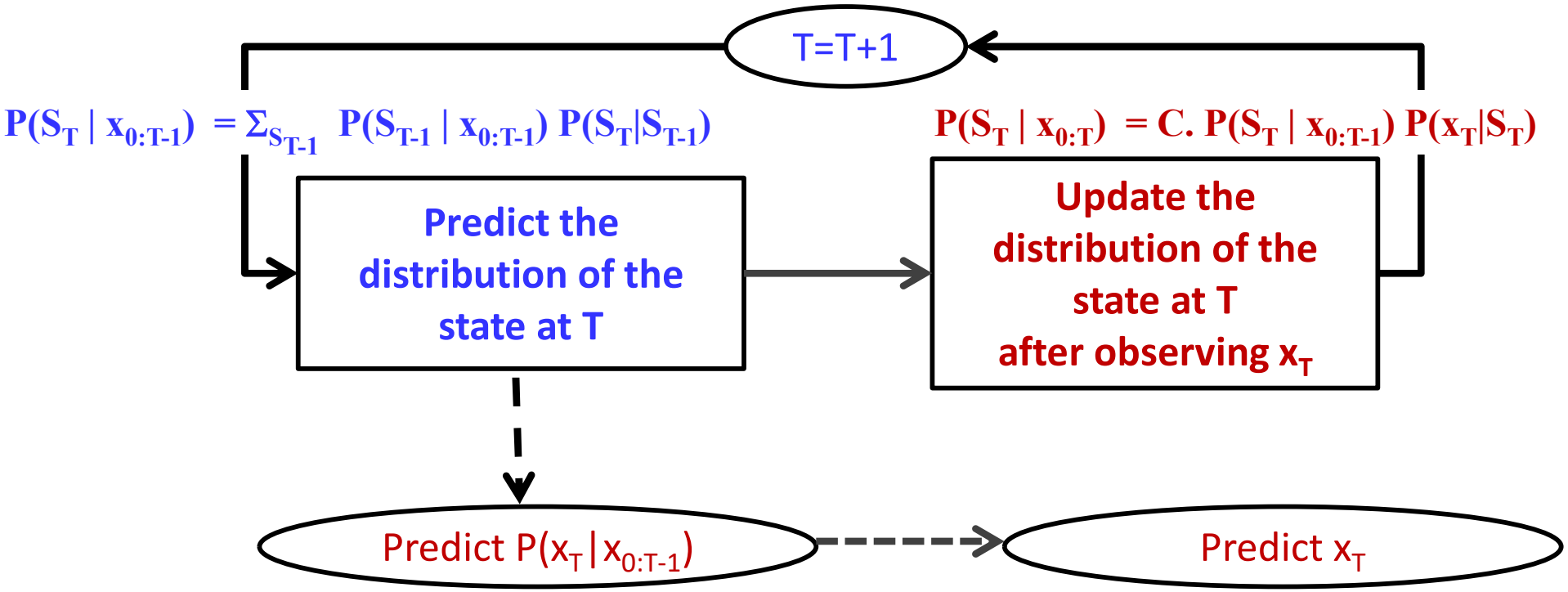
$$\hat{s}_t = \bar{s}_t + K(o_t - \bar{o}_t)$$

This is actually an implementation of the following



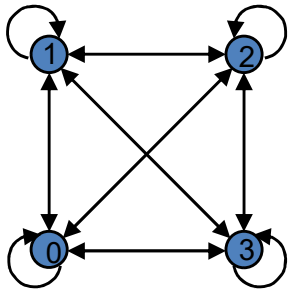
- The state is estimated from the updated distribution
  - The updated distribution is propagated into time, not the state

You can also predict the next observation

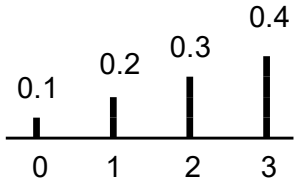


- The probability distribution for the observations at the next time is a mixture:
- $P(X_t|X_{0:t-1}) = \sum_{S_t} P(X_t|S_t)P(S_t|X_{0:t-1})$
- The actual observation can be predicted from $P(x_T \mid x_{0:T-1})$

Discrete vs. Continuous State Systems



$\pi =$



$$s_t = f(s_{t-1}, \varepsilon_t)$$

$$o_t = g(s_t, \gamma_t)$$

Prediction at time t:

$$P(S_t|O_{0:t-1}) = \sum_{S_{t-1}} P(S_{t-1}|O_{0:t-1})P(S_t|S_{t-1})$$

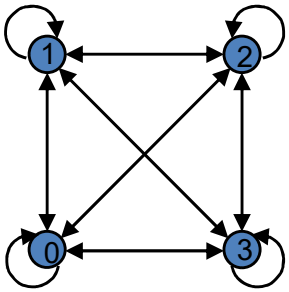
$$P(S_t|O_{0:t-1}) = \int_{-\infty}^{\infty} P(S_{t-1}|O_{0:t-1})P(S_t|S_{t-1})dS_{t-1}$$

Update after observing O_t:

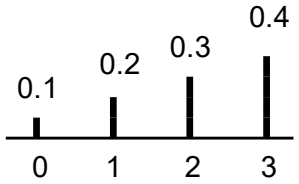
$$P(S_t|O_{0:t}) = C.P(S_t|O_{0:t-1})P(O_t|S_t)$$

$$P(S_t|O_{0:t}) = C.P(S_t|O_{0:t-1})P(O_t|S_t)$$

Discrete vs. Continuous State Systems



$\pi =$



$$s_t = f(s_{t-1}, \varepsilon_t)$$

$$o_t = g(s_t, \gamma_t)$$

Parameters		
Initial state prob.	π	$P(s)$
Transition prob	$P(s_t = j \mid s_{t-1} = i)$	$P(s_t \mid s_{t-1})$
Observation prob	$P(O \mid s)$	$P(O \mid s)$

Special case: Linear Gaussian model

$$s_t = A_t s_{t-1} + \varepsilon_t$$

$$P(\varepsilon) = \frac{1}{\sqrt{(2\pi)^d \,|\Theta_\varepsilon|}} \exp\left(-0.5(\varepsilon - \mu_\varepsilon)^T \Theta_\varepsilon^{-1}(\varepsilon - \mu_\varepsilon)\right)$$

$$o_t = B_t s_t + \gamma_t$$

$$P(\gamma) = \frac{1}{\sqrt{(2\pi)^d \,|\Theta_\gamma|}} \exp\left(-0.5(\gamma - \mu_\gamma)^T \Theta_\gamma^{-1}(\gamma - \mu_\gamma)\right)$$

- A *linear* state dynamics equation
  - Probability of state driving term ε is Gaussian
  - Sometimes viewed as a driving term μ_ε and additive zero-mean noise
- A *linear* observation equation
  - Probability of observation noise γ is Gaussian
- A_t, B_t and Gaussian parameters assumed known
  - May vary with time

Linear model example
The wind and the target

- **State:** Wind speed at time t depends on speed at time t-1

$$S_t = S_{t-1} + \epsilon_t$$

- **Observation:** Arrow position at time t depends on wind speed at time t

$$O_t = BS_t + \gamma_t$$

The Kalman filter

- Prediction (based on state equation)

$$\bar{s}_t = A_t \hat{s}_{t-1} + \mu_\varepsilon \qquad s_t = A_t s_{t-1} + \varepsilon_t$$

$$R_t = \Theta_\varepsilon + A_t \hat{R}_{t-1} A_t^T$$

- Update (using observation and observation equation)

$$K_t = R_t B_t^T \left(B_t R_t B_t^T + \Theta_\gamma \right)^{-1} \qquad o_t = B_t s_t + \gamma_t$$

$$\hat{s}_t = \bar{s}_t + K_t \left(o_t - B_t \bar{s}_t - \mu_\gamma \right)$$

$$\hat{R}_t = \left(I - K_t B_t \right) R_t$$

Problems

$$s_t = f(s_{t-1}, \varepsilon_t)$$

$$o_t = g(s_t, \gamma_t)$$

- f() and/or g() may not be nice linear functions
  - Conventional Kalman update rules are no longer valid
- ε and/or γ may not be Gaussian
  - Gaussian based update rules no longer valid

Problems

$$s_t = f(s_{t-1}, \varepsilon_t)$$

$$o_t = g(s_t, \gamma_t)$$

- f() and/or g() may not be nice linear functions
  - Conventional Kalman update rules are no longer valid

- ε and/or γ may not be Gaussian
  - Gaussian based update rules no longer valid

The Extended Kalman filter

$$s_t = f(s_{t-1}) + \varepsilon$$

$$o_t = g(s_t) + \gamma$$

- Prediction

$$\bar{s}_t = f(\hat{s}_{t-1})$$

$$R_t = \Theta_\varepsilon + A_t \hat{R}_{t-1} A_t^T$$

$$A_t = J_f(\hat{s}_{t-1})$$

$$B_t = J_g(\bar{s}_t)$$

Jacobians used in Linearization

- Update

$$K_t = R_t B_t^T \left(B_t R_t B_t^T + \Theta_\gamma \right)^{-1}$$

$$\hat{s}_t = \bar{s}_t + K_t \left(o_t - g(\bar{s}_t) \right)$$

Assuming ε and γ are 0 mean for simplicity

$$\hat{R}_t = \left(I - K_t B_t \right) R_t$$

A different problem: Non-Gaussian PDFs

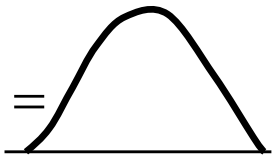
$$o_t = g(s_t) + \gamma \qquad s_t = f(s_{t-1}) + \varepsilon$$

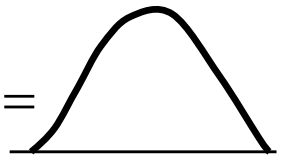
- We have assumed so far that:
  - $P_0(s)$ is Gaussian or can be approximated as Gaussian
  - $P(\varepsilon)$ is Gaussian
  - $P(\gamma)$ is Gaussian
- This has a happy consequence: All distributions remain Gaussian
  - Even if $g()$ and/or $f()$ are nonlinear.

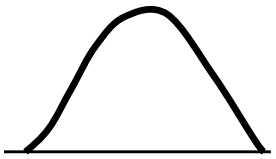
Linear Gaussian Model

$$s_t = A_t s_{t-1} + \varepsilon_t$$

$$o_t = B_t s_t + \gamma_t$$

$P(s)$ = 
a priori

$P(s_t|s_{t-1})$ = 
Transition prob.

$P(O_t|s_t)$ = 
State output prob

$P(s_0) = P(s)$

$P(s_0|O_0) = C\,P(s_0)\,P(O_0|s_0)$

$$P(s_1|O_0) = \int_{-\infty}^{\infty} P(s_0|O_0)P(s_1|s_0)ds_0$$

$P(s_1|O_{0:1}) = C\,P(s_1|O_0)\,P(O_1|s_0)$

$$P(s_2|O_{0:1}) = \int_{-\infty}^{\infty} P(s_1|O_{0:1})P(s_2|s_1)ds_1$$

$P(s_2|O_{0:2}) = C\,P(s_2|O_{0:1})\,P(O_2|s_2)$

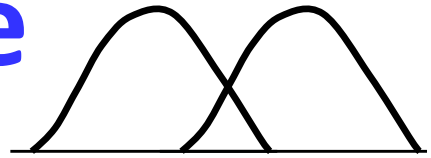
All distributions remain Gaussian

A different problem: Non-Gaussian PDFs

$$o_t = g(s_t) + \gamma \qquad s_t = f(s_{t-1}) + \varepsilon$$

- We have assumed so far that:
  - $P_0(s)$ is Gaussian or can be approximated as Gaussian
  - $P(\varepsilon)$ is Gaussian
  - $P(\gamma)$ is Gaussian
- This has a happy consequence: All distributions remain Gaussian
  - Even if $g()$ and/or $f()$ are nonlinear.
- But when any of these are not Gaussian, the results are not so happy

A simple case

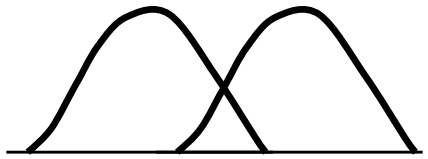


$$o_t = Bs_t + \gamma$$

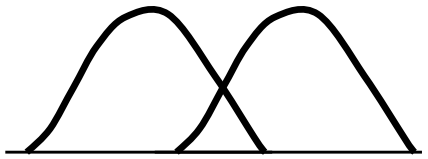
$$P(\gamma) = \sum_{i=0}^{1} w_i Gaussian(\gamma; \mu_i, \Theta_i)$$

- P(γ) is a mixture of only two Gaussians
- *o* is a linear function of *s*
  - Non-linear functions would be linearized anyway
- P(o|s) is also a Gaussian mixture!

$$P(o_t \mid s_t) = P(\gamma = o_t - Bs_t) = \sum_{i=0}^{1} w_i Gaussian(o; \mu_i + Bs_t, \Theta_i)$$

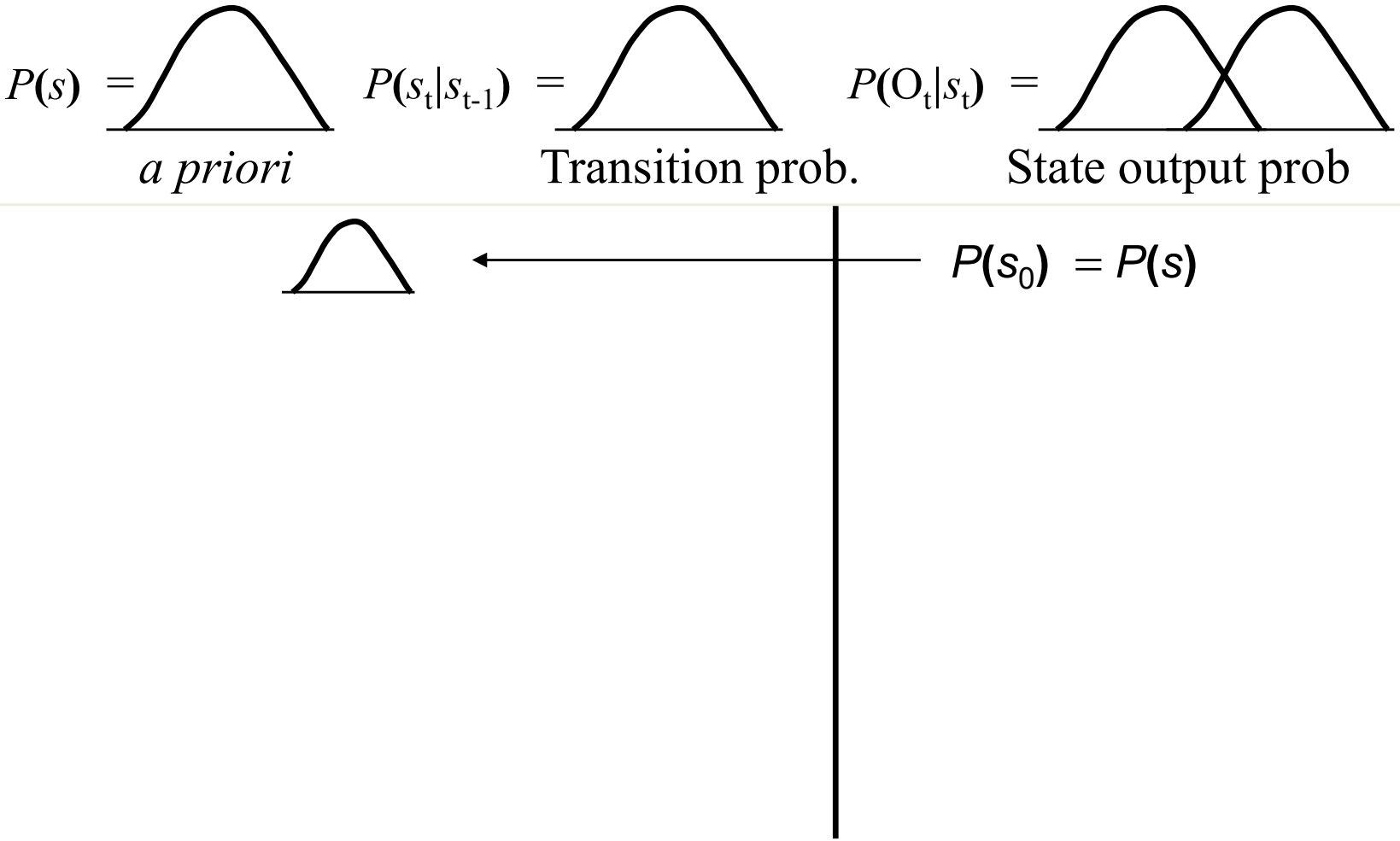


$P(\gamma)$

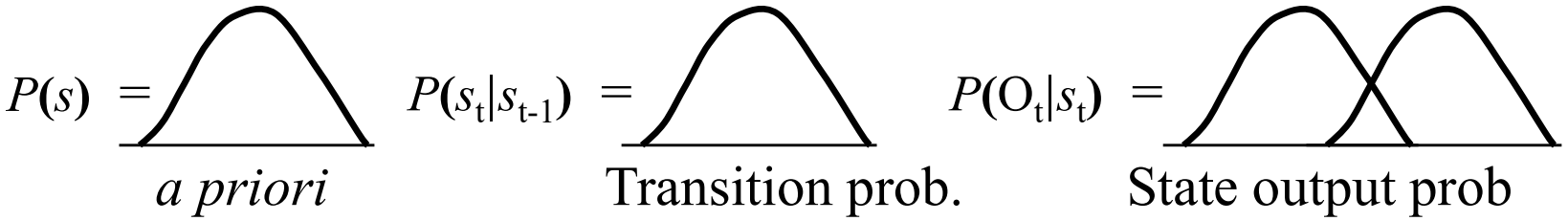


$P(o_t \mid s_t)$

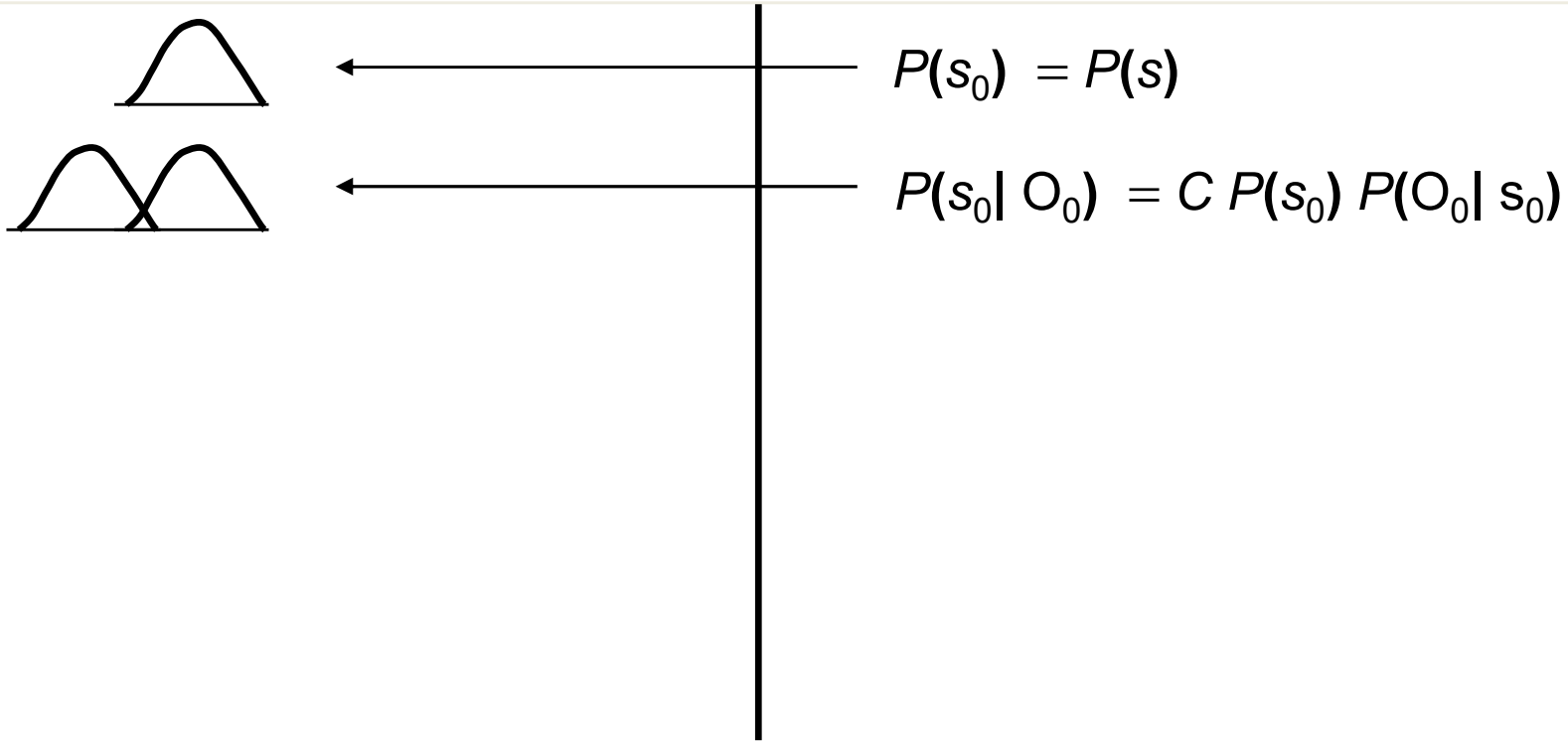
When distributions are not Gaussian



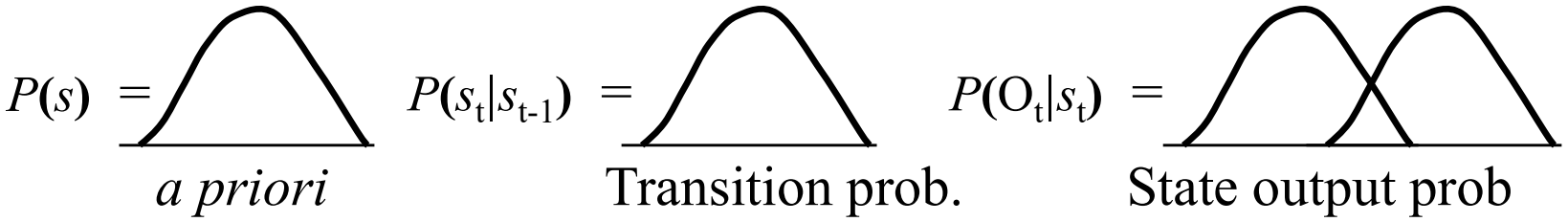
When distributions are not Gaussian



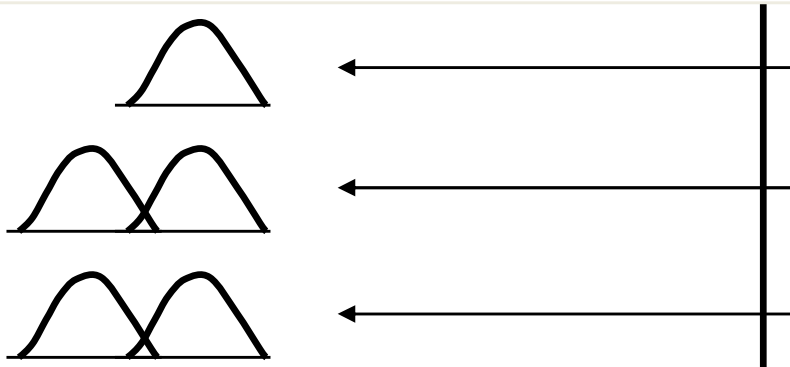
a priori Transition prob. State output prob



When distributions are not Gaussian



a priori Transition prob. State output prob

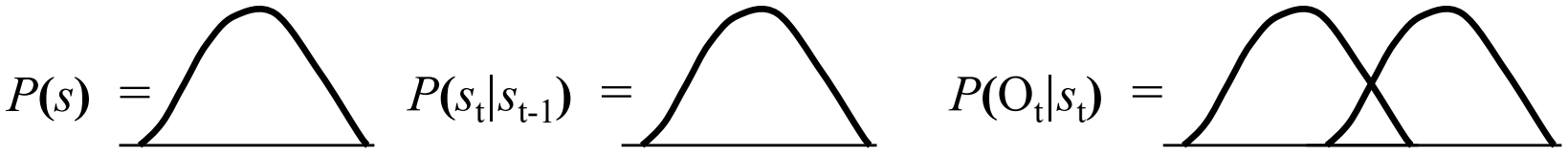


$$P(s_0) = P(s)$$

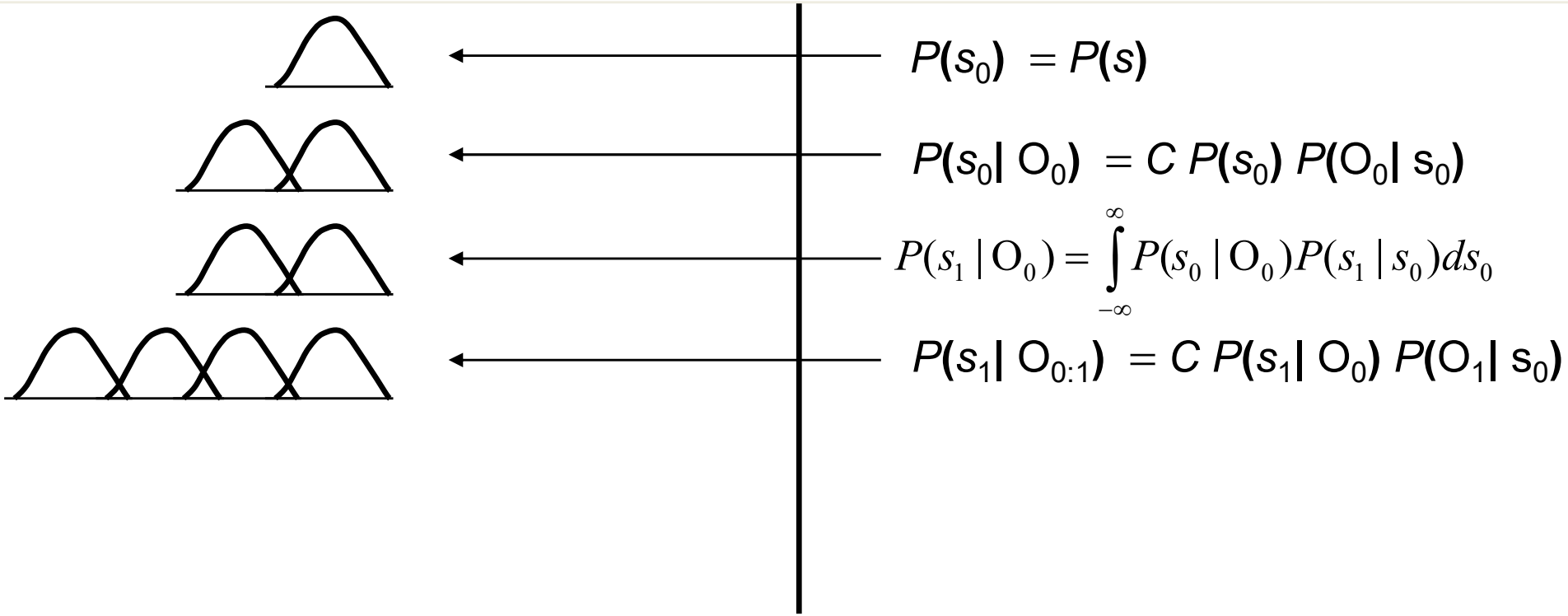
$$P(s_0 | O_0) = C\, P(s_0)\, P(O_0 | s_0)$$

$$P(s_1 | O_0) = \int_{-\infty}^{\infty} P(s_0 | O_0) P(s_1 | s_0) ds_0$$

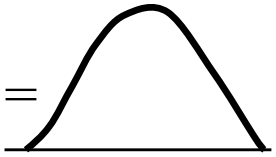
When distributions are not Gaussian



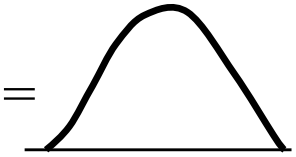
a priori Transition prob. State output prob



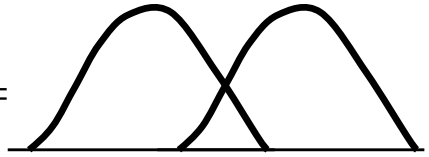
When distributions are not Gaussian

$P(s)$ = 

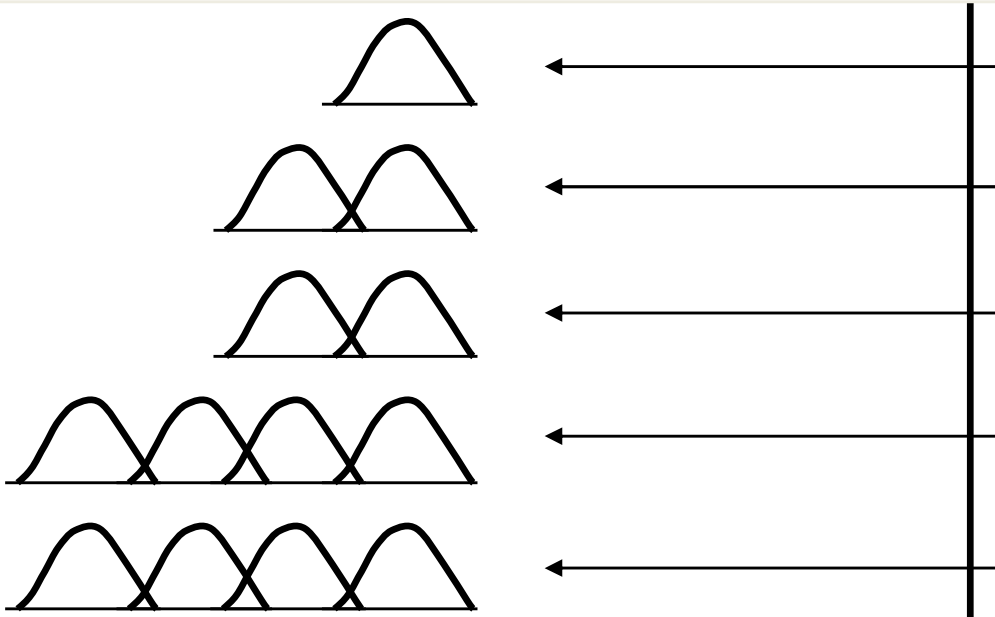
a priori

$P(s_t|s_{t-1})$ = 

Transition prob.

$P(O_t|s_t)$ = 

State output prob



$P(s_0) = P(s)$

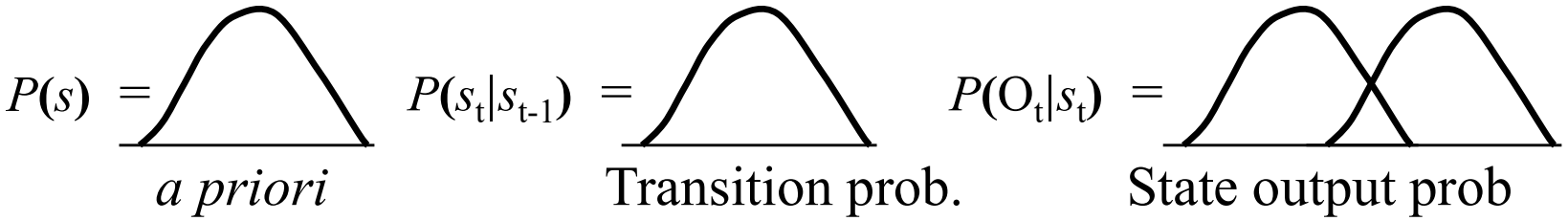
$P(s_0|O_0) = C\,P(s_0)\,P(O_0|s_0)$

$$P(s_1 \mid O_0) = \int_{-\infty}^{\infty} P(s_0 \mid O_0) P(s_1 \mid s_0) ds_0$$

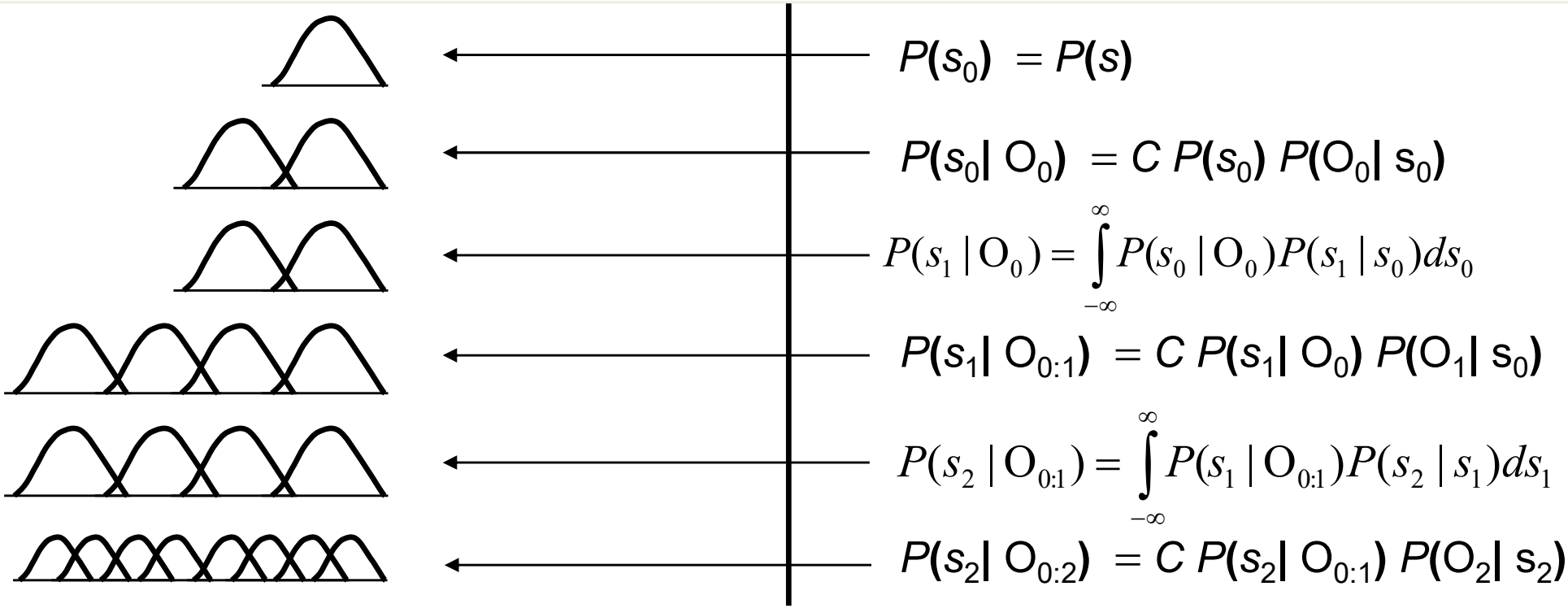
$P(s_1|O_{0:1}) = C\,P(s_1|O_0)\,P(O_1|s_0)$

$$P(s_2 \mid O_{0:1}) = \int_{-\infty}^{\infty} P(s_1 \mid O_{0:1}) P(s_2 \mid s_1) ds_1$$

When distributions are not Gaussian



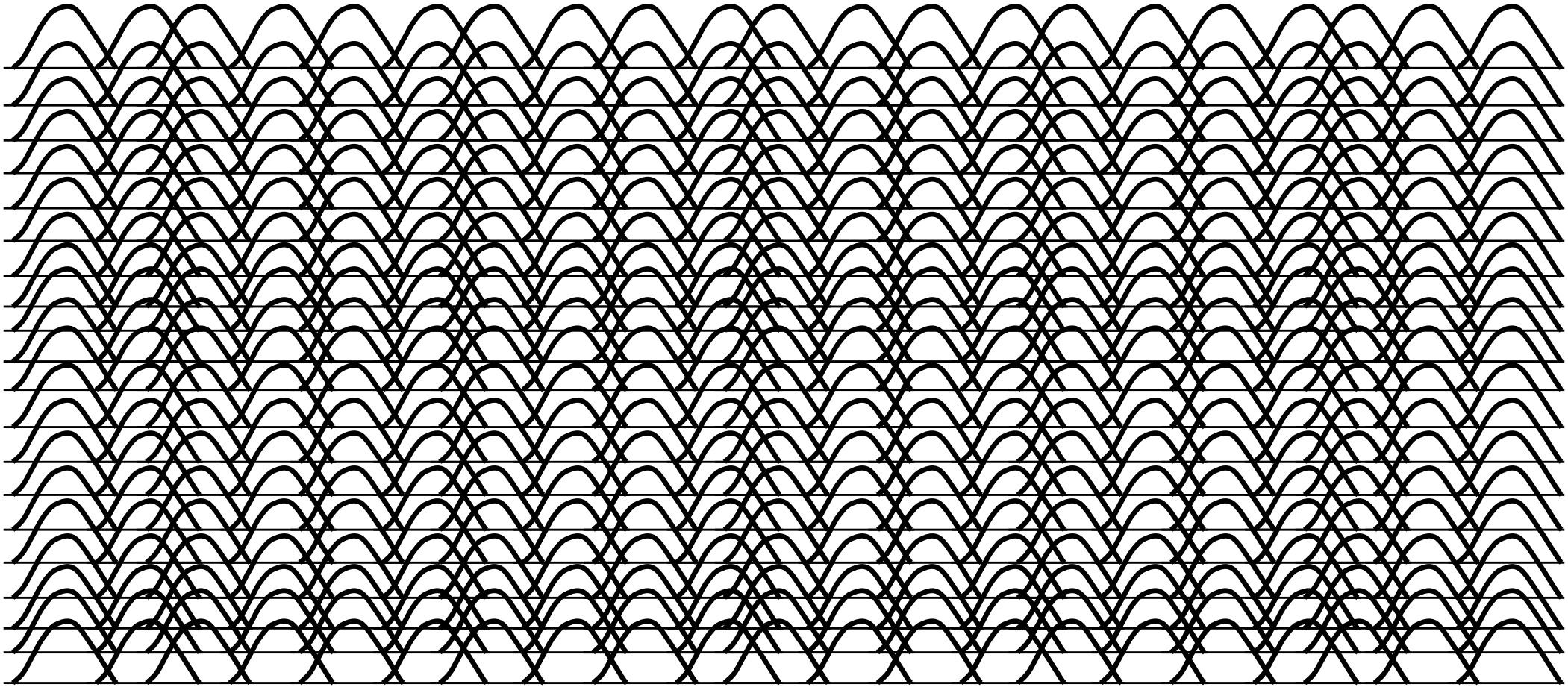
a priori Transition prob. State output prob



When $P(O_t|s_t)$ has more than one Gaussian, after only a few time steps…

When distributions are not Gaussian

$P(s_t \mid \mathrm{O}_{0:t}) =$



We have too many Gaussians for comfort..

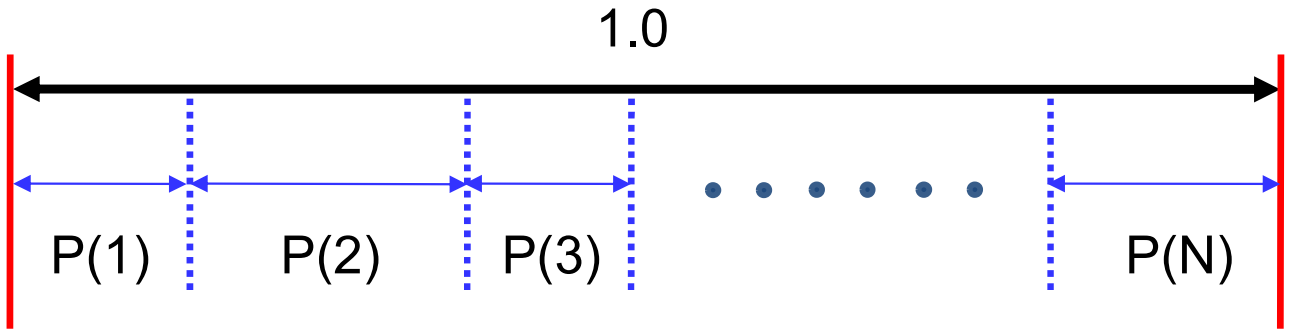
Related Topic: How to sample from a Distribution?

- "Sampling from a Distribution P(x; Γ) with parameters Γ"
- Generate random numbers such that
  - The distribution of a large number of generated numbers is P(x; Γ)
  - The parameters of the distribution are Γ
- Many algorithms to generate RVs from a variety of distributions
  - Generation from a uniform distribution is well studied
  - Uniform RVs used to sample from category distributions
  - Other distributions: Most commonly, transform a uniform RV to the desired distribution

Sampling from a category PDF

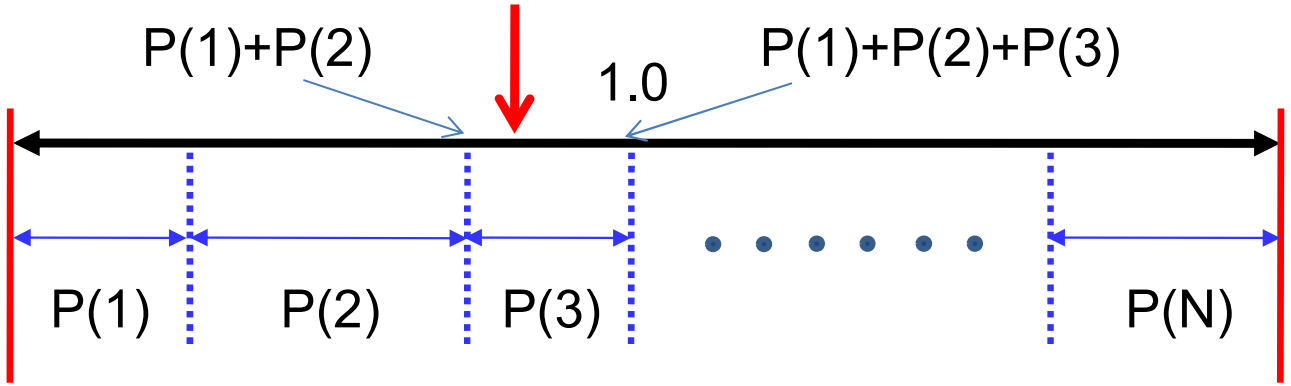
- Given a category PDF over N symbols, with probability of i^{th} symbol = P(i)
- Randomly generate symbols from this distribution
- Can be done by sampling from a uniform distribution

Sampling a category PDF



- Segment a range (0,1) according to the probabilities P(i)
  - The P(i) terms will sum to 1.0

Sampling a category PDF



- Segment a range (0,1) according to the probabilities P(i)
  - The P(i) terms will sum to 1.0
- Randomly generate a number from a uniform distribution
  - Matlab: "rand".
  - Generates a number between 0 and 1 with uniform probability
- If the number falls in the i^{th} segment, select the i^{th} symbol

Related Topic: Sampling from a Gaussian

- Many algorithms
  - Simplest: add many samples from a uniform RV
  - The sum of 12 uniform RVs (uniform in (0,1)) is approximately Gaussian with mean 6 and variance 1
  - For scalar Gaussian, mean μ, std dev σ:

$$x = \sum_{i=1}^{12} r_i - 6$$

- Matlab : x = μ + randn* σ
  - "randn" draws from a Gaussian of mean=0, variance=1

Related Topic: Sampling from a Gaussian

- Multivariate (d-dimensional) Gaussian with mean μ and covariance Θ
  - Compute eigen value matrix Λ and eigenvector matrix E for Θ
    - $\Theta = E \Lambda E^T$
  - Generate d 0-mean unit-variance numbers $x_1..x_d$
  - Arrange them in a vector:

$$X = [x_1 .. x_d]^T$$

  - Multiply X by the square root of Λ and E, add μ

$$Y = \mu + E \, sqrt(\Lambda) \, X$$

Sampling from a Gaussian Mixture

$$\sum_i w_i Gaussian(X; \mu_i, \Theta_i)$$

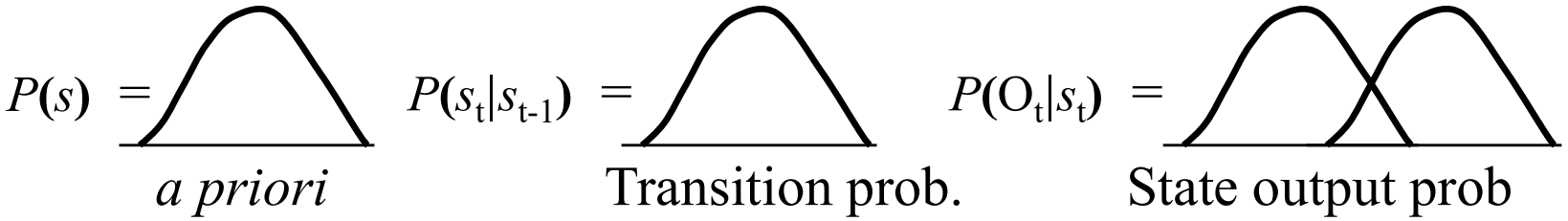
- Select a Gaussian by sampling the multinomial distribution of weights:

$$j \sim Category(w_1, w_2, \ldots)$$

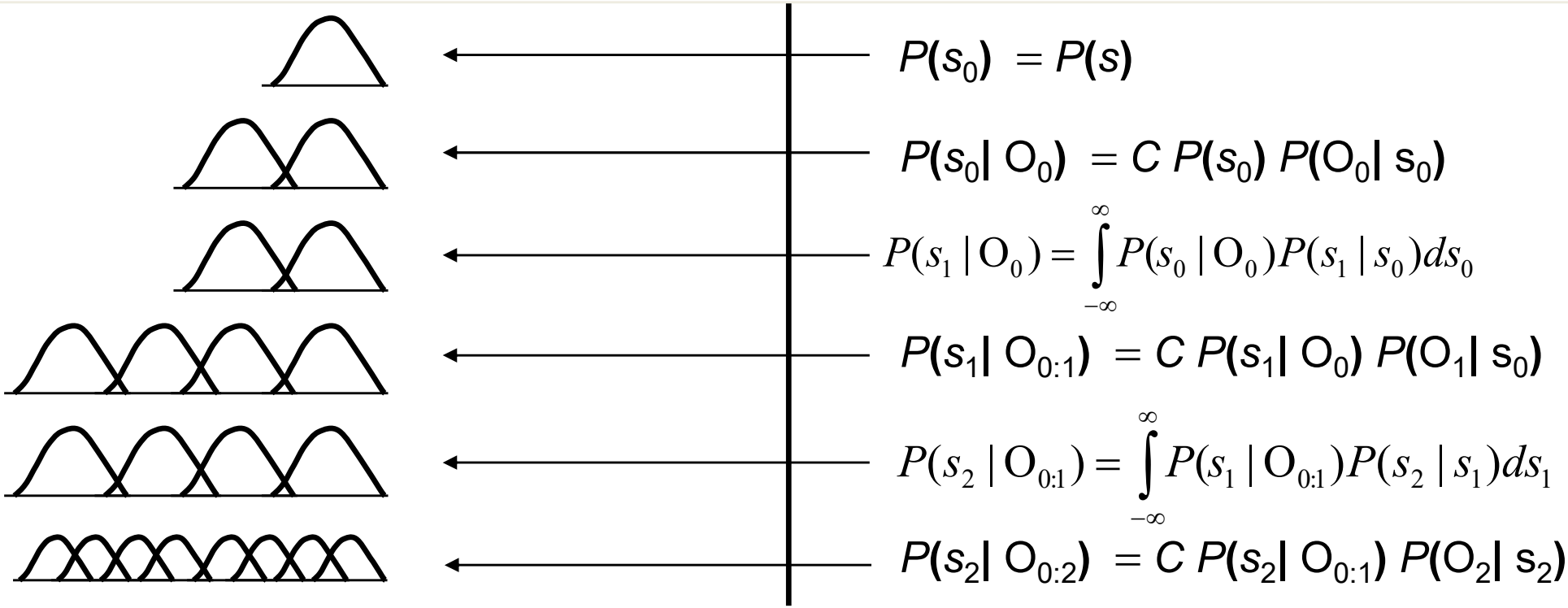
- Sample from the selected Gaussian

$$Gaussian(X; \mu_j, \Theta_j)$$

When distributions are not Gaussian



a priori Transition prob. State output prob

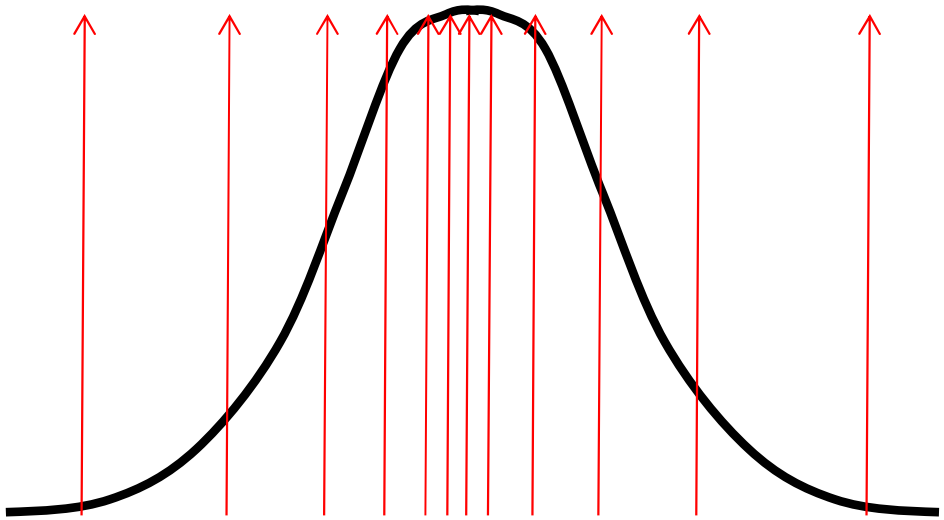


When $P(O_t|s_t)$ has more than one Gaussian, after only a few time steps…

The problem of the exploding distribution

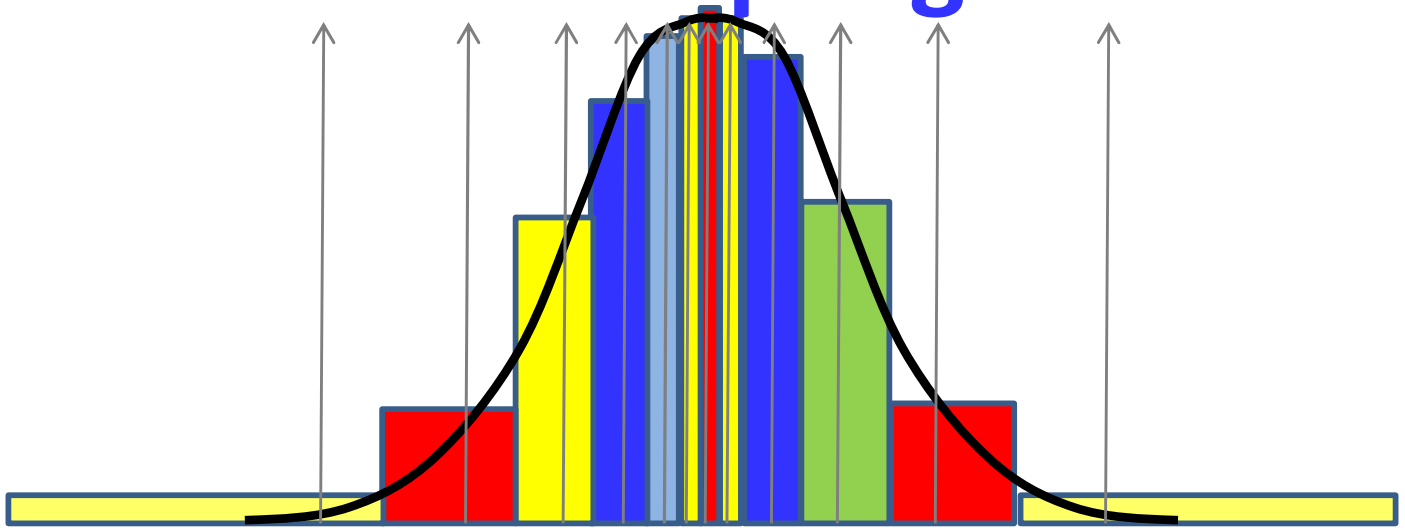
- The complexity of the distribution increases exponentially with time
- This is a consequence of having a *continuous* state space
  - Only Gaussian PDFs propagate without increase of complexity
- *Discrete-state* systems do not have this problem
  - The number of states in an HMM stays fixed
  - However, discrete state spaces are too coarse
- Solution: Combine the two concepts
  - *Discretize* the state space dynamically

Discrete approximation to a distribution



- A large-enough collection of randomly-drawn samples from a distribution will approximately quantize the space of the random variable into equi-probable regions
  - We have more random samples from high-probability regions and fewer samples from low-probability reigons

Discrete approximation: Random sampling



- A PDF can be approximated as a uniform probability distribution over randomly drawn samples
  - Since each sample represents approximately the same probability mass (1/M if there are M samples)

$$P(x) \approx \frac{1}{M} \sum_{i=0}^{M-1} \delta(x - x_i)$$

Note: Properties of a discrete distribution

$$P(x) \approx \frac{1}{M}\sum_{i=0}^{M-1}\delta(x - x_i)$$

$$P(x)P(y \mid x) \propto \sum_{i=0}^{M-1} P(y \mid x_i)\delta(x - x_i)$$

- The product of a discrete distribution with another distribution is simply a weighted discrete probability

$$P(x) \approx \sum_{i=0}^{M-1} w_i\delta(x - x_i)$$

$$\int_{-\infty}^{\infty} P(x)P(y \mid x)dx = \sum_{i=0}^{M-1} w_i P(y \mid x_i)$$

- The integral of the product is a mixture distribution

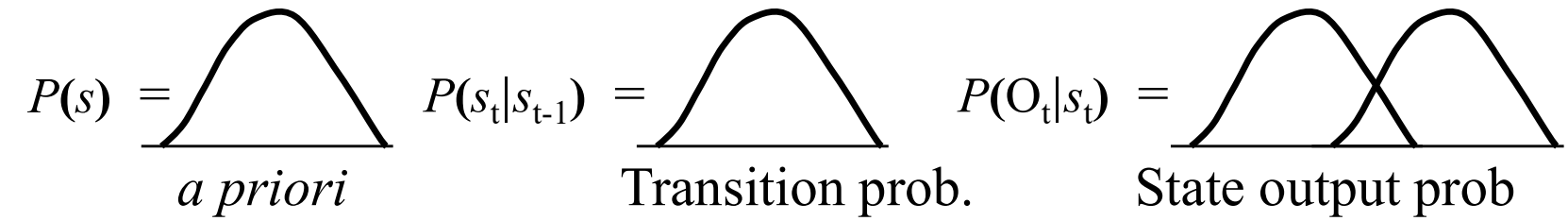
Discretizing the state space

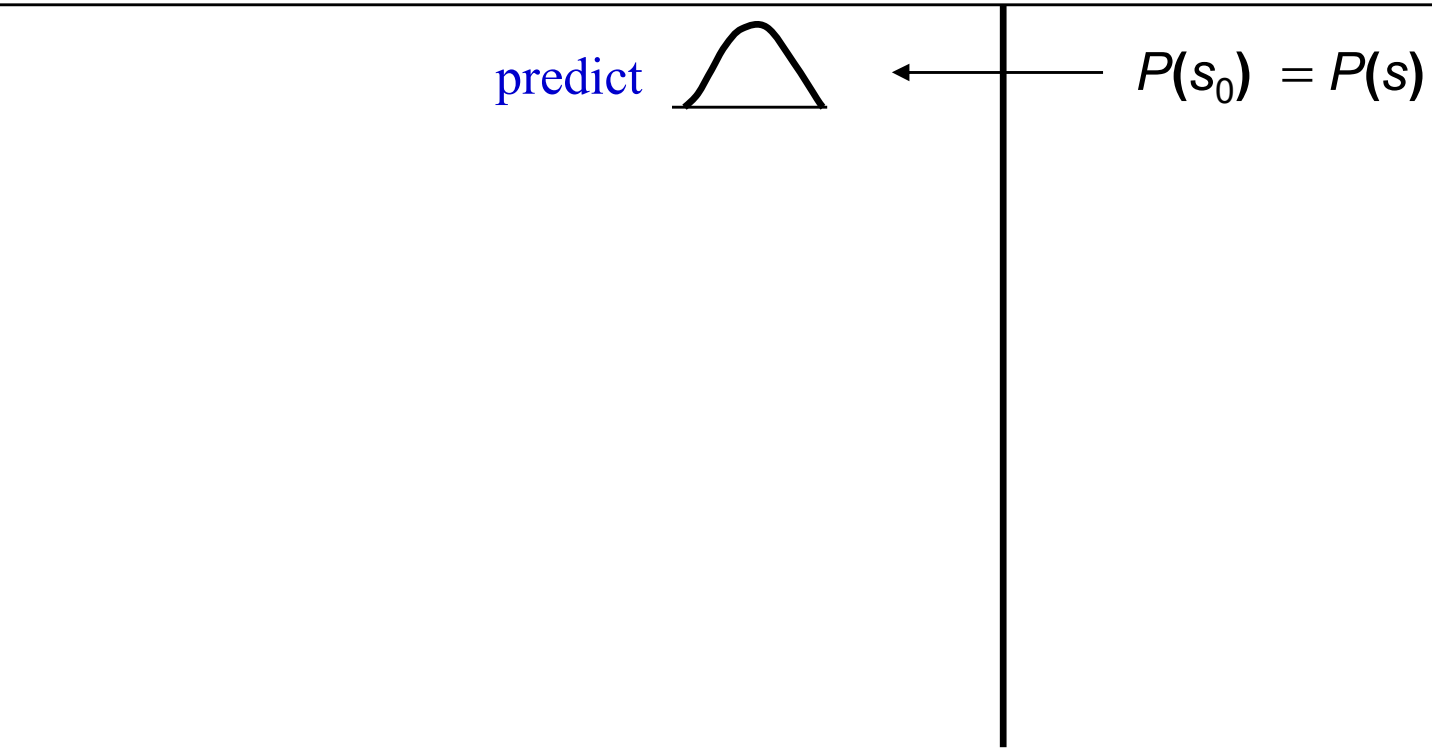
- At each time, discretize the predicted state space

$$P(s_t \mid o_{0:t}) \approx \frac{1}{M} \sum_{i=0}^{M-1} \delta(s_t - s_i)$$

  - s_i are randomly drawn samples from $P(s_t \mid o_{0:t})$
- Propagate the discretized distribution

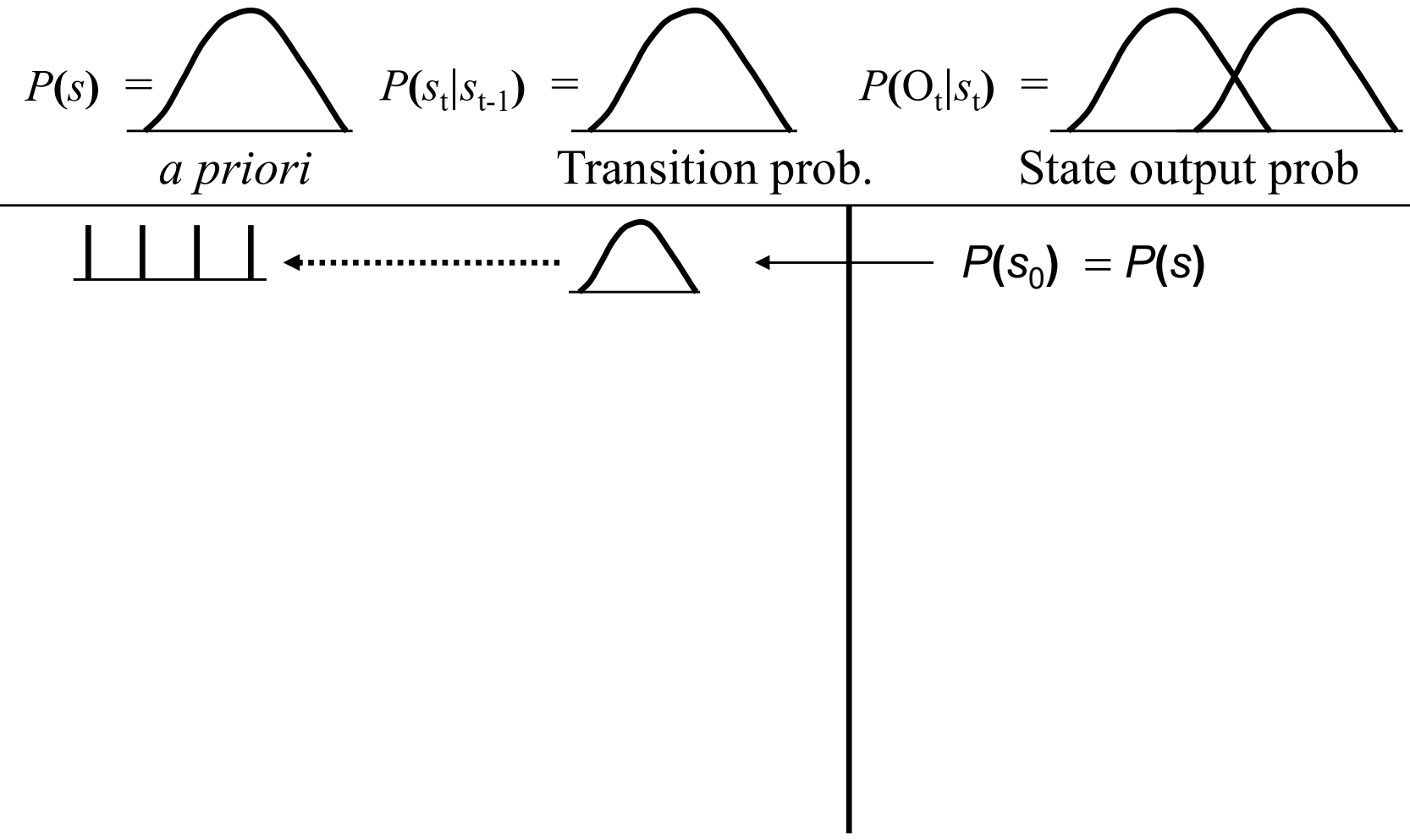
Particle Filtering





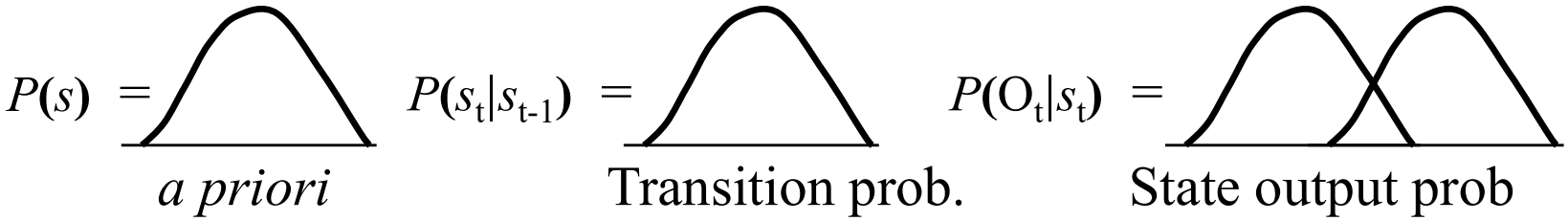
Assuming that we only generate ***FOUR*** samples from the predicted distributions

Particle Filtering

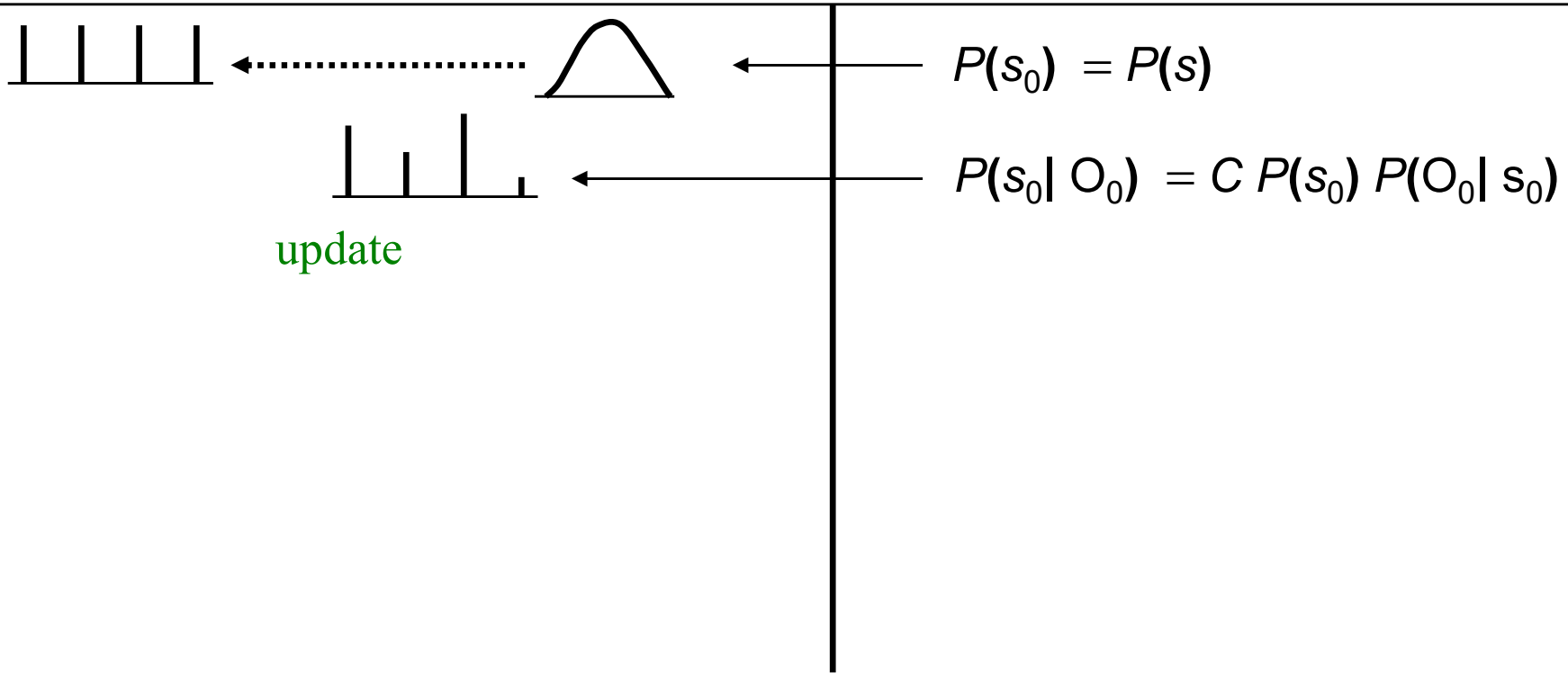


Assuming that we only generate ***FOUR*** samples from the predicted distributions

Particle Filtering

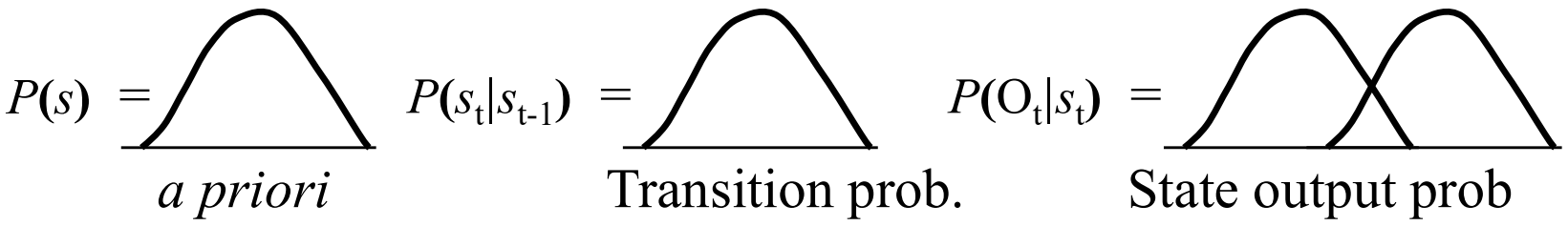


a priori Transition prob. State output prob

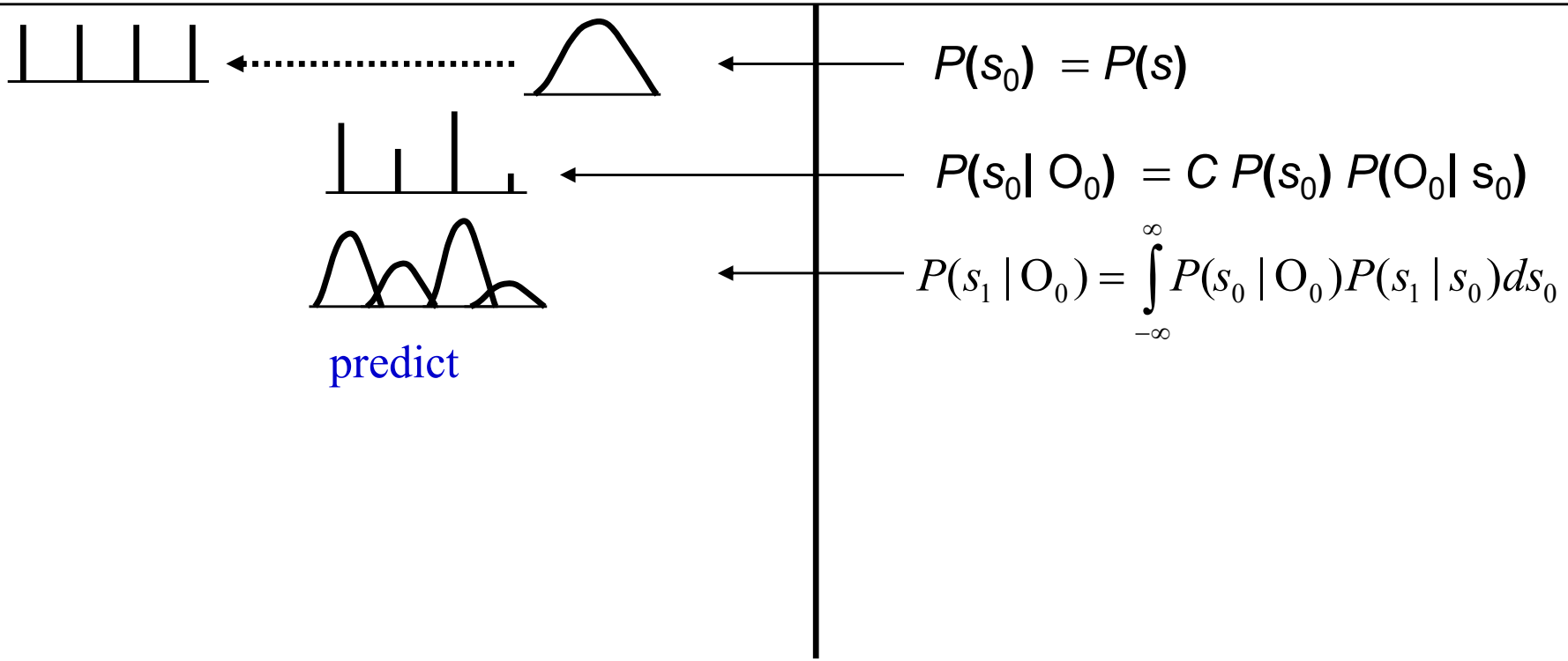


Assuming that we only generate ***FOUR*** samples from the predicted distributions

Particle Filtering

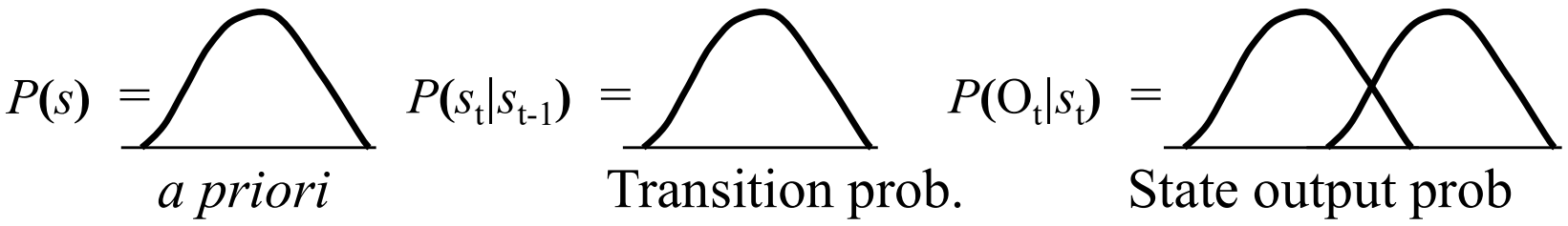


a priori Transition prob. State output prob

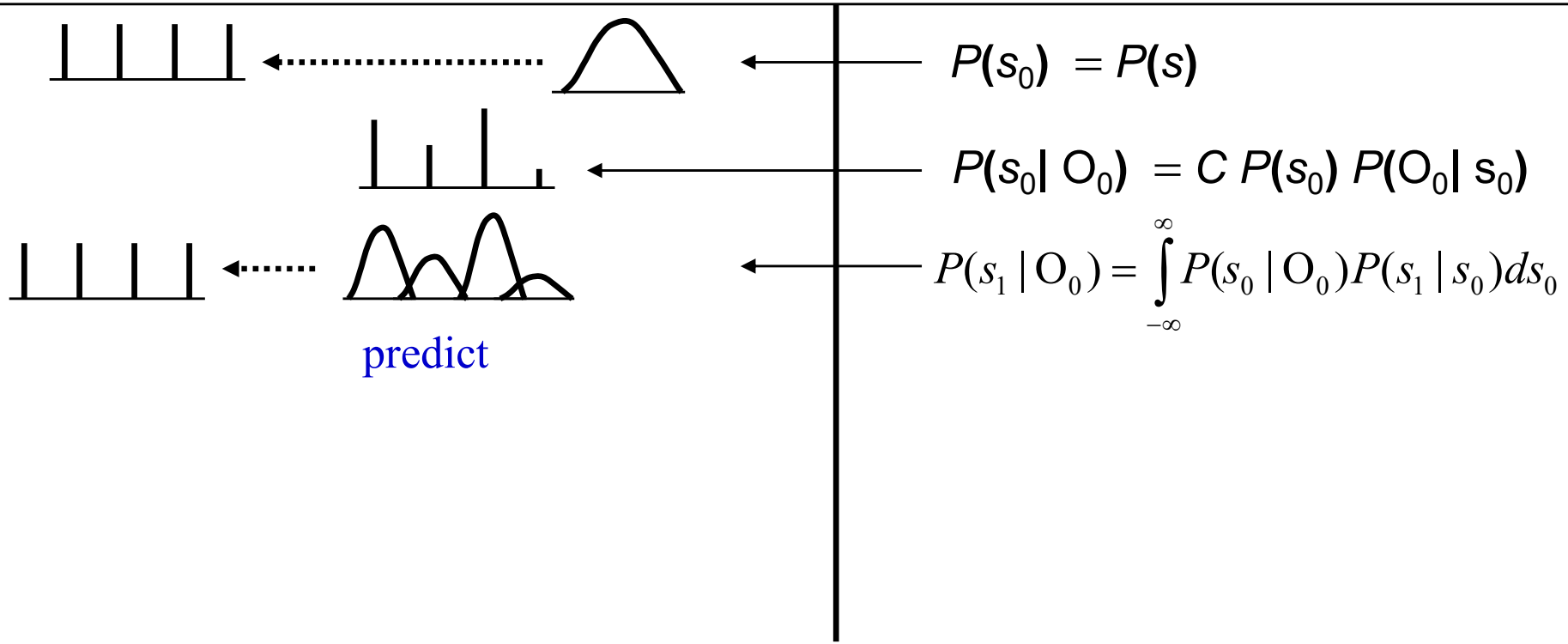


Assuming that we only generate ***FOUR*** samples from the predicted distributions

Particle Filtering

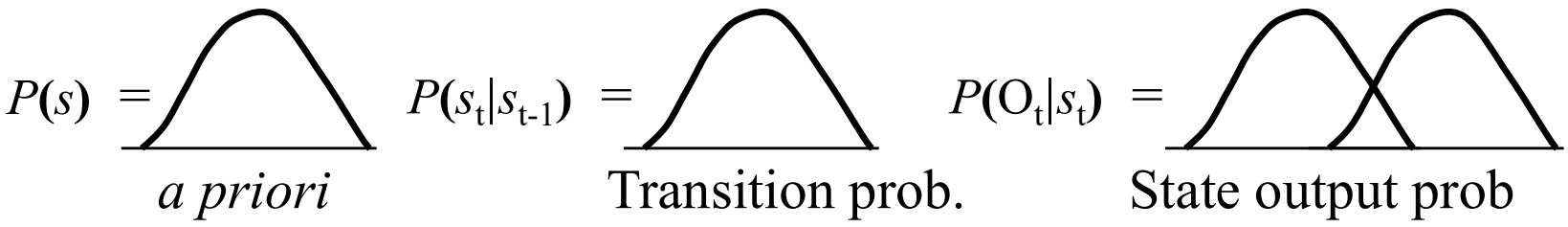


a priori Transition prob. State output prob

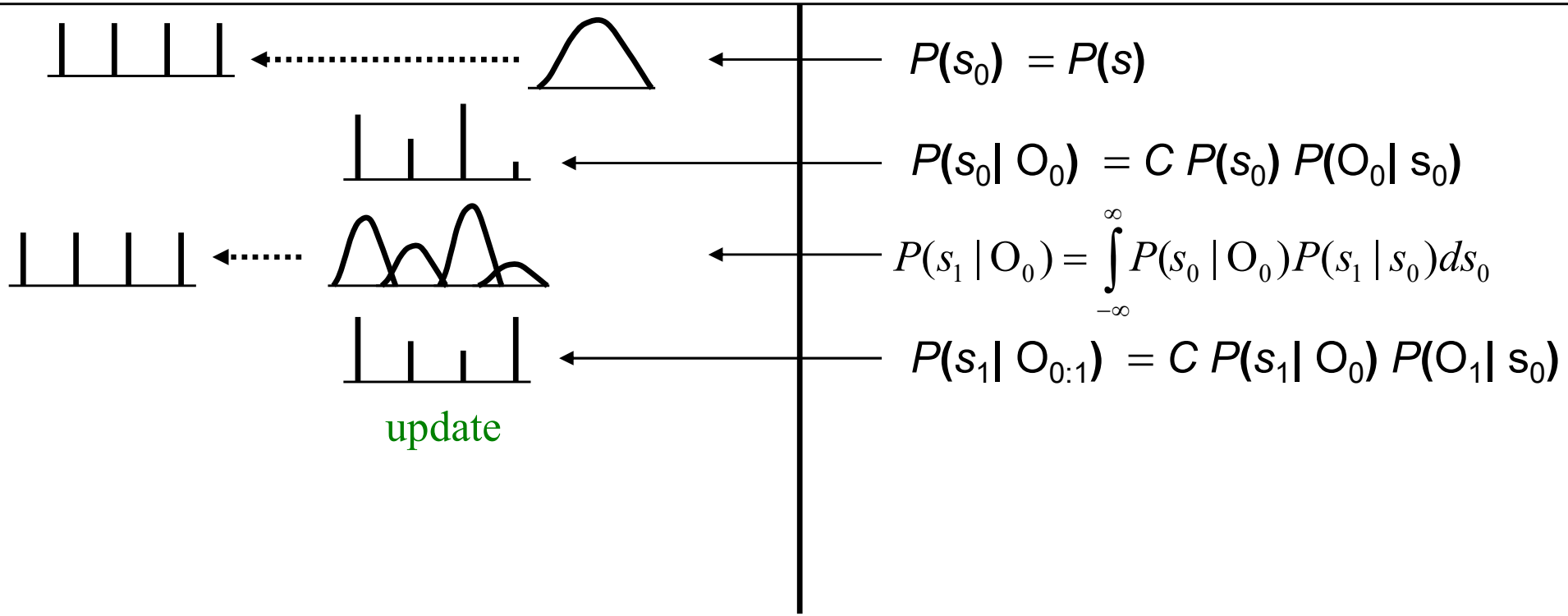


Assuming that we only generate ***FOUR*** samples from the predicted distributions

Particle Filtering

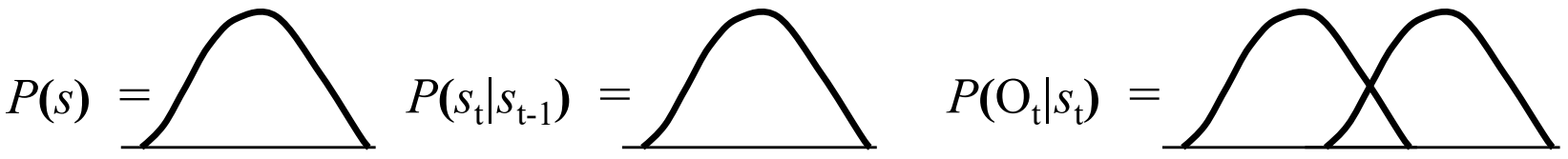


a priori Transition prob. State output prob

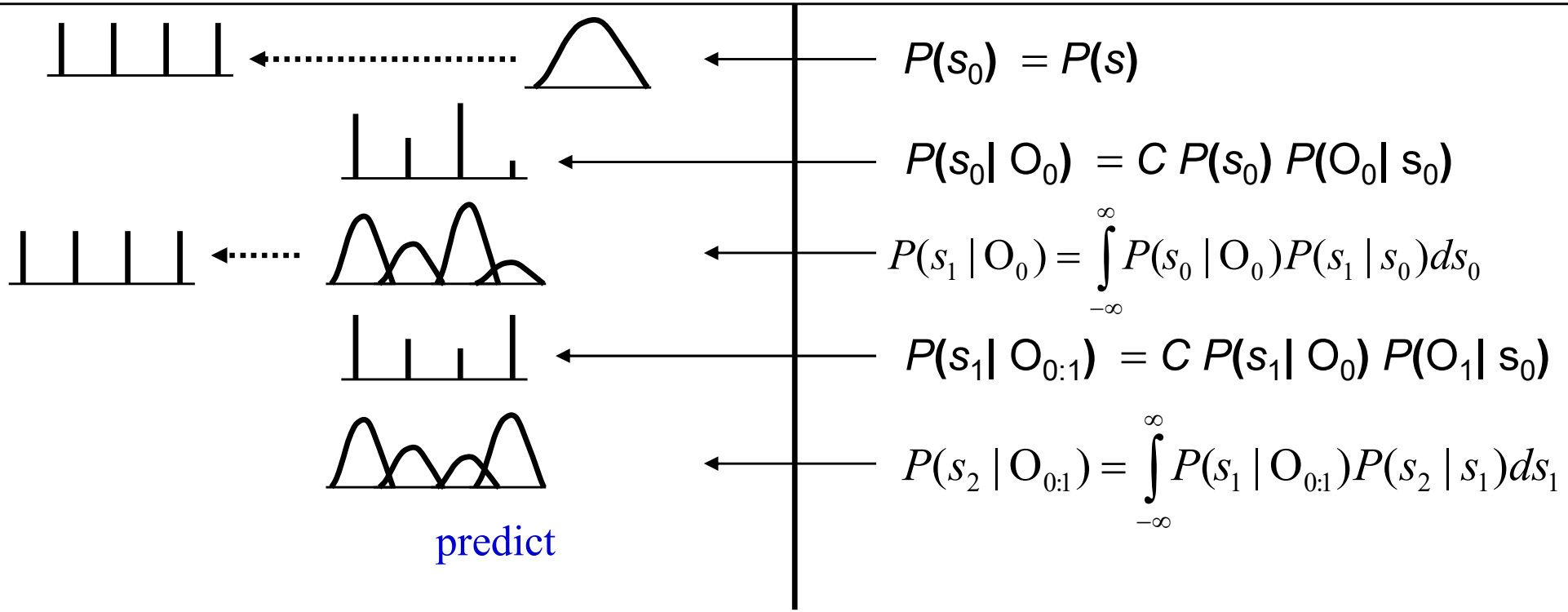


Assuming that we only generate ***FOUR*** samples from the predicted distributions

Particle Filtering

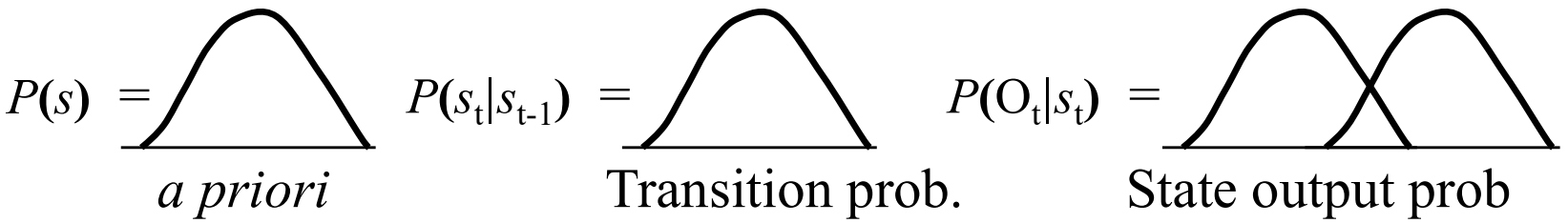


a priori Transition prob. State output prob

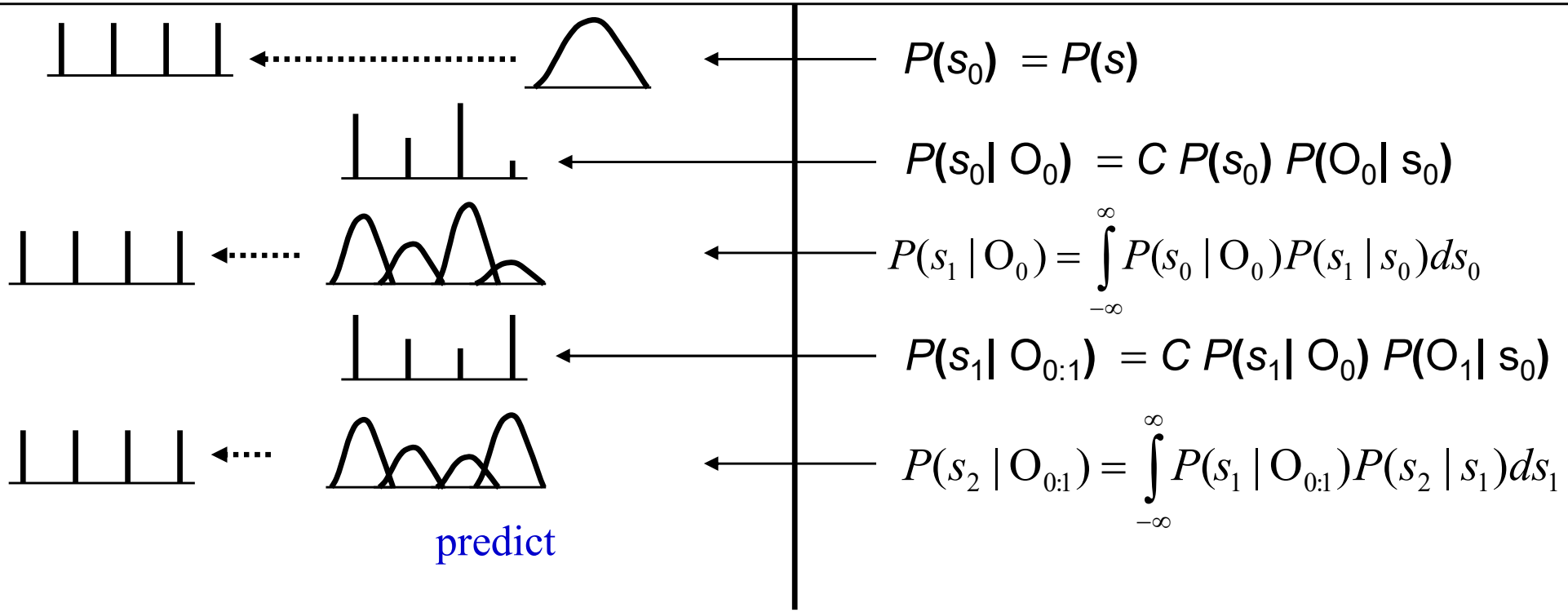


Assuming that we only generate ***FOUR*** samples from the predicted distributions

Particle Filtering

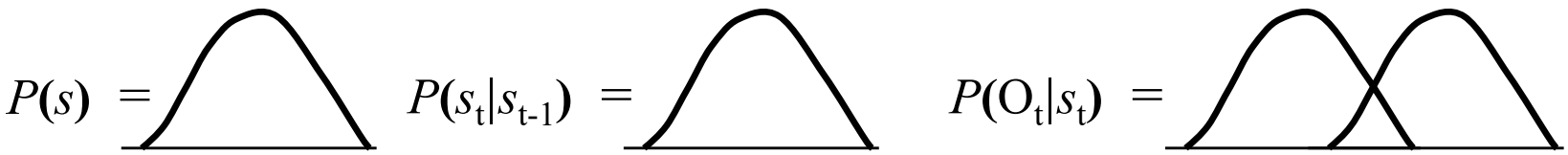


a priori Transition prob. State output prob

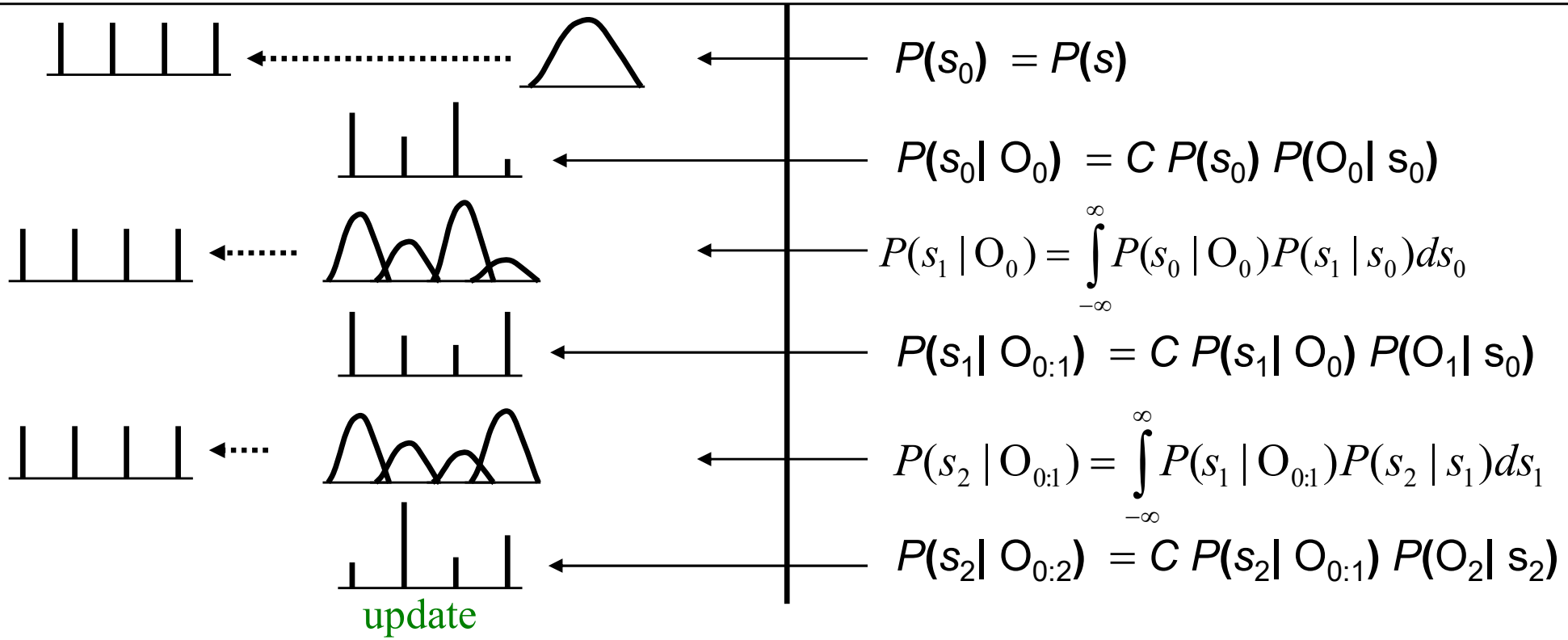


Assuming that we only generate ***FOUR*** samples from the predicted distributions

Particle Filtering



a priori Transition prob. State output prob

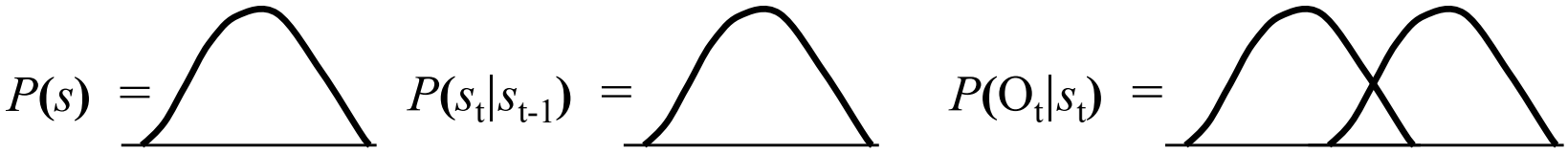


Assuming that we only generate ***FOUR*** samples from the predicted distributions

Particle Filtering

- Discretize state space at the prediction step
  - By sampling the continuous predicted distribution
    - If appropriately sampled, all generated samples may be considered to be equally probable
  - Sampling results in a **discrete** uniform distribution
- Update step updates the distribution of the quantized state space
  - Results in a **discrete** non-uniform distribution
- Predicted state distribution for the next time instant will again be continuous
  - Must be **discretized** again by sampling
- At any step, the current state distribution will not have more components than the number of samples generated at the previous sampling step
  - The complexity of distributions remains constant

Particle Filtering



a priori Transition prob. State output prob

predict

Prediction at time t:

$$P(s_t \mid O_{0:t-1}) = \int_{-\infty}^{\infty} P(s_{t-1} \mid O_{0:t-1}) P(s_t \mid s_{t-1}) ds_{t-1}$$

Update at time t:

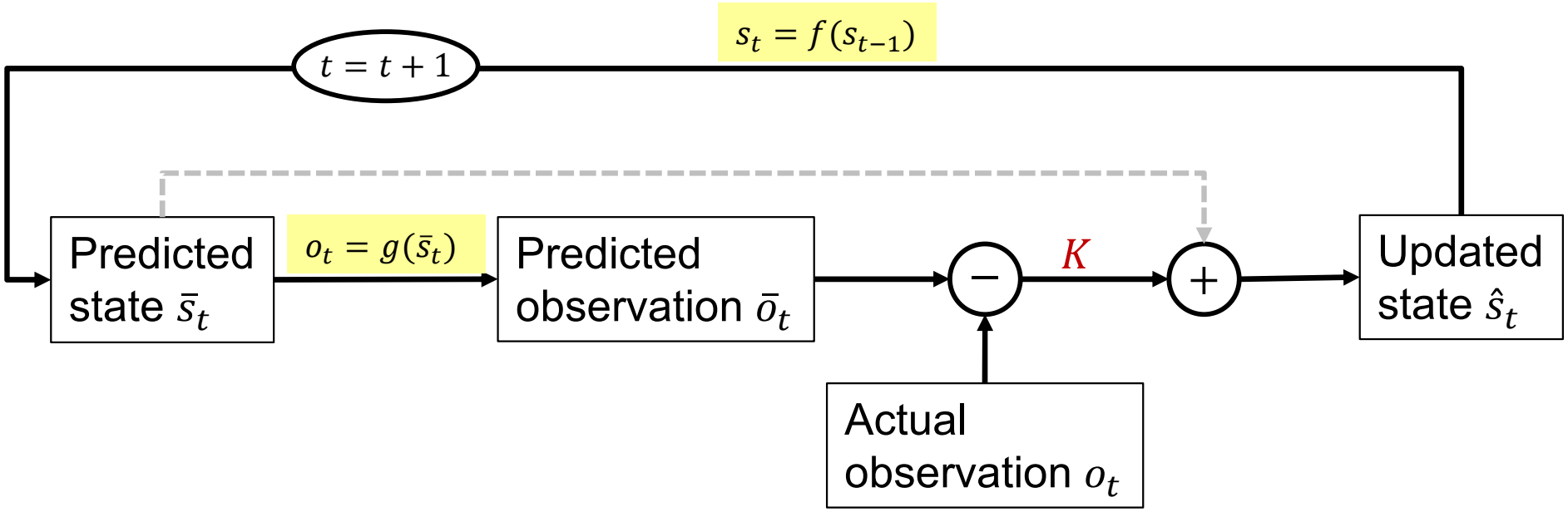
update

$$P(s_t \mid O_{0:t}) = CP(s_t \mid O_{0:t-1}) P(O_t \mid s_t)$$

Number of mixture components in predicted distribution governed by number of samples in discrete distribution

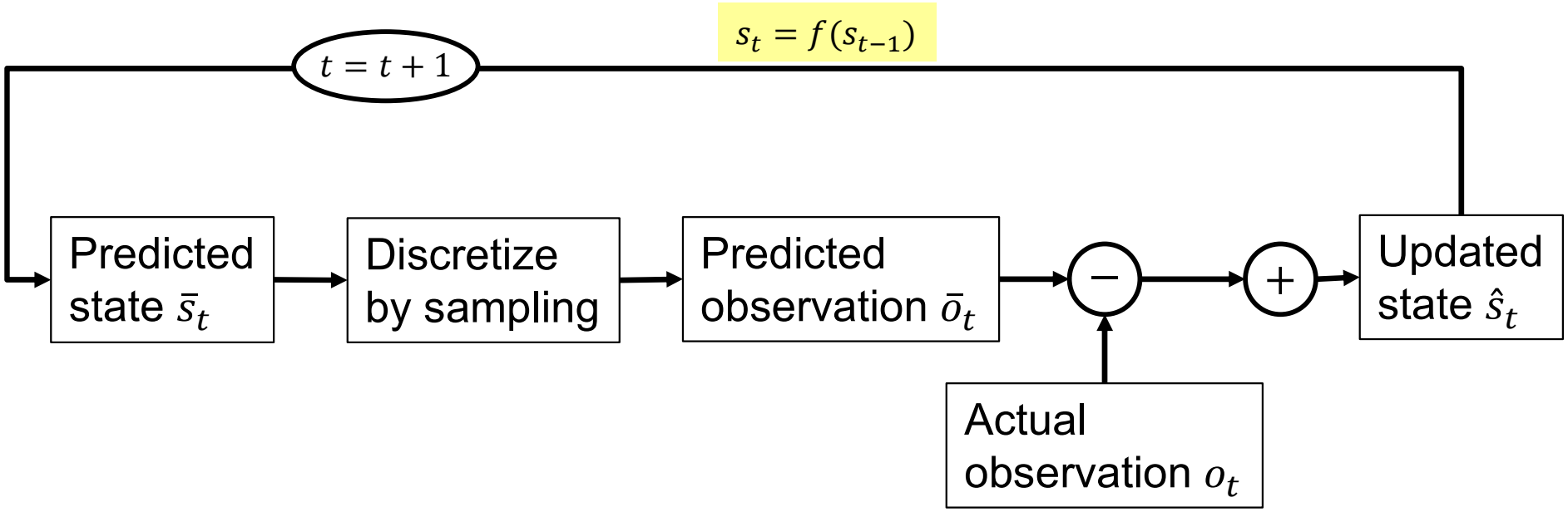
By deriving a small (100-1000) number of samples at each time instant, all distributions are kept manageable

Standard KF/EKF



- Predict state
- Predict measurement
- Compute measurement error
- Update state

Particle Filter



- Predict state distribution
  - Sample to discretize
- Predict measurement distribution
  - Compute measurement error
- Update discretized state distribution

Particle Filtering

$$o_t = g(s_t) + \gamma \qquad s_t = f(s_{t-1}) + \varepsilon$$

$$P_\gamma(\gamma) \qquad P_\varepsilon(\varepsilon)$$

- At t = 0, sample the initial state distribution

$$P(s_0 \mid o_{-1}) = P(s_0) \approx \frac{1}{M}\sum_{i=0}^{M-1}\delta(s_0 - \bar{s}_i^0) \text{ where } \bar{s}_i^0 \leftarrow P_0(s)$$

- Update the state distribution with the observation

$$P(s_t \mid o_{0:t}) = C\sum_{i=0}^{M-1} P_\gamma(o_t - g(\bar{s}_i^t))\delta(s_t - \bar{s}_i^t)$$

$$C = \frac{1}{\sum_{i=0}^{M-1} P_\gamma(o_t - g(\bar{s}_i^t))}$$

Particle Filtering

$$o_t = g(s_t) + \gamma$$

$$s_t = f(s_{t-1}) + \varepsilon$$

$$P_\gamma(\gamma)$$

$$P_\varepsilon(\varepsilon)$$

- Predict the state distribution at the next time

$$P(s_t \mid o_{0:t-1}) = C \sum_{i=0}^{M-1} P_\gamma(o_{t-1} - g(\bar{s}_i^{t-1})) P_\varepsilon(s_t - f(\bar{s}_i^{t-1}))$$

- Sample the predicted state distribution

$$P(s_t \mid o_{0:t-1}) \approx \frac{1}{M} \sum_{i=0}^{M-1} \delta(s_t - \bar{s}_i^t) \text{ where } \bar{s}_i^t \leftarrow P(s_t \mid o_{0:t-1})$$

Particle Filtering

$$o_t = g(s_t) + \gamma \quad P_\gamma(\gamma) \qquad s_t = f(s_{t-1}) + \varepsilon \quad P_\varepsilon(\varepsilon)$$

- Predict the state distribution at t

$$P(s_t \mid o_{0:t-1}) = C \sum_{i=0}^{M-1} P_\gamma(o_{t-1} - g(\bar{s}_i^{t-1})) P_\varepsilon(s_t - f(\bar{s}_i^{t-1}))$$

- Sample the predicted state distribution at t

$$P(s_t \mid o_{0:t-1}) \approx \frac{1}{M} \sum_{i=0}^{M-1} \delta(s_t - \bar{s}_i^t) \text{ where } \bar{s}_i^t \leftarrow P(s_t \mid o_{0:t-1})$$

- Update the state distribution at t

$$P(s_t \mid o_{0:t}) = C \sum_{i=0}^{M-1} P_\gamma(o_t - g(\bar{s}_i^t)) \delta(s_t - \bar{s}_i^t)$$

$$C = \frac{1}{\sum_{i=0}^{M-1} P_\gamma(o_t - g(\bar{s}_i^t))}$$

Estimating a state

- The algorithm gives us a discrete updated distribution over states:

$$P(s_t \mid o_{0:t}) = C\sum_{i=0}^{M-1} P_\gamma(o_t - g(\bar{s}_i^t))\delta(s_t - \bar{s}_i^t)$$

- The actual state can be estimated as the mean of this distribution

$$\hat{s}_t = C\sum_{i=0}^{M-1} \bar{s}_i^t P_\gamma(o_t - g(\bar{s}_i^t))$$

- Alternately, it can be the most likely sample

$$\hat{s}_t = \bar{s}_j^t \;:\; j = \arg\max_i P_\gamma(o_t - g(\bar{s}_i^t))$$

Simulations with a Linear Model

$$s_t = s_{t-1} + \varepsilon_t \qquad o_t = s_t + x_t$$

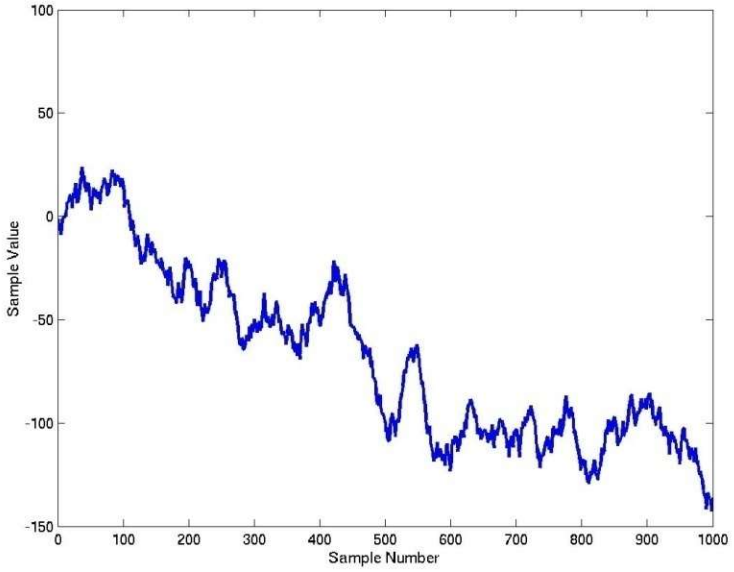
- ε_t has a Gaussian distribution with 0 mean, known variance
- x_t has a mixture Gaussian distribution with known parameters
- Simulation:
  - Generate state sequence S_t from model
  - Generate sequence of X_t from model with one X_t term for every S_t term
  - Generate observation sequence O_t from S_t and X_t
  - Attempt to estimate S_t from O_t

Simulation: Synthesizing data

Generate state sequence according to:

$$s_t = s_{t-1} + \varepsilon_t$$

ε_t is Gaussian with mean 0 and variance 10



Simulation: Synthesizing data

Generate state sequence according to: $s_t = s_{t-1} + \varepsilon_t$

ε_t is Gaussian with mean 0 and variance 10

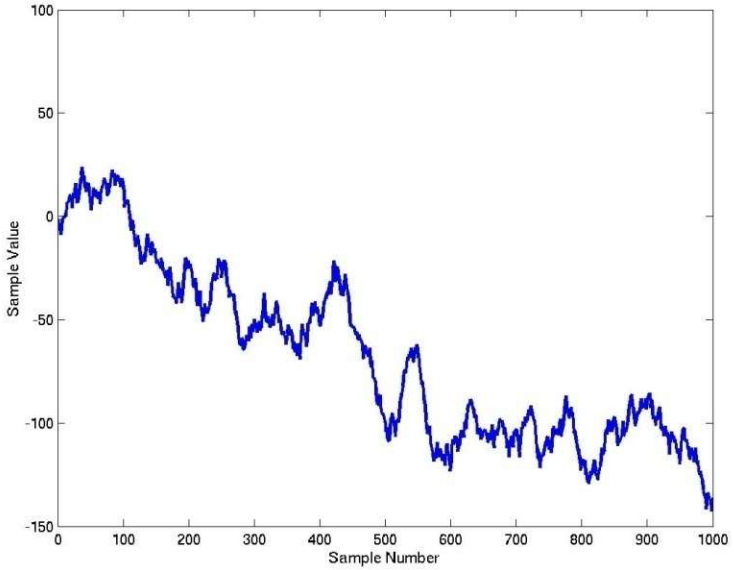
Generate observation sequence from state sequence according to: $o_t = s_t + x_t$

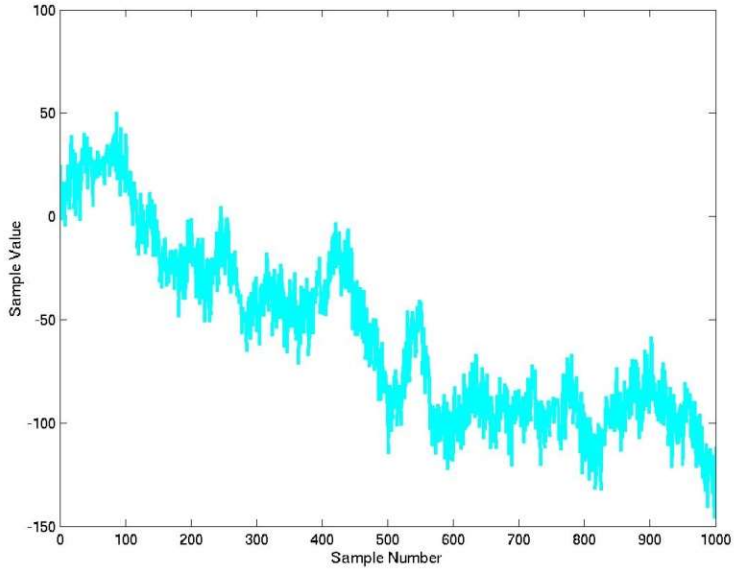
x_t is mixture Gaussian with parameters:

Means = [-4, 0, 4, 8, 12, 16, 18, 20]

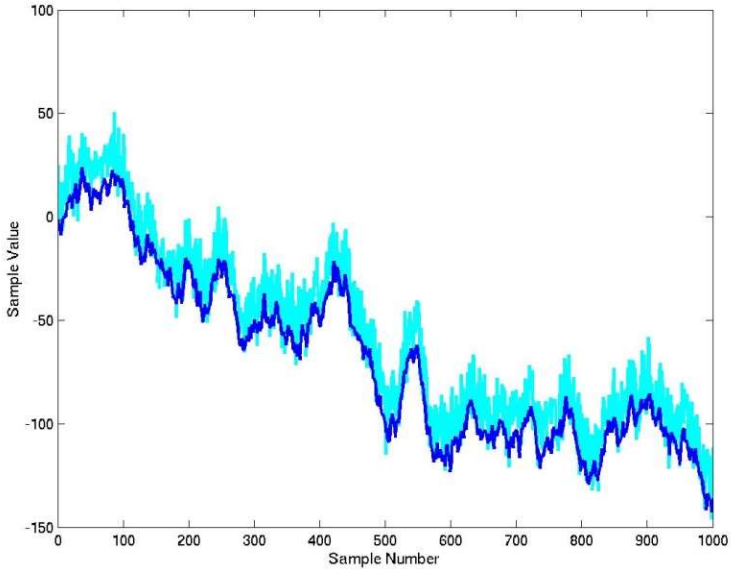
Variances = [10, 10, 10, 10, 10, 10, 10, 10]

Mixture weights = [0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125]



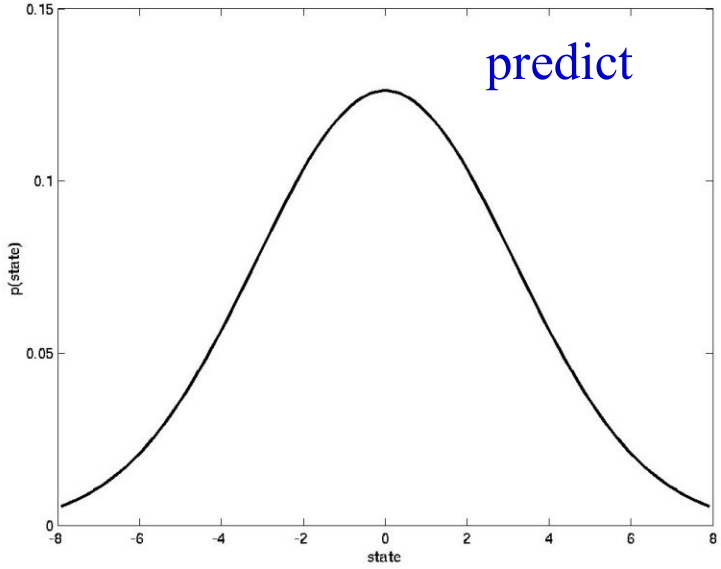


Simulation: Synthesizing data

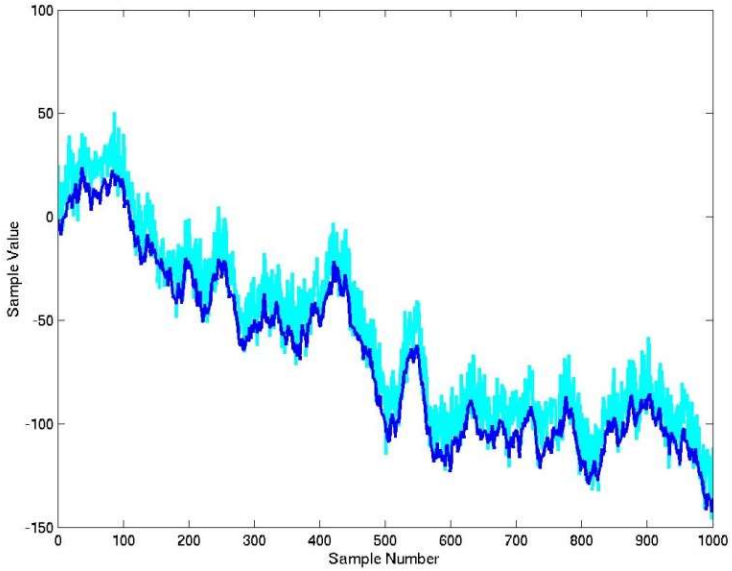


Combined figure for more compact representation

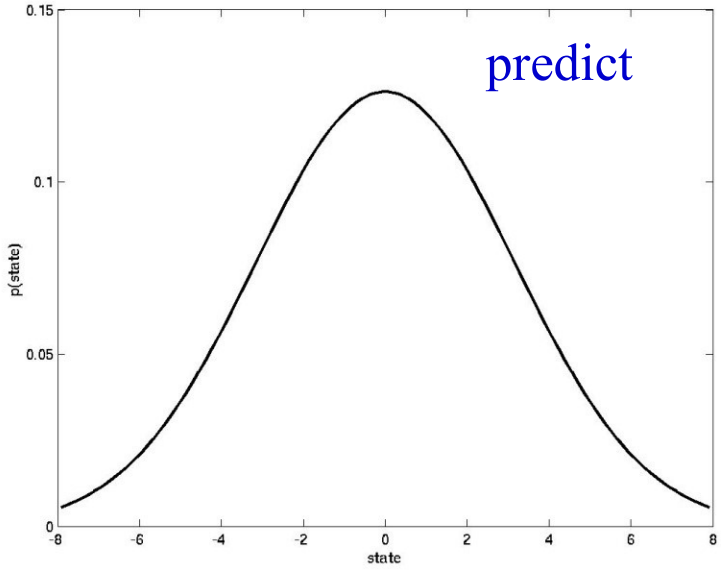
SIMULATION: TIME = 1

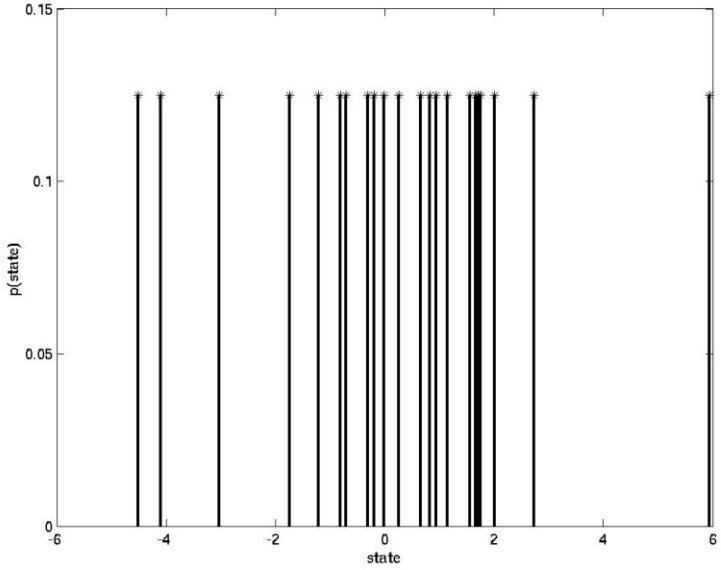


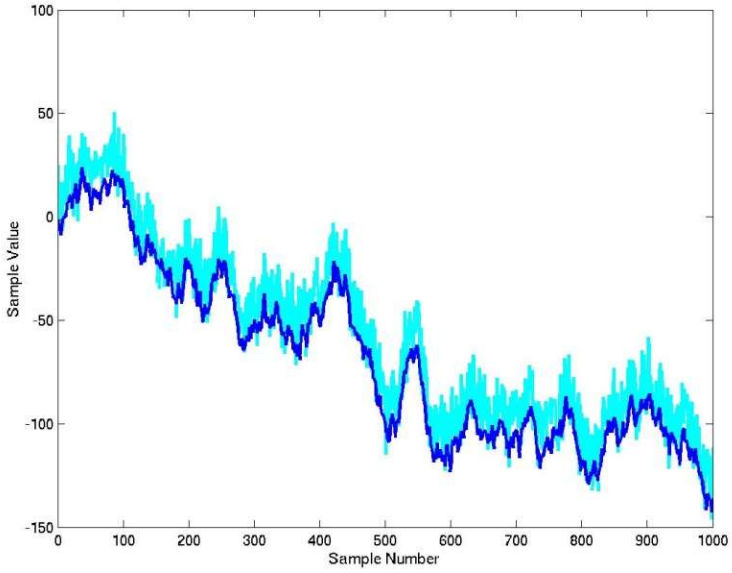
PREDICTED STATE DISTRIBUTION AT TIME = 1



SIMULATION: TIME = 1

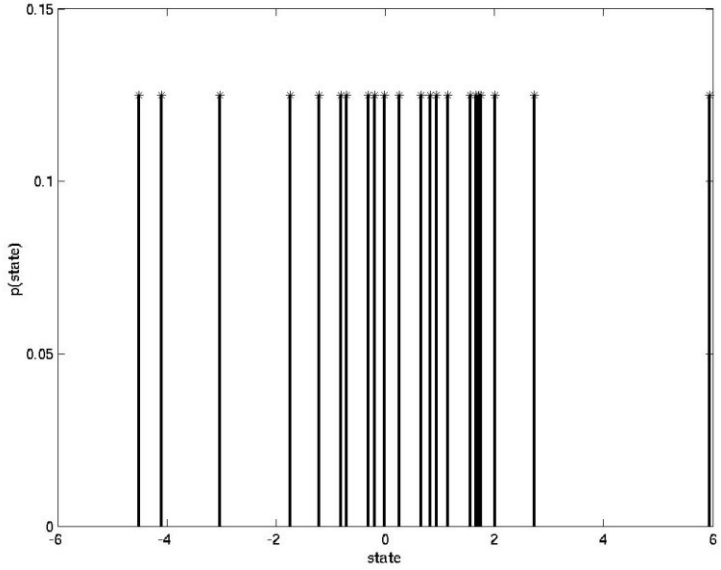


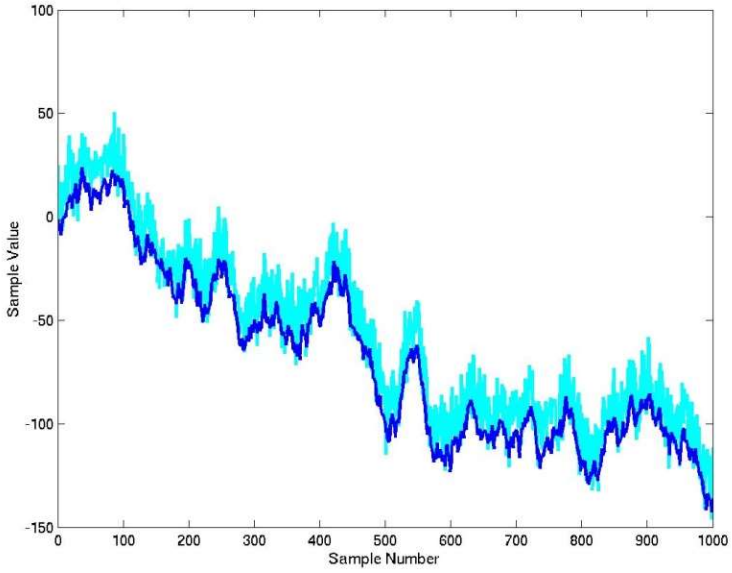




SAMPLED VERSION OF PREDICTED STATE DISTRIBUTION AT TIME = 1

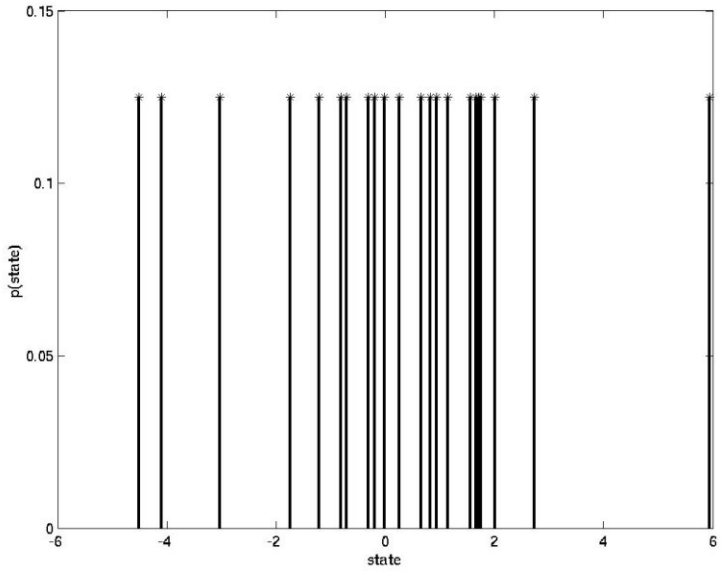
SIMULATION: TIME = 1

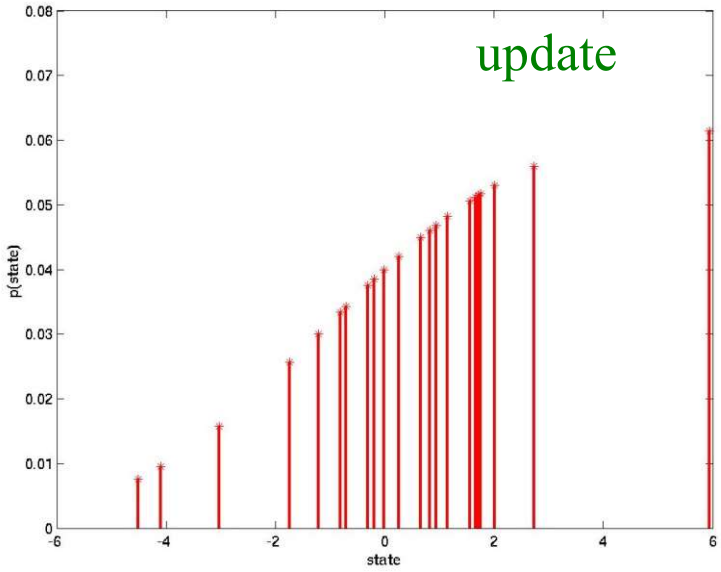


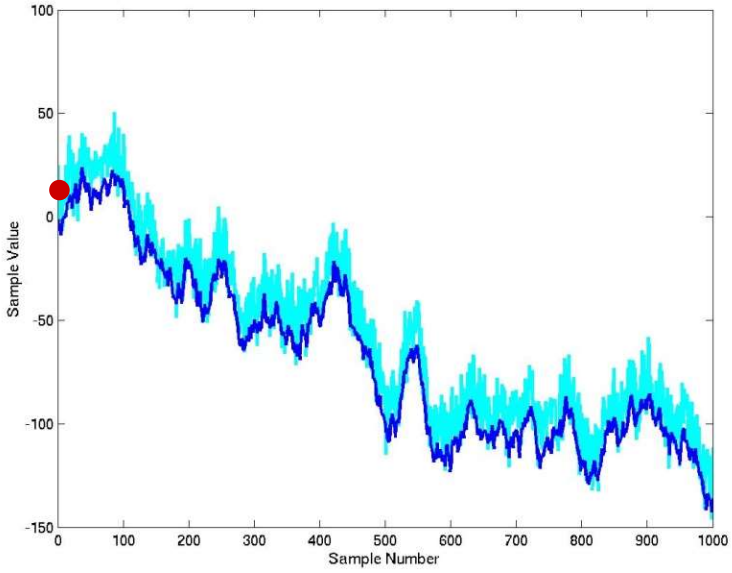


SAMPLED VERSION OF PREDICTED STATE DISTRIBUTION AT TIME = 1

SIMULATION: TIME = 1

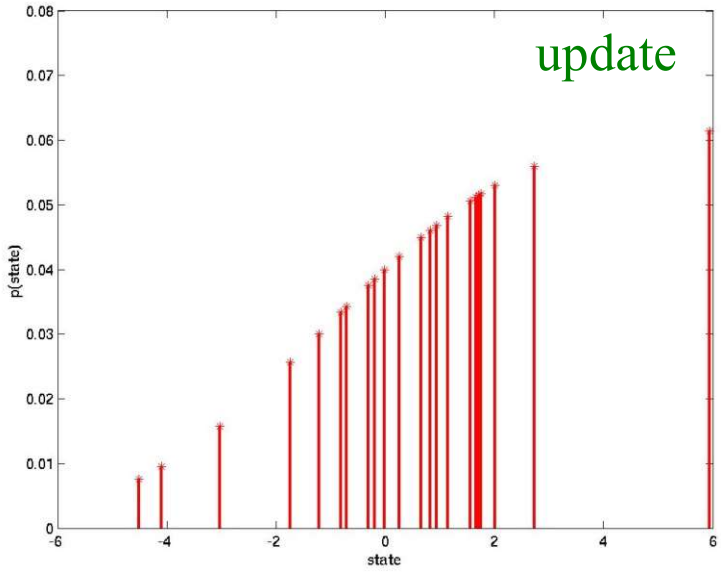


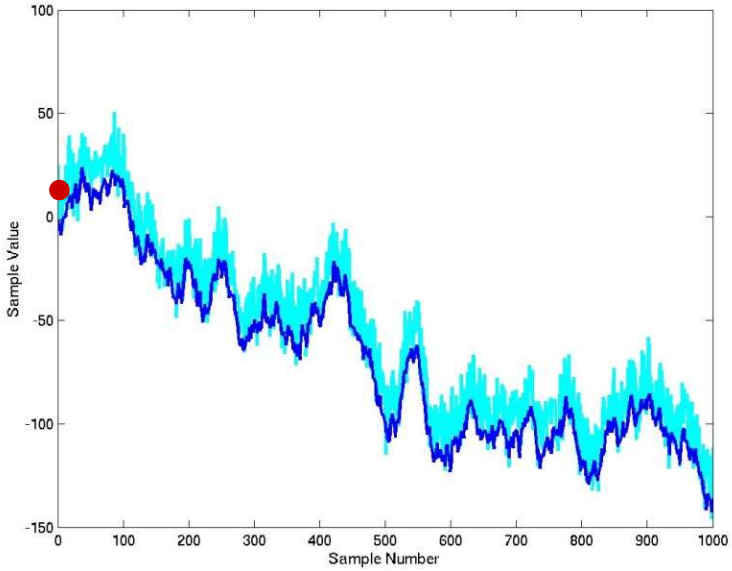


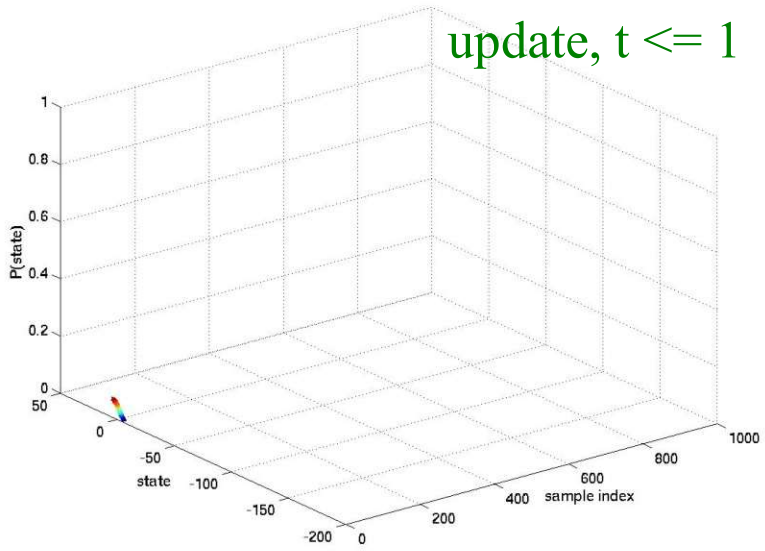


UPDATED VERSION OF
SAMPLED VERSION OF
PREDICTED STATE DISTRIBUTION
AT TIME = 1
AFTER SEEING FIRST OBSERVATION

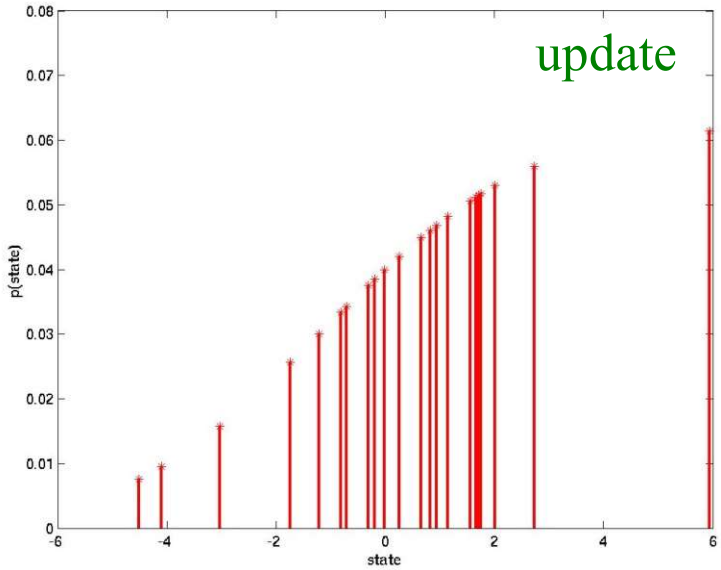
SIMULATION: TIME = 1

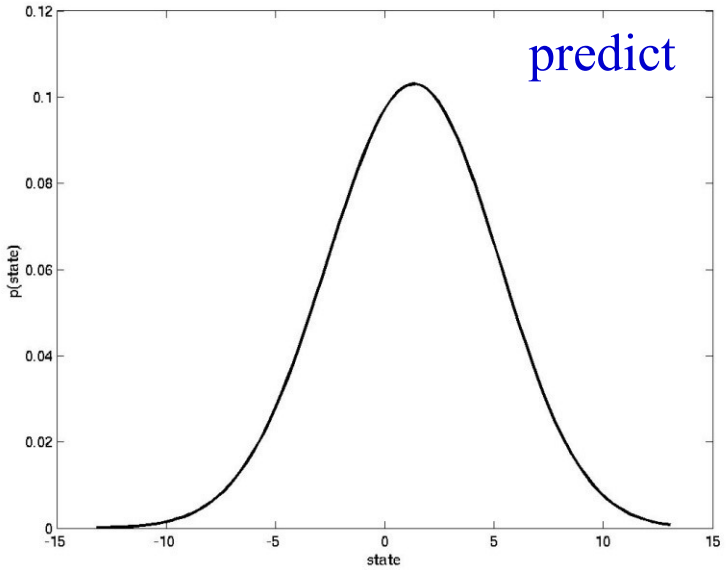


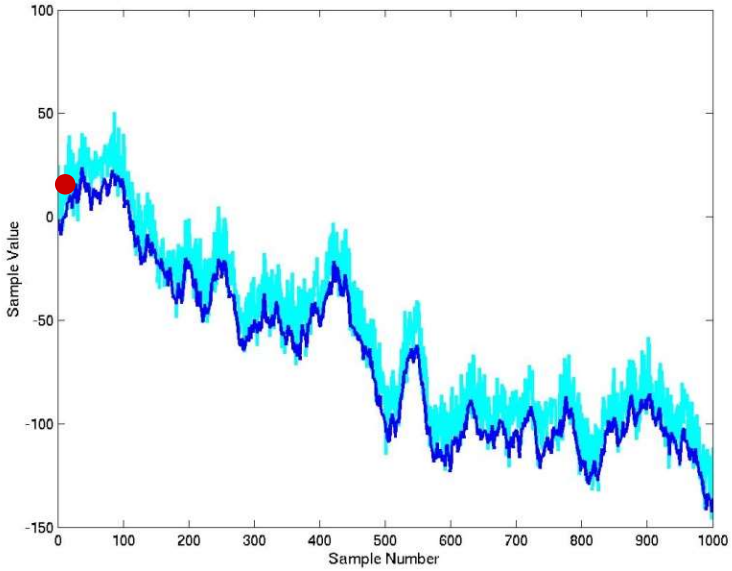


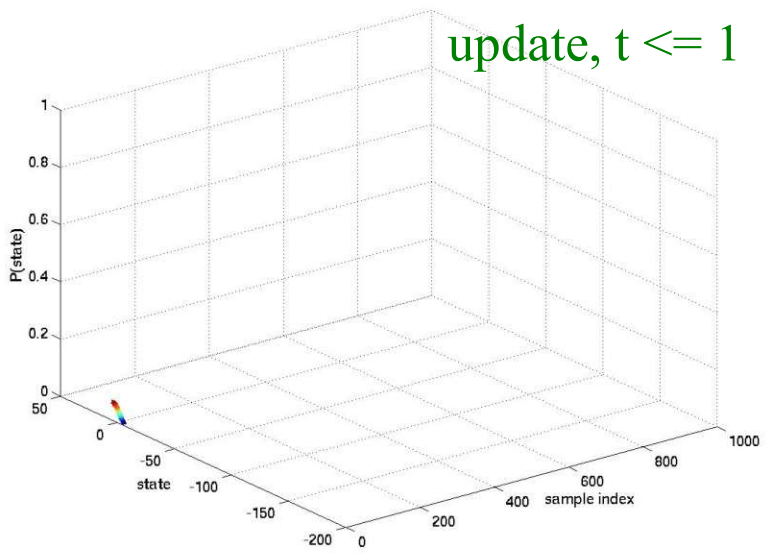


SIMULATION: TIME = 2

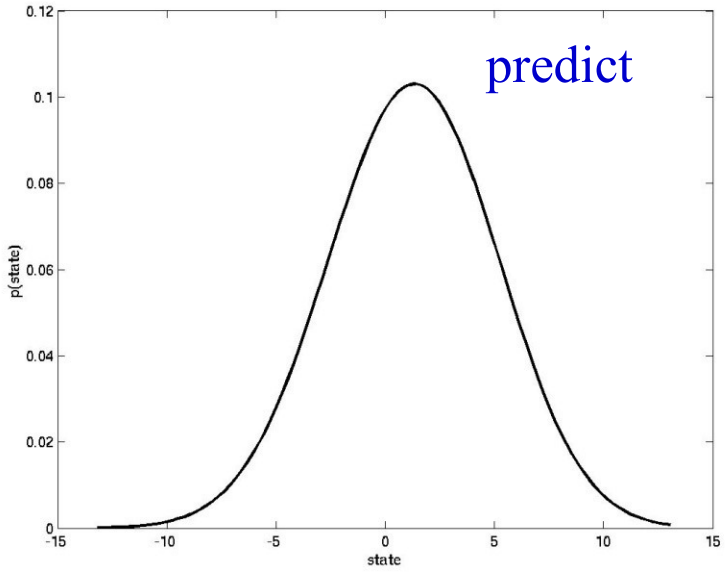


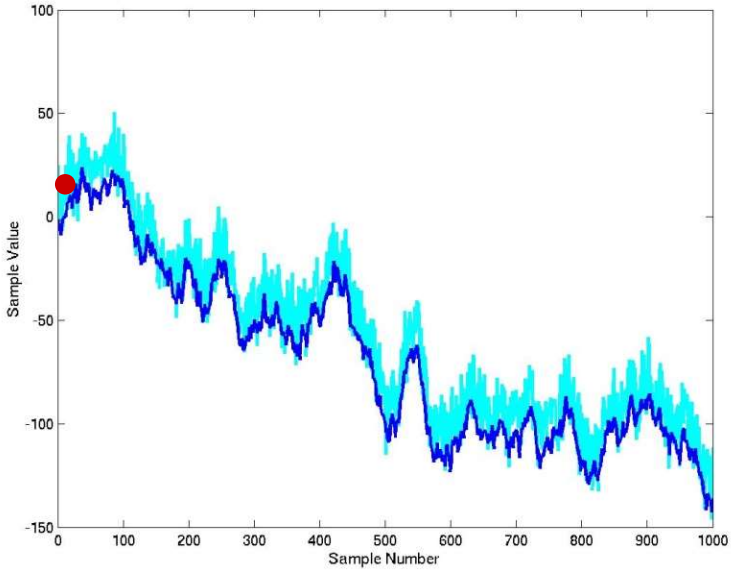


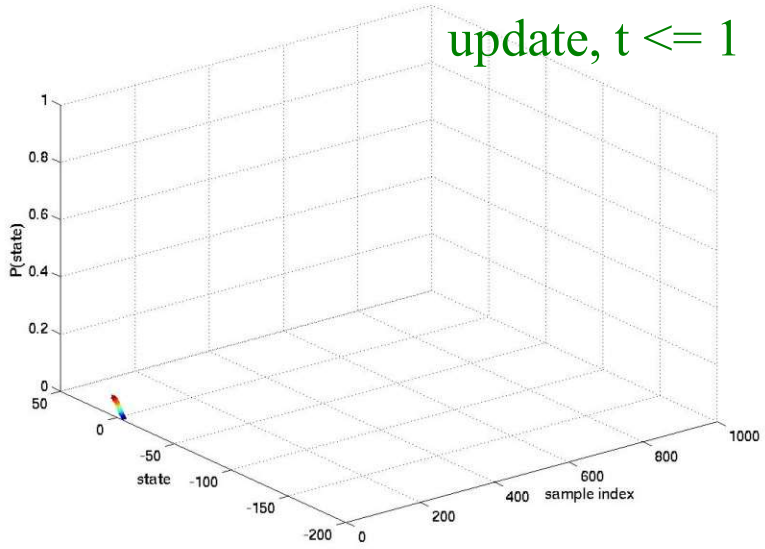




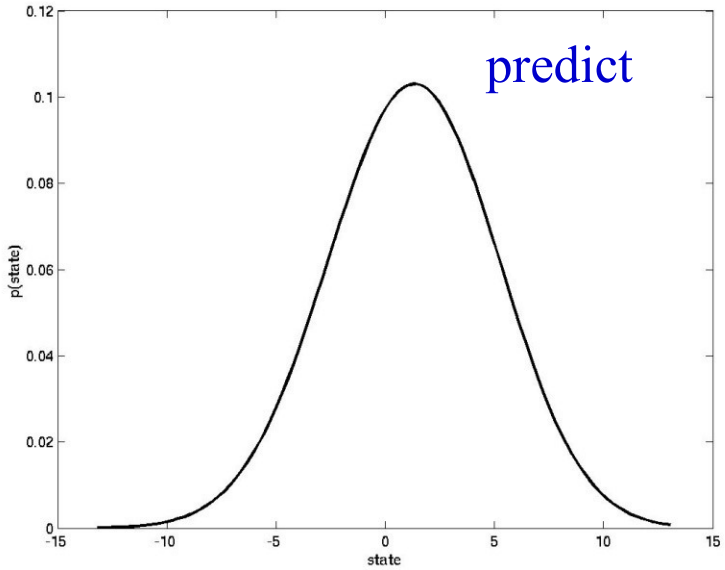
SIMULATION: TIME = 2

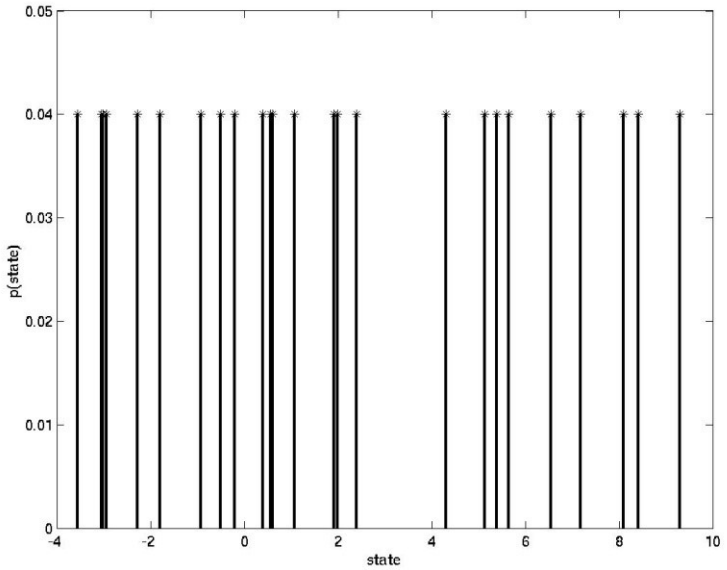


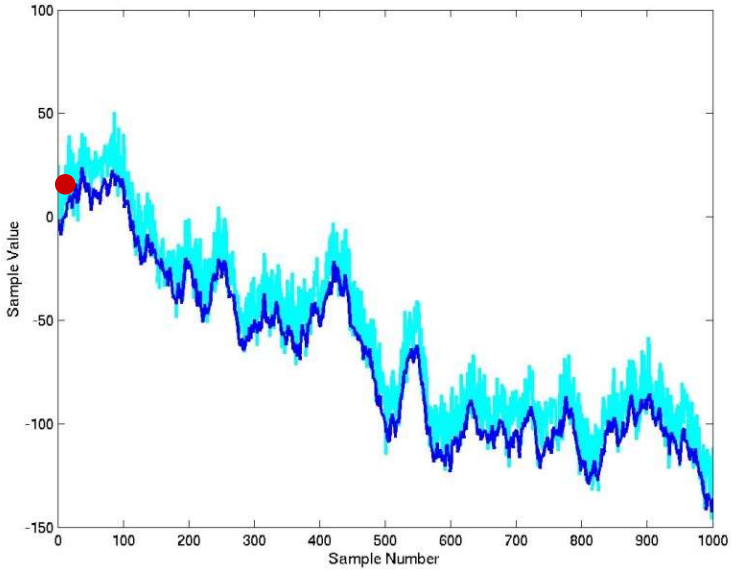


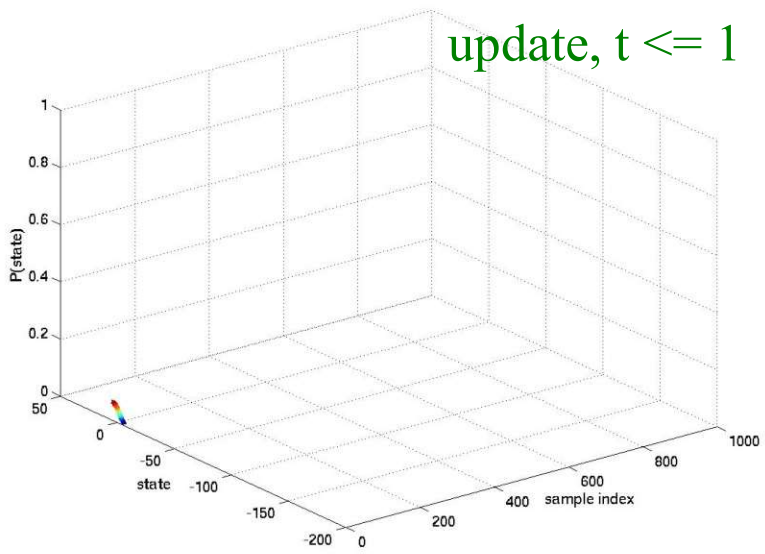


SIMULATION: TIME = 2

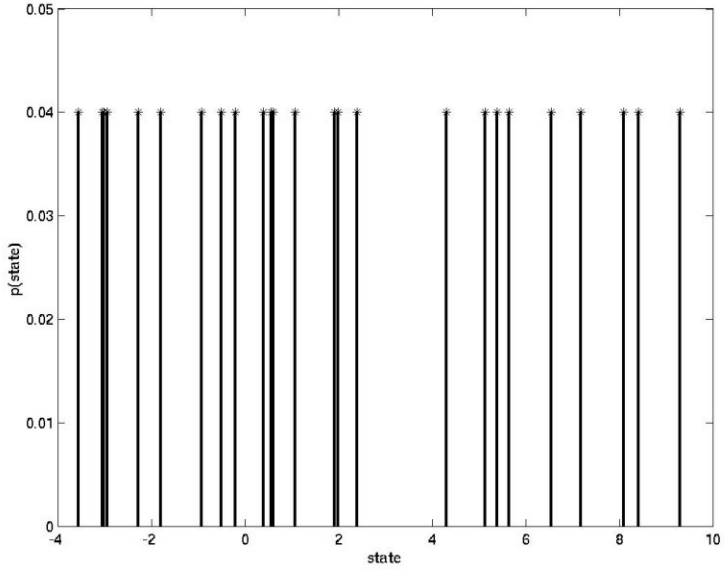


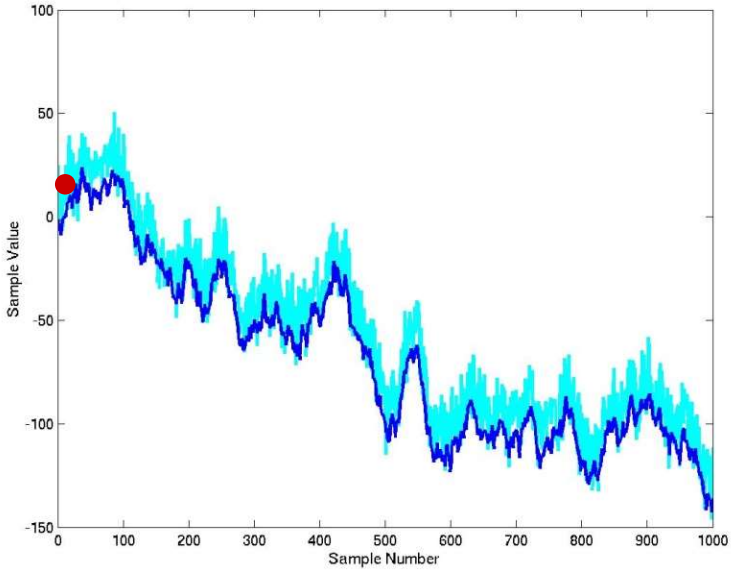




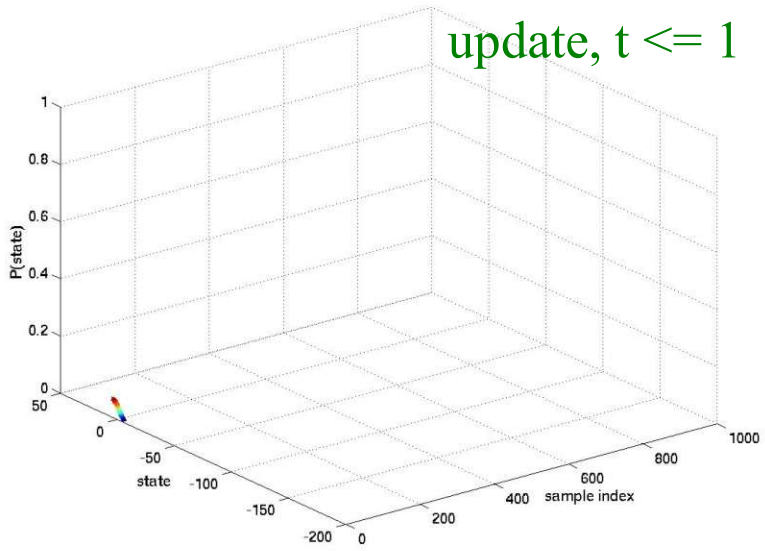


SIMULATION: TIME = 2

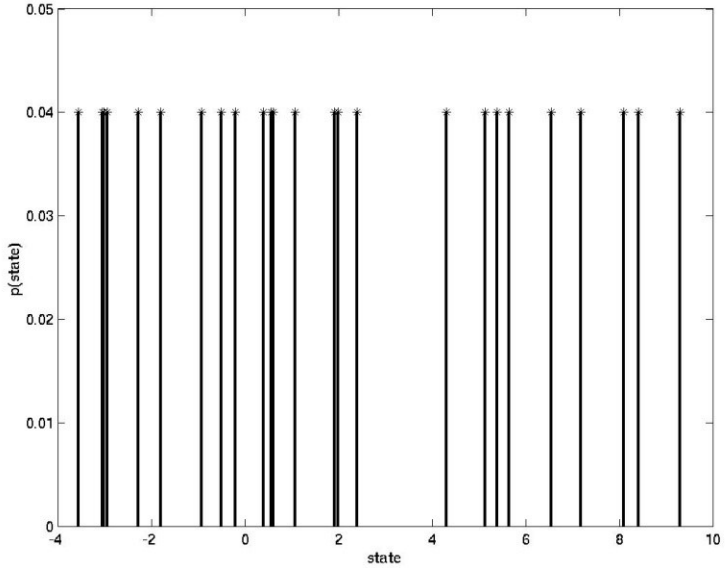


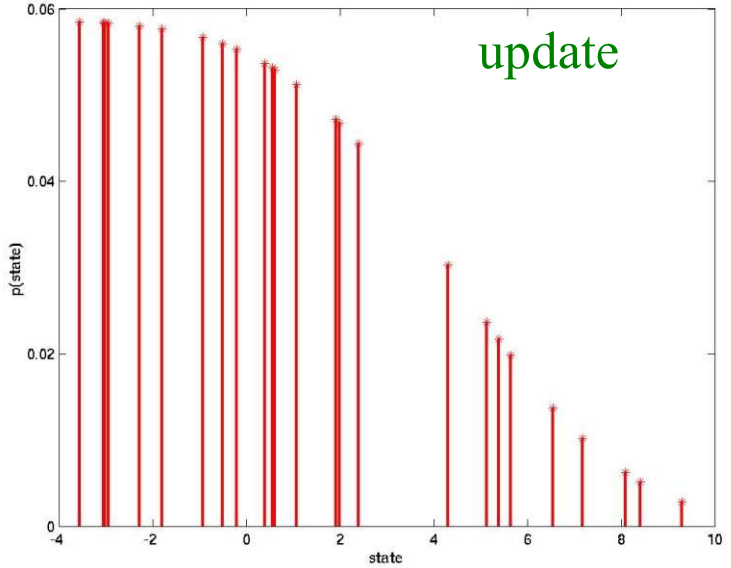


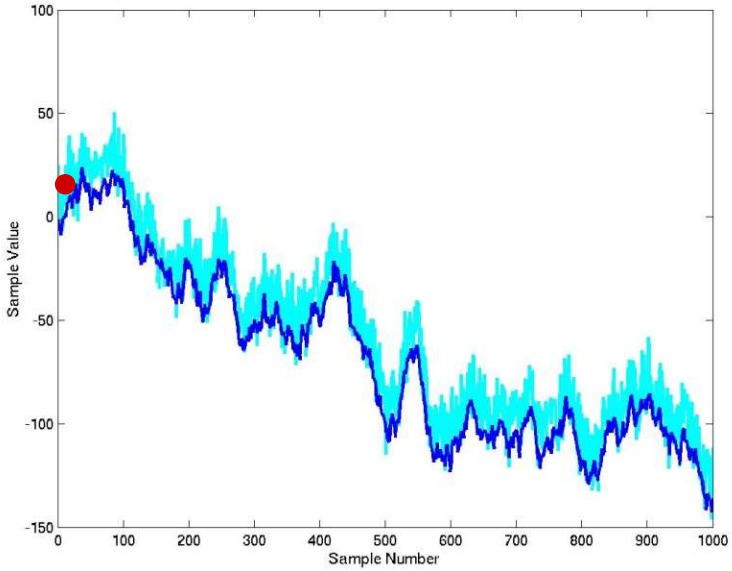
update, t <= 1

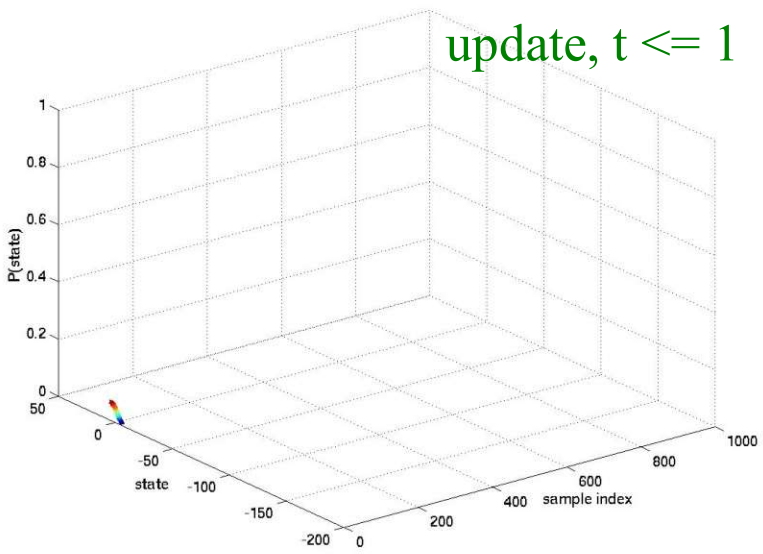


SIMULATION: TIME = 2

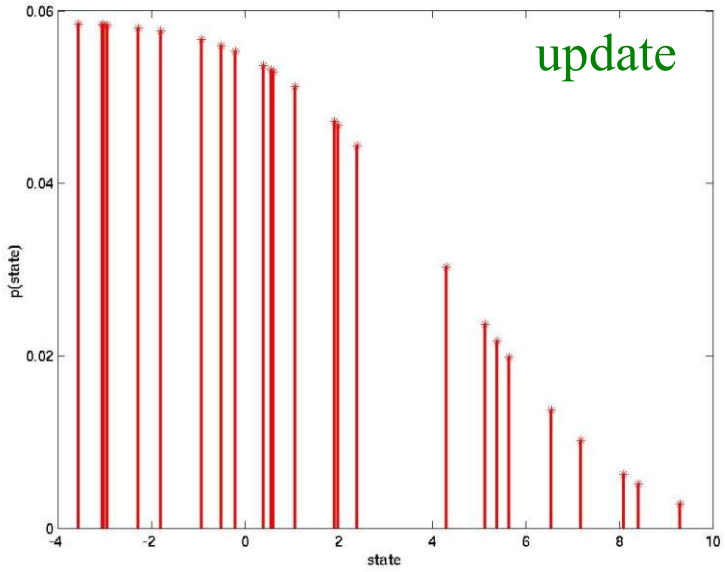


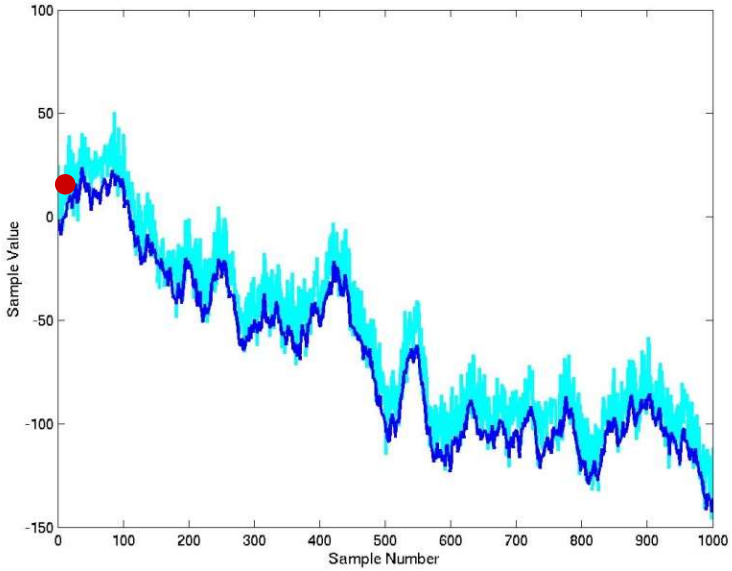


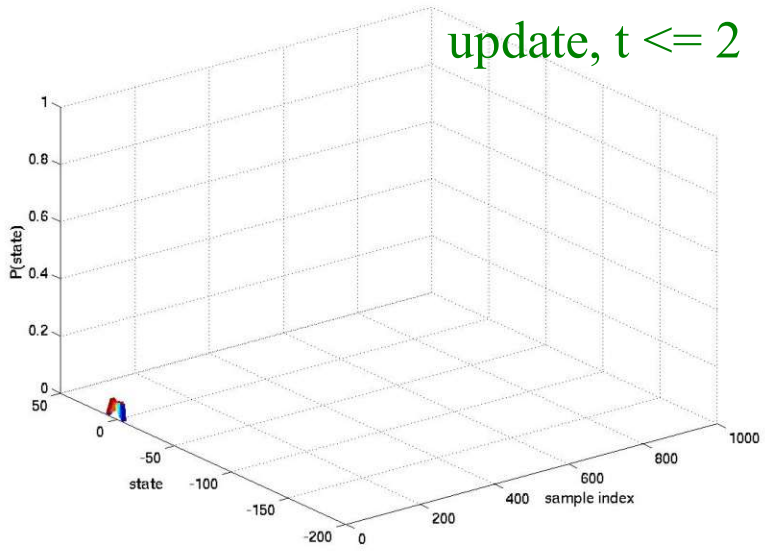




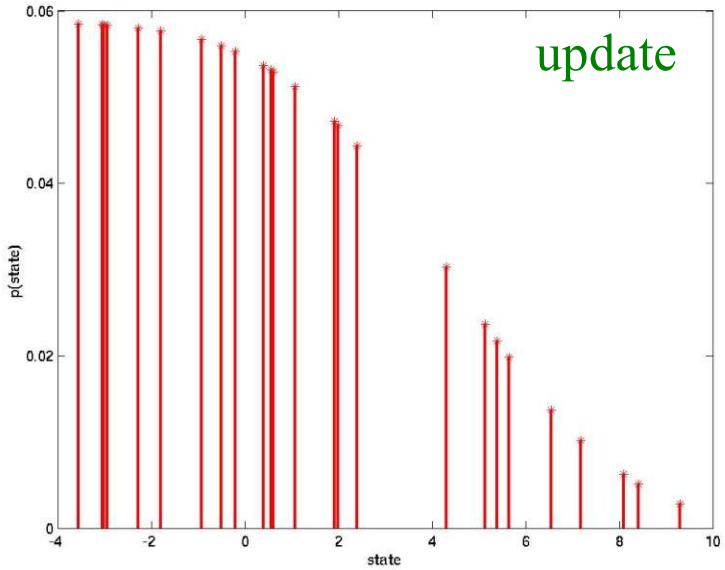
SIMULATION: TIME = 2

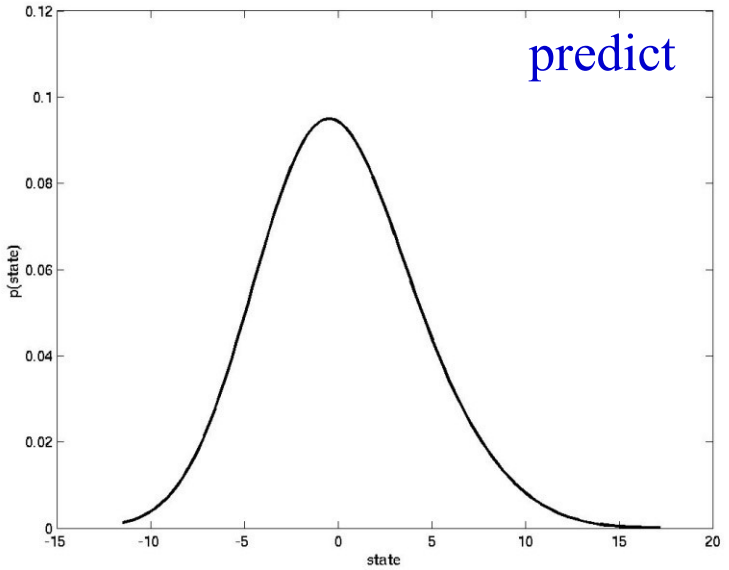


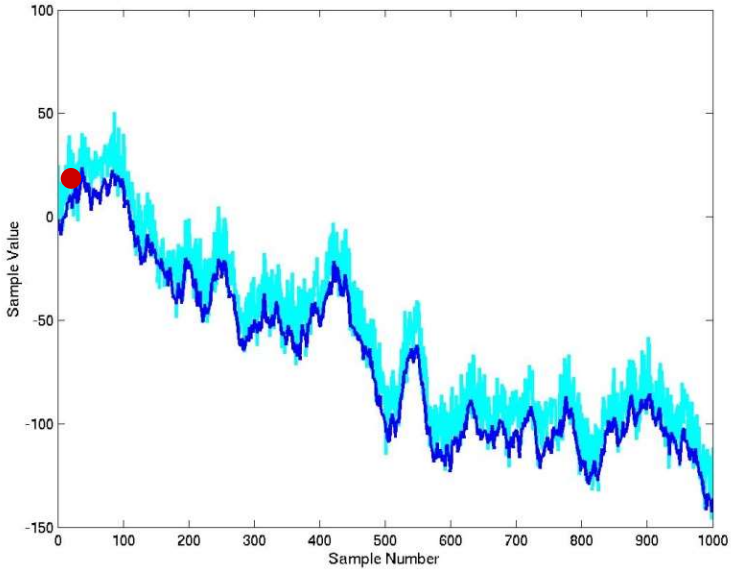


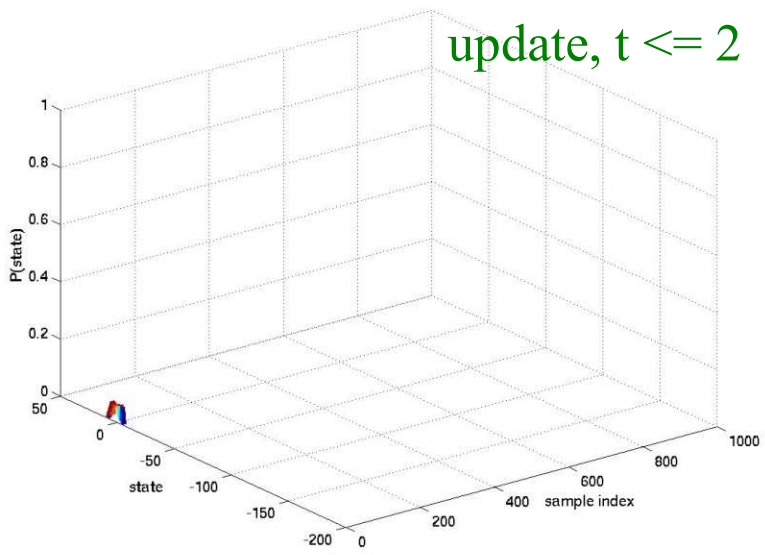


SIMULATION: TIME = 3

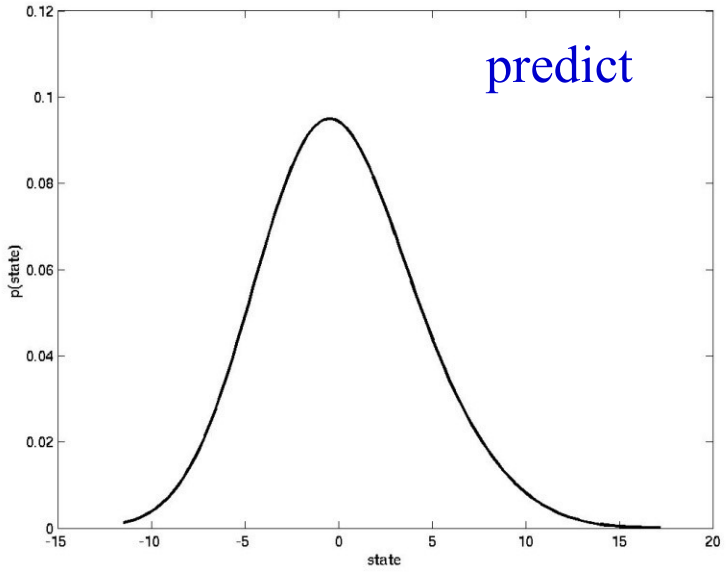


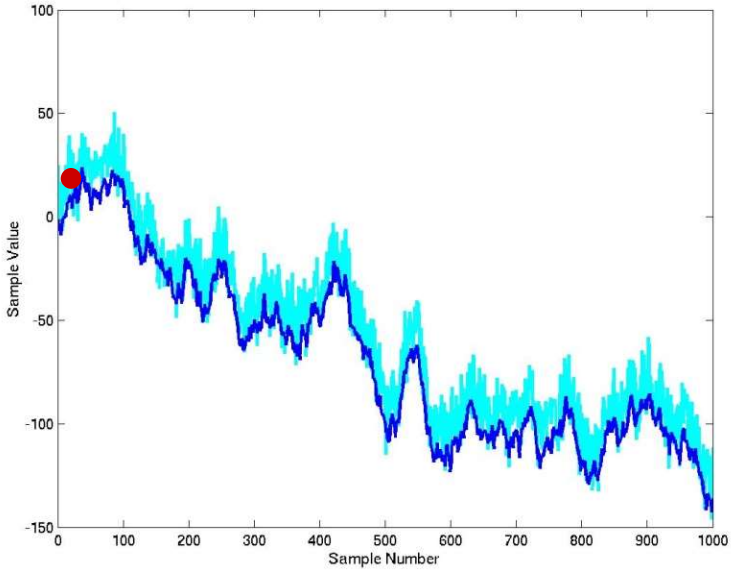


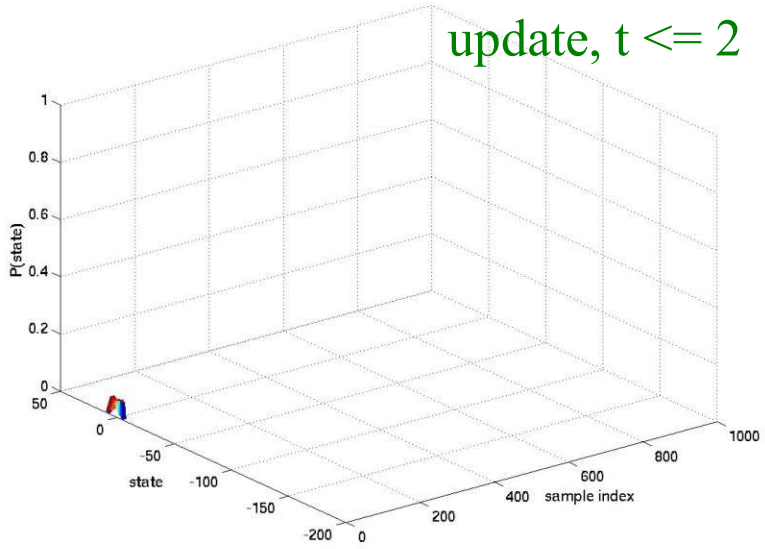




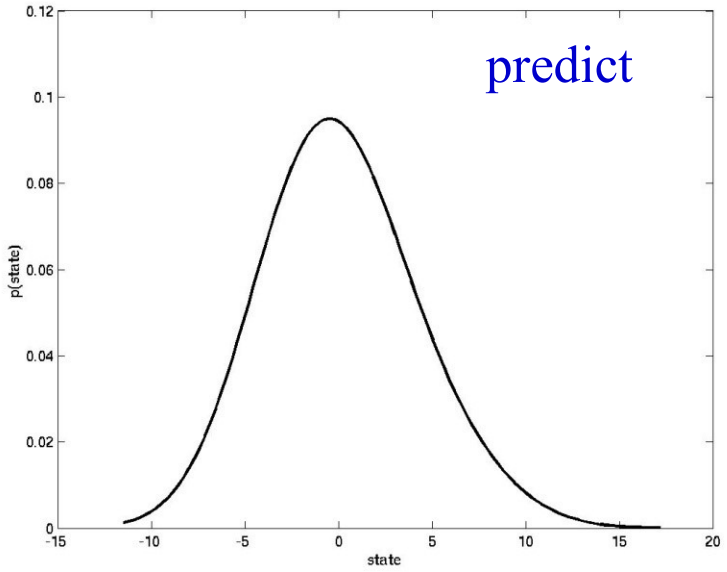
SIMULATION: TIME = 3

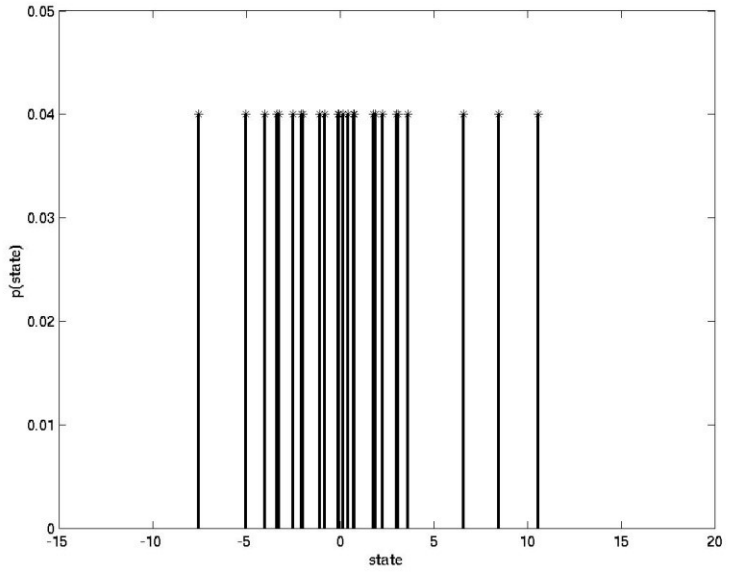


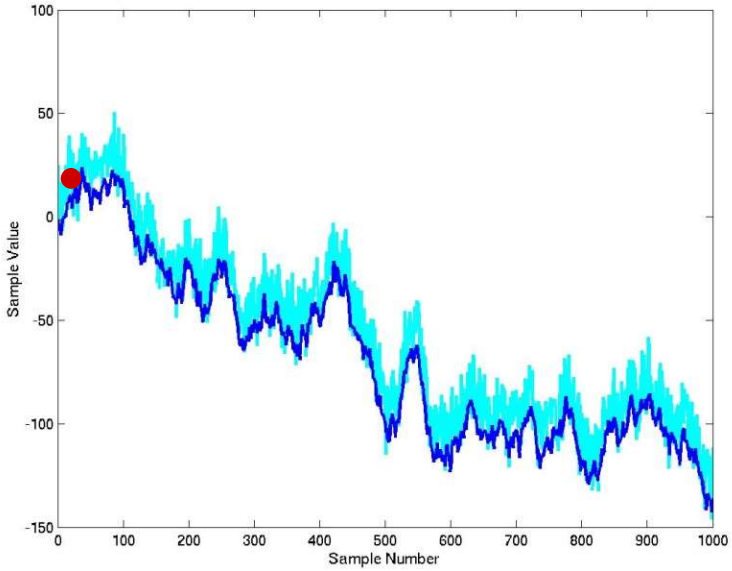


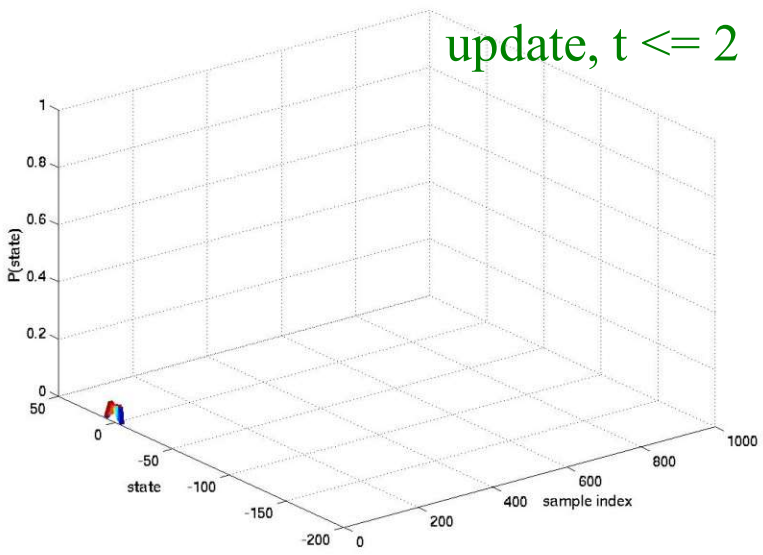


SIMULATION: TIME = 3

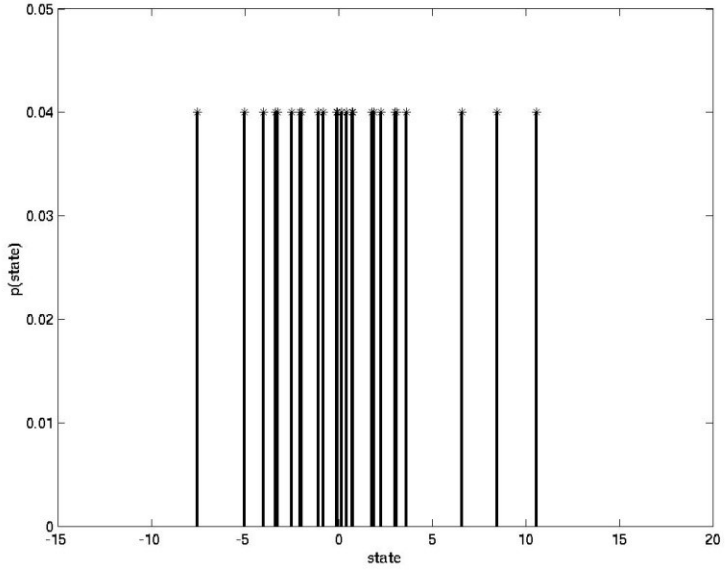


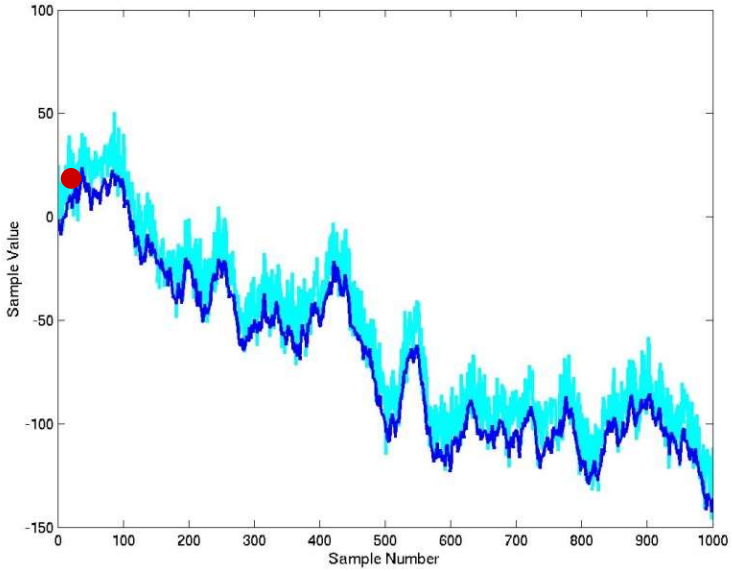




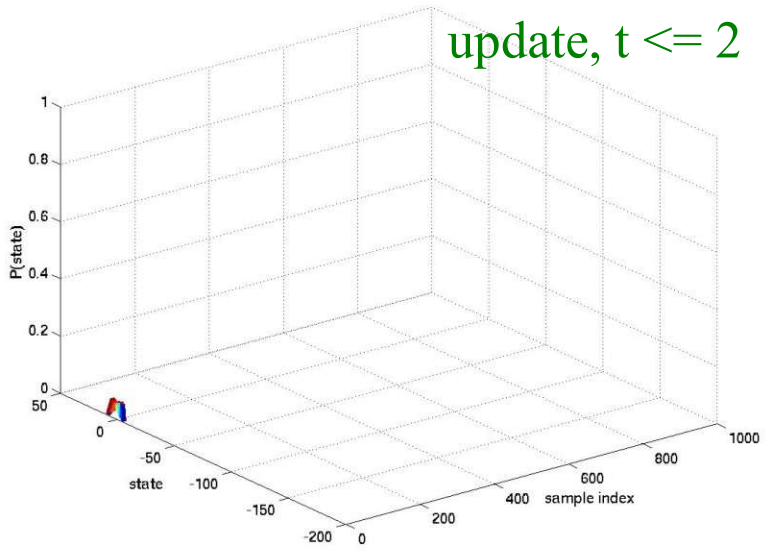


SIMULATION: TIME = 3

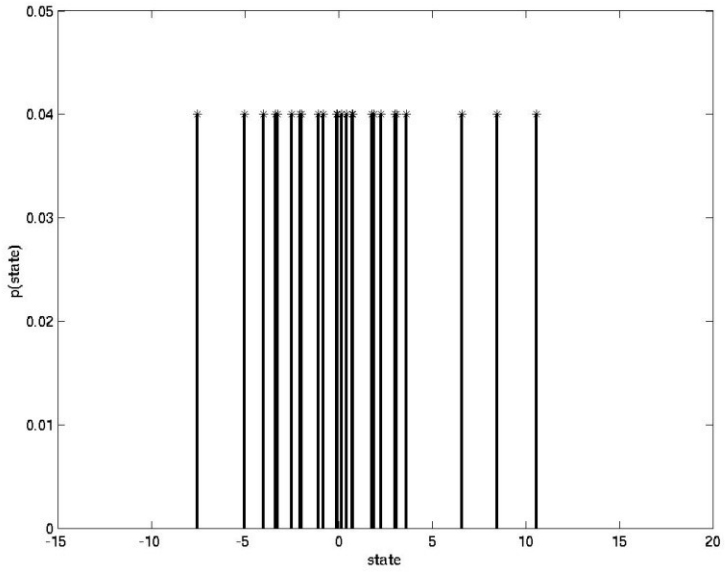


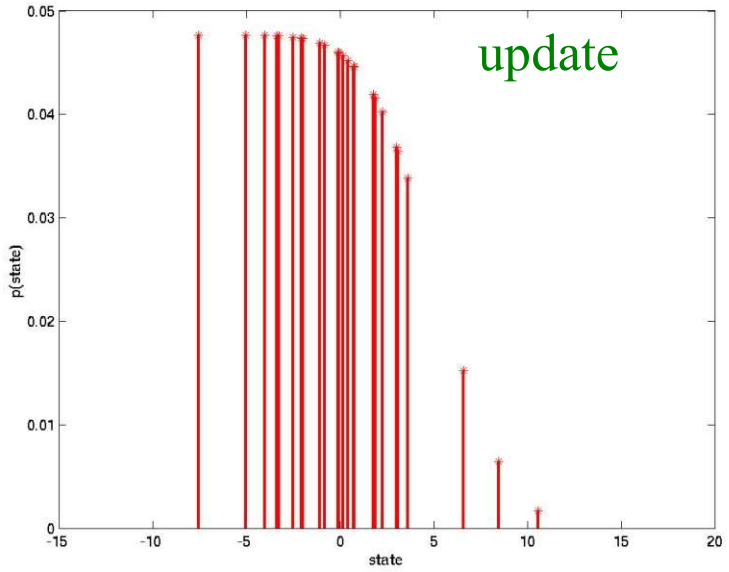


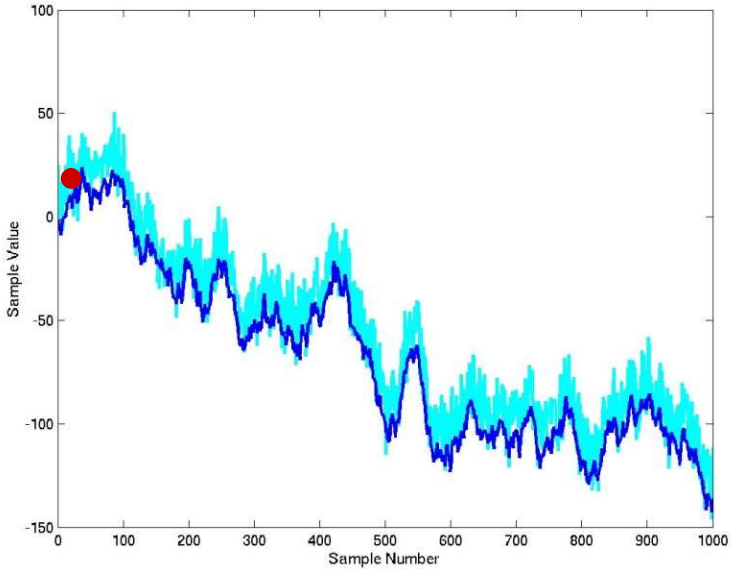
update, t <= 2

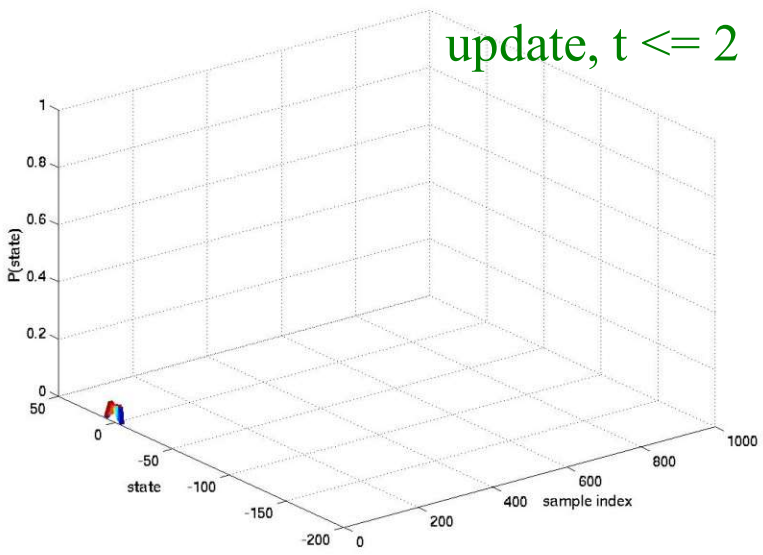


SIMULATION: TIME = 3

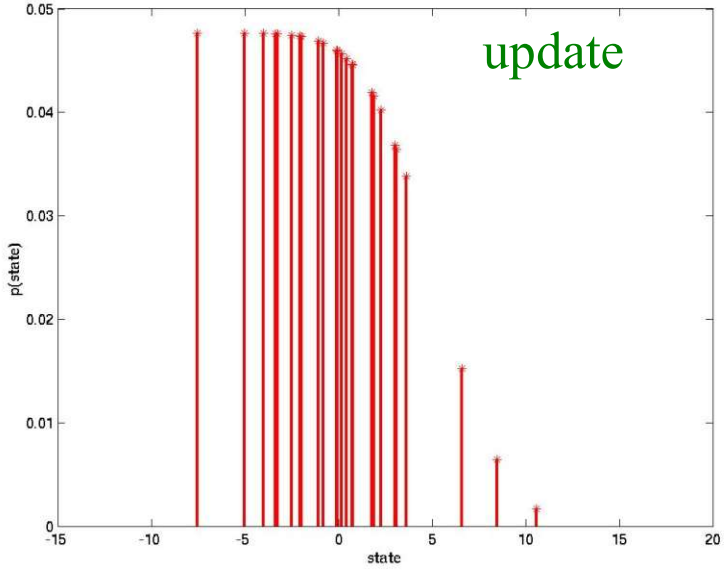


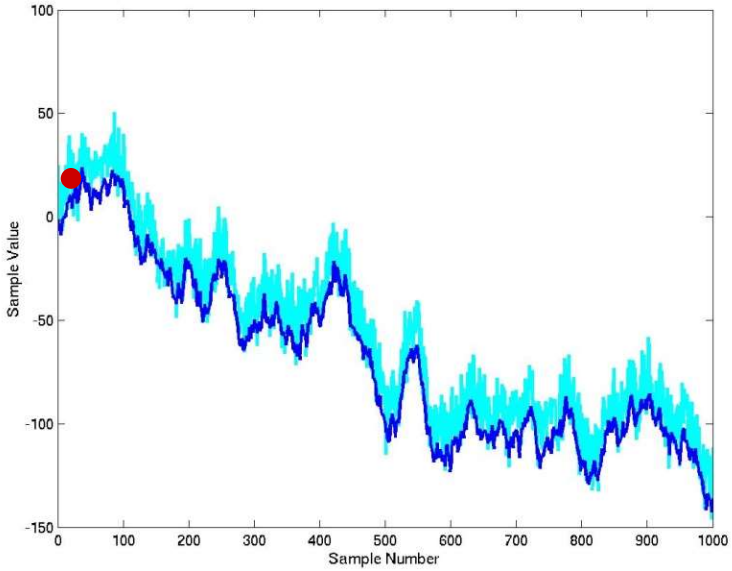




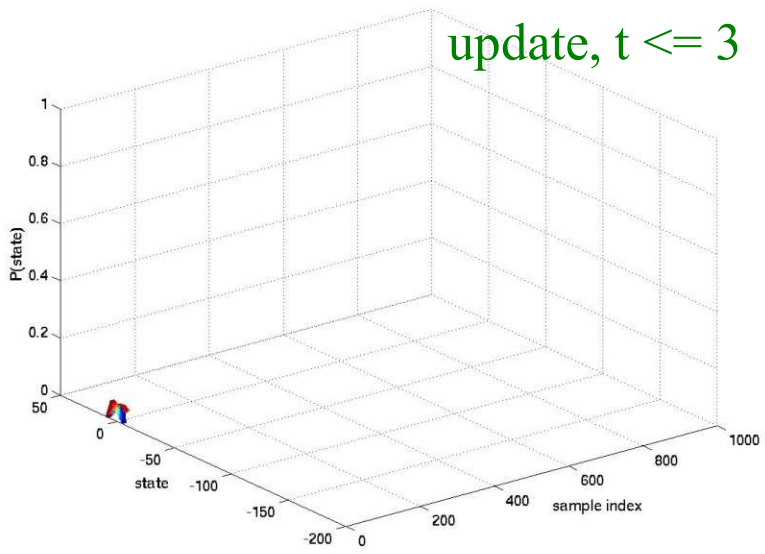


SIMULATION: TIME = 3

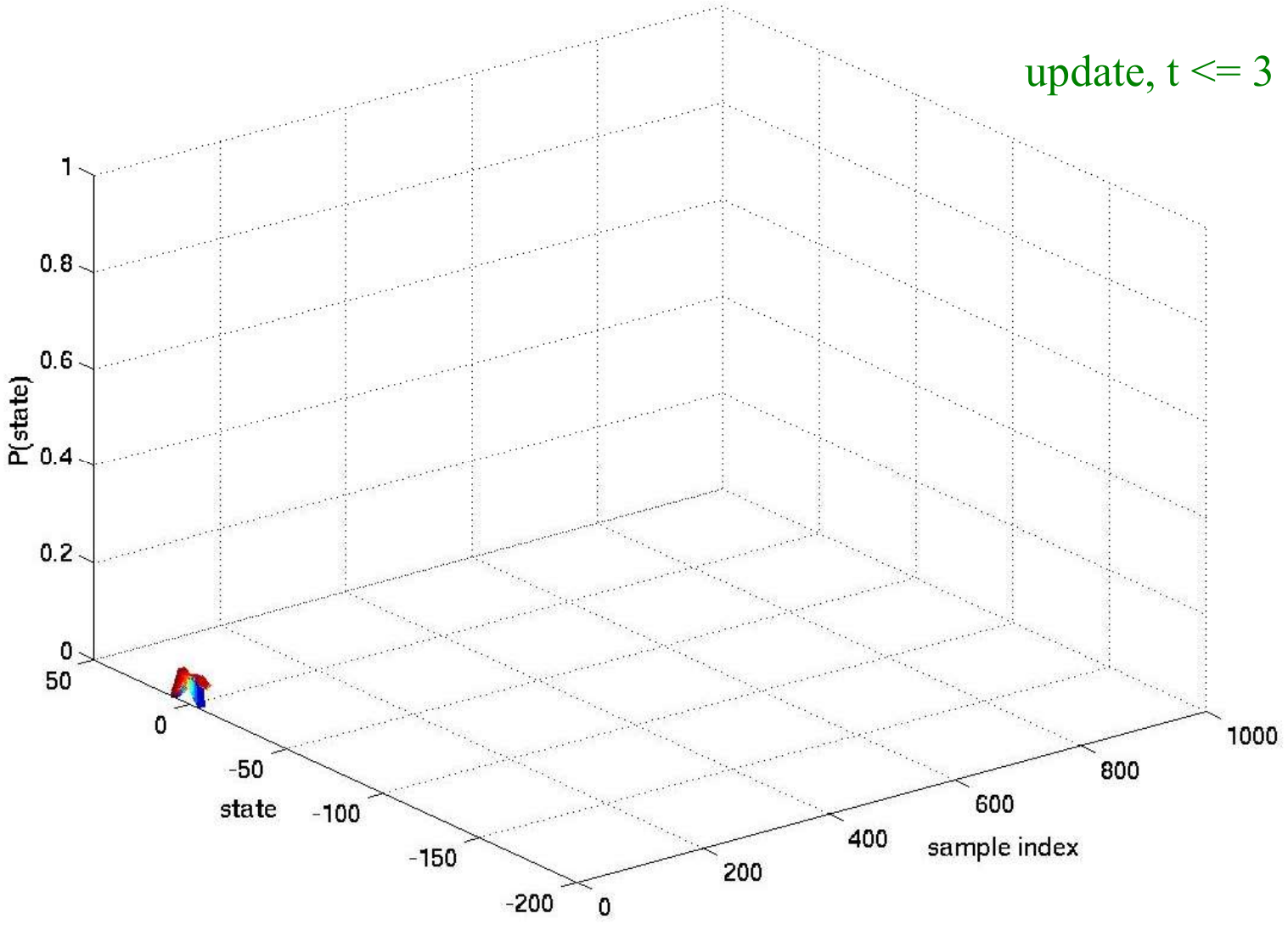




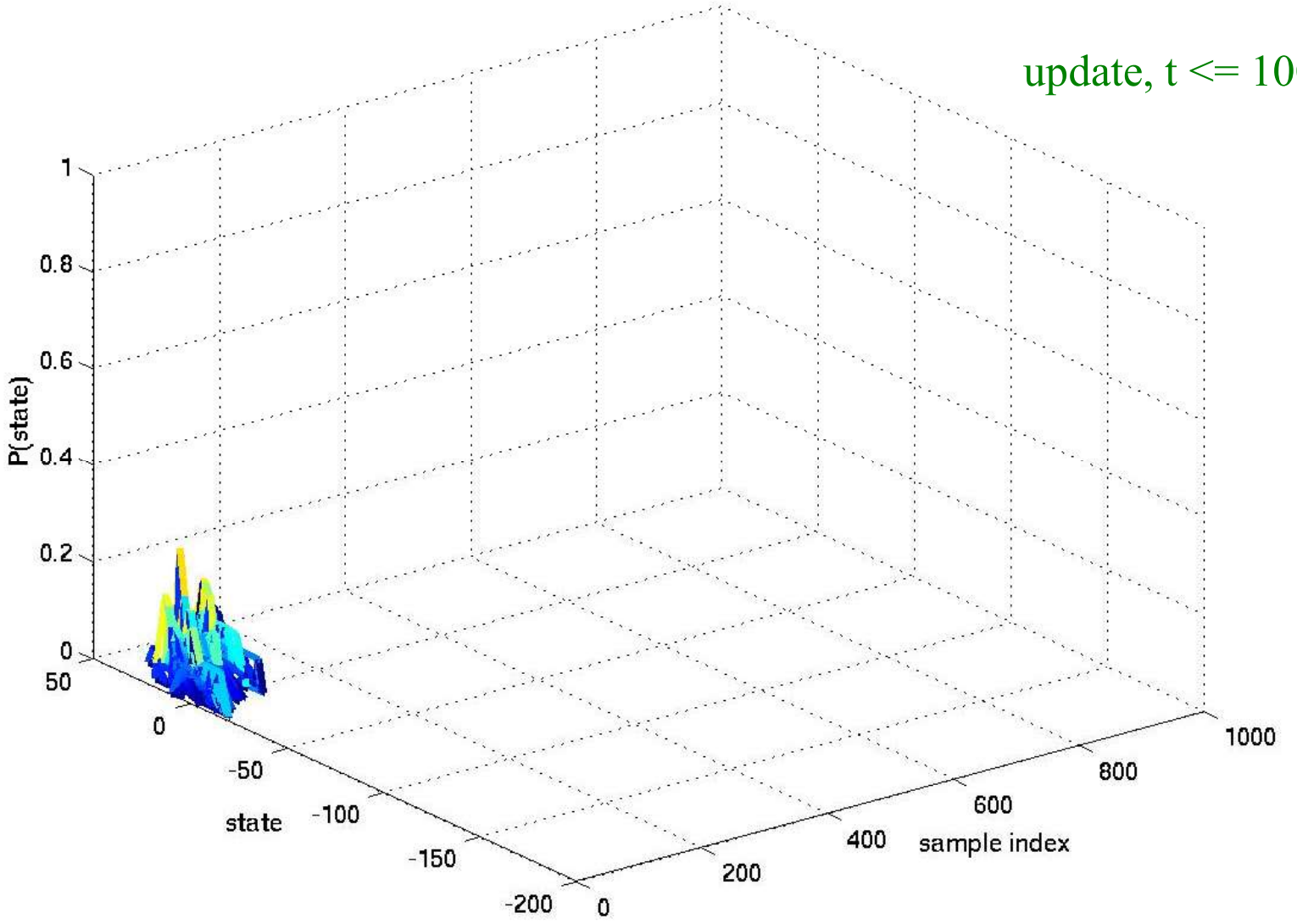
The figure below shows the contour of the updated state probabilities for all time instants until the current instant



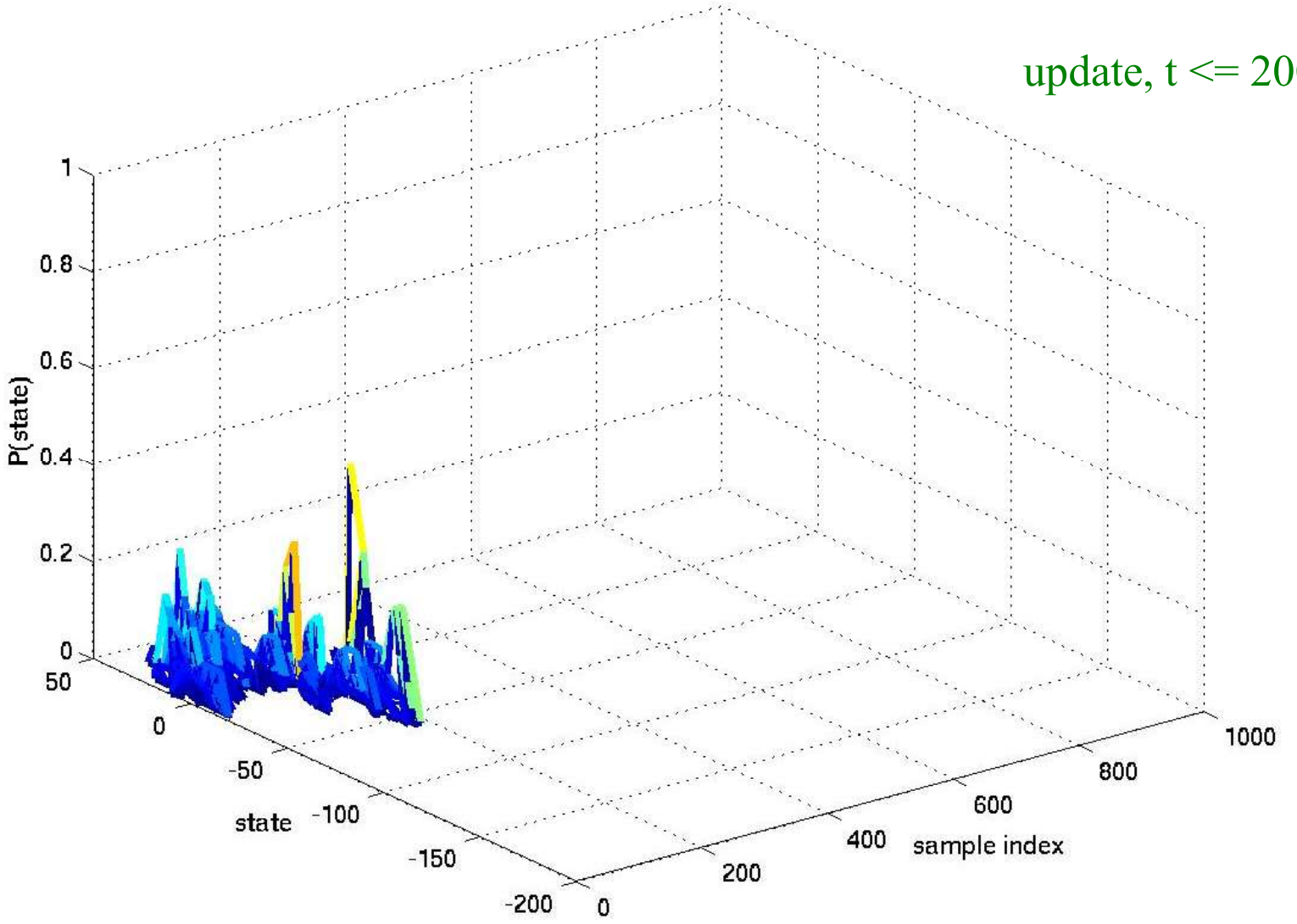
Simulation: Updated Probs Until



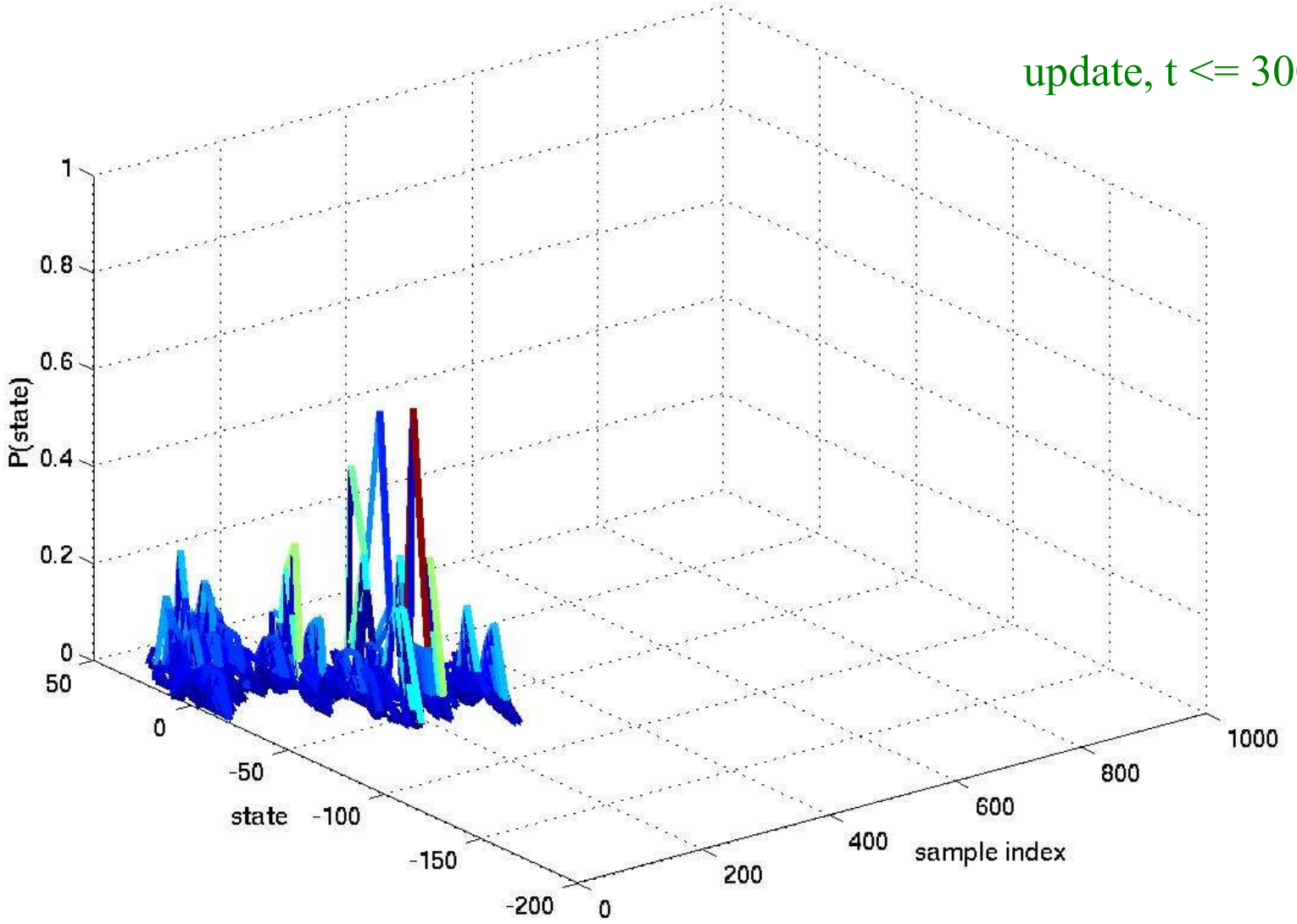
Simulation: Updated Probs Until T=100



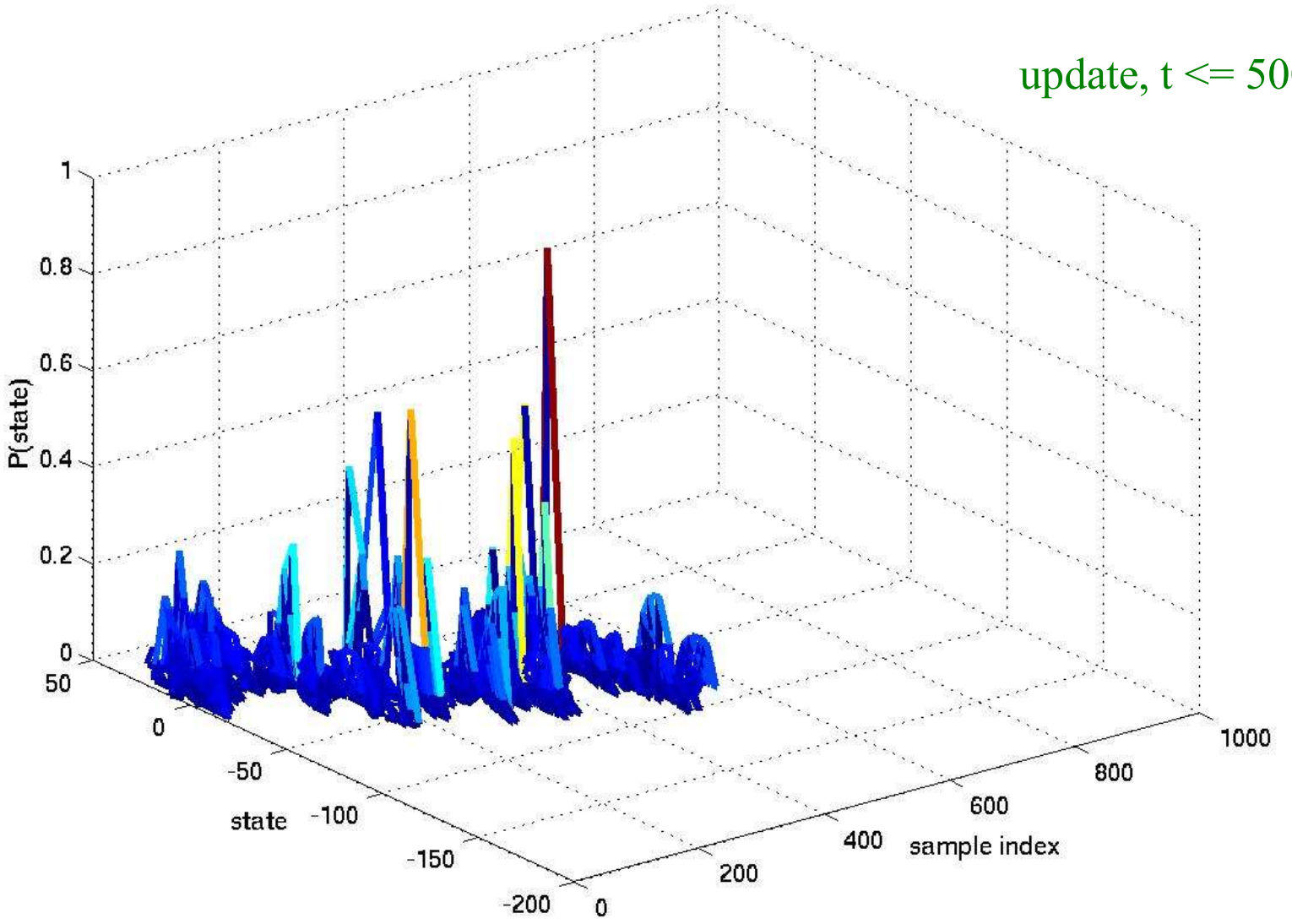
Simulation: Updated Probs Until T=200



Simulation: Updated Probs Until T=300

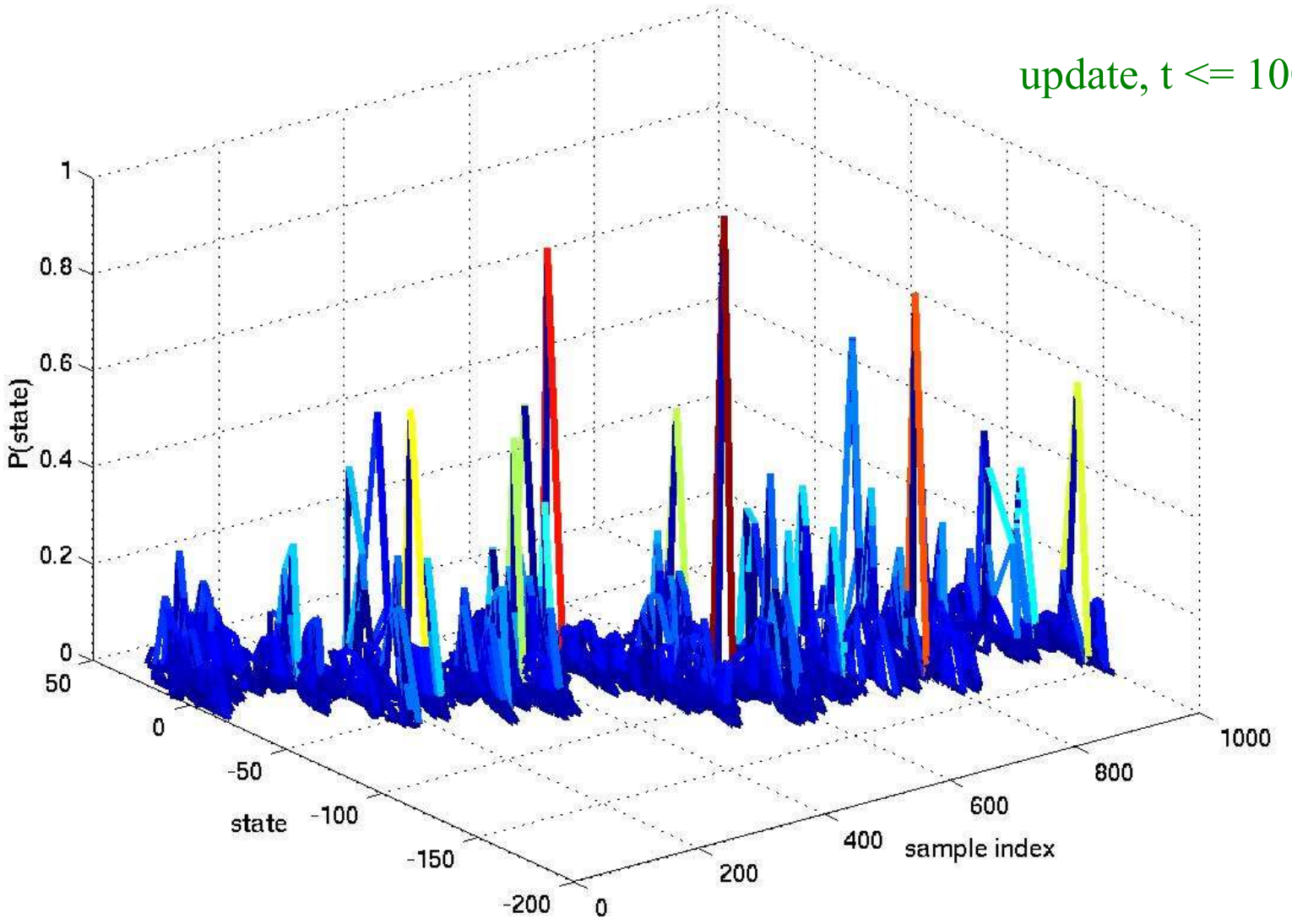


Simulation: Updated Probs Until T=500

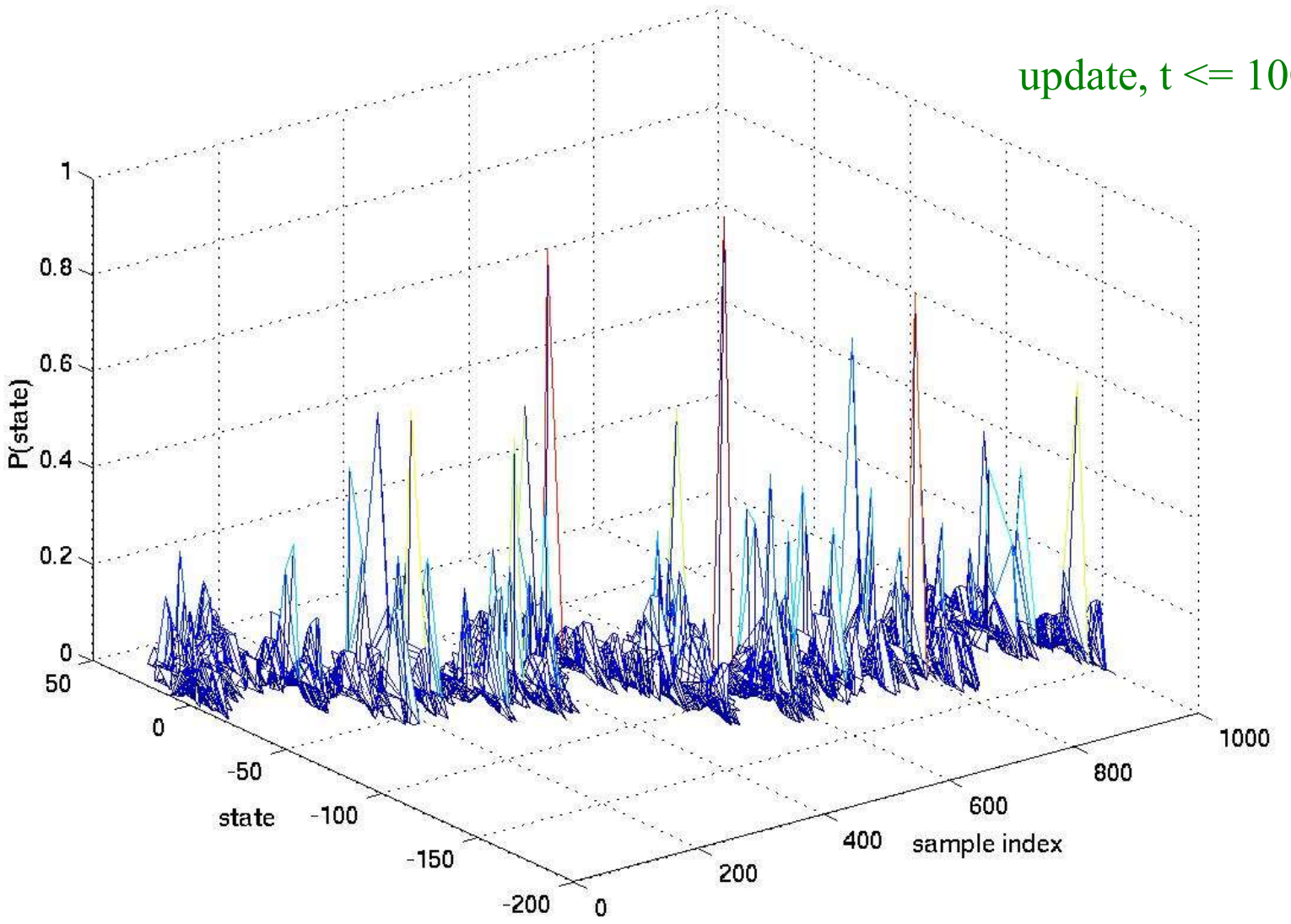


Simulation: Updated Probs Until T=1000

update, t <= 1000

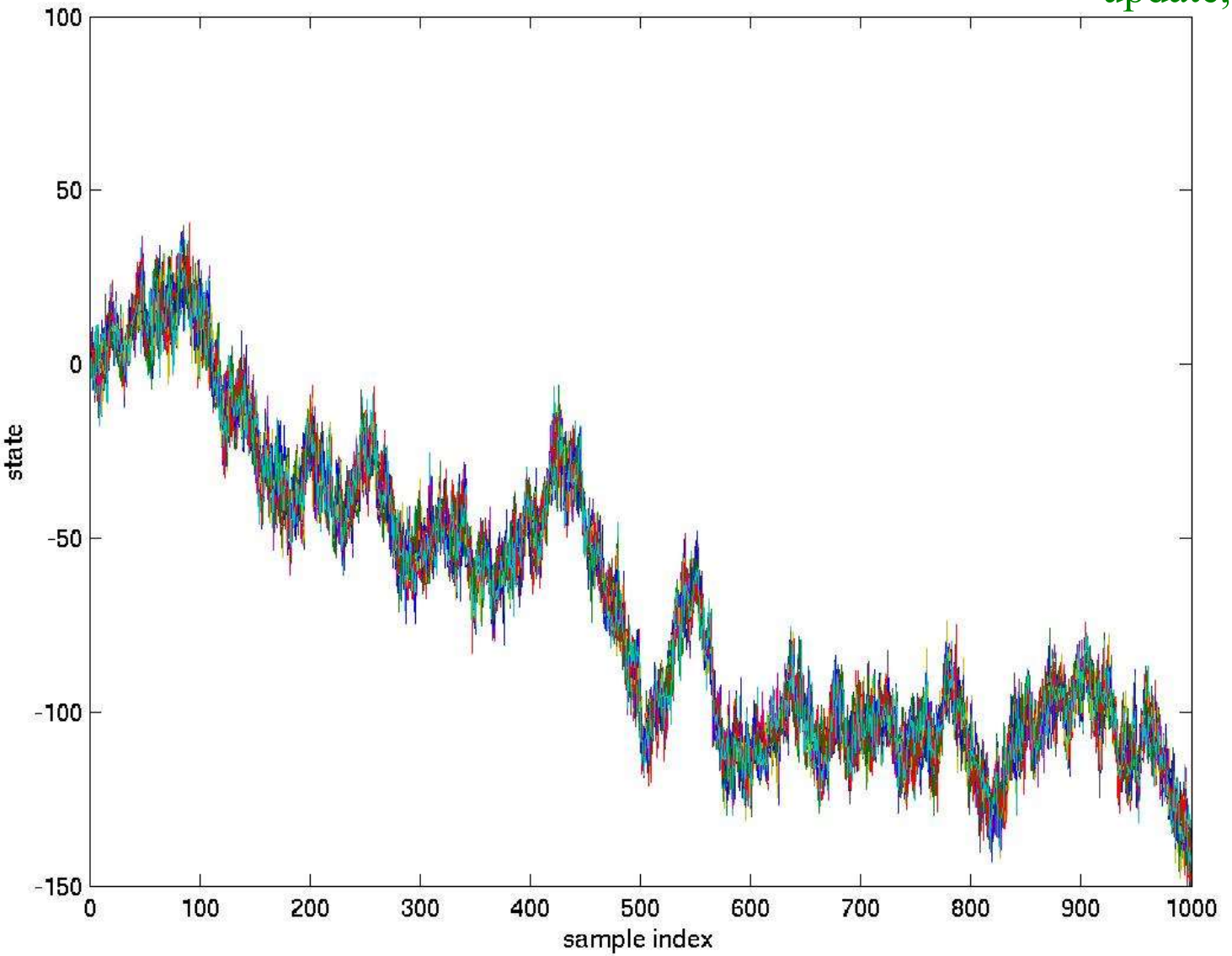


Updated Probs Until T = 1000



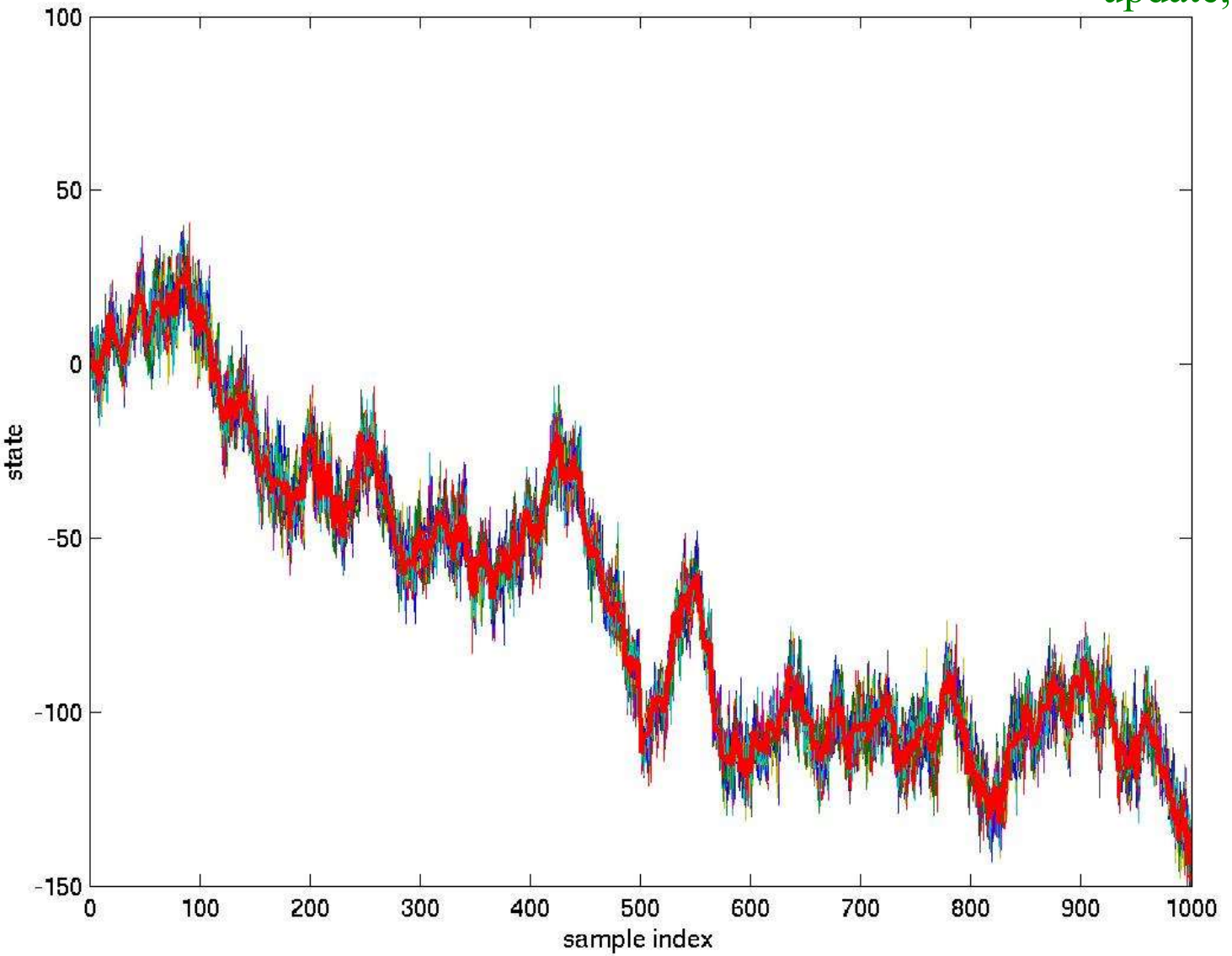
Updated Probs Until T = 1000

update, t <= 1000

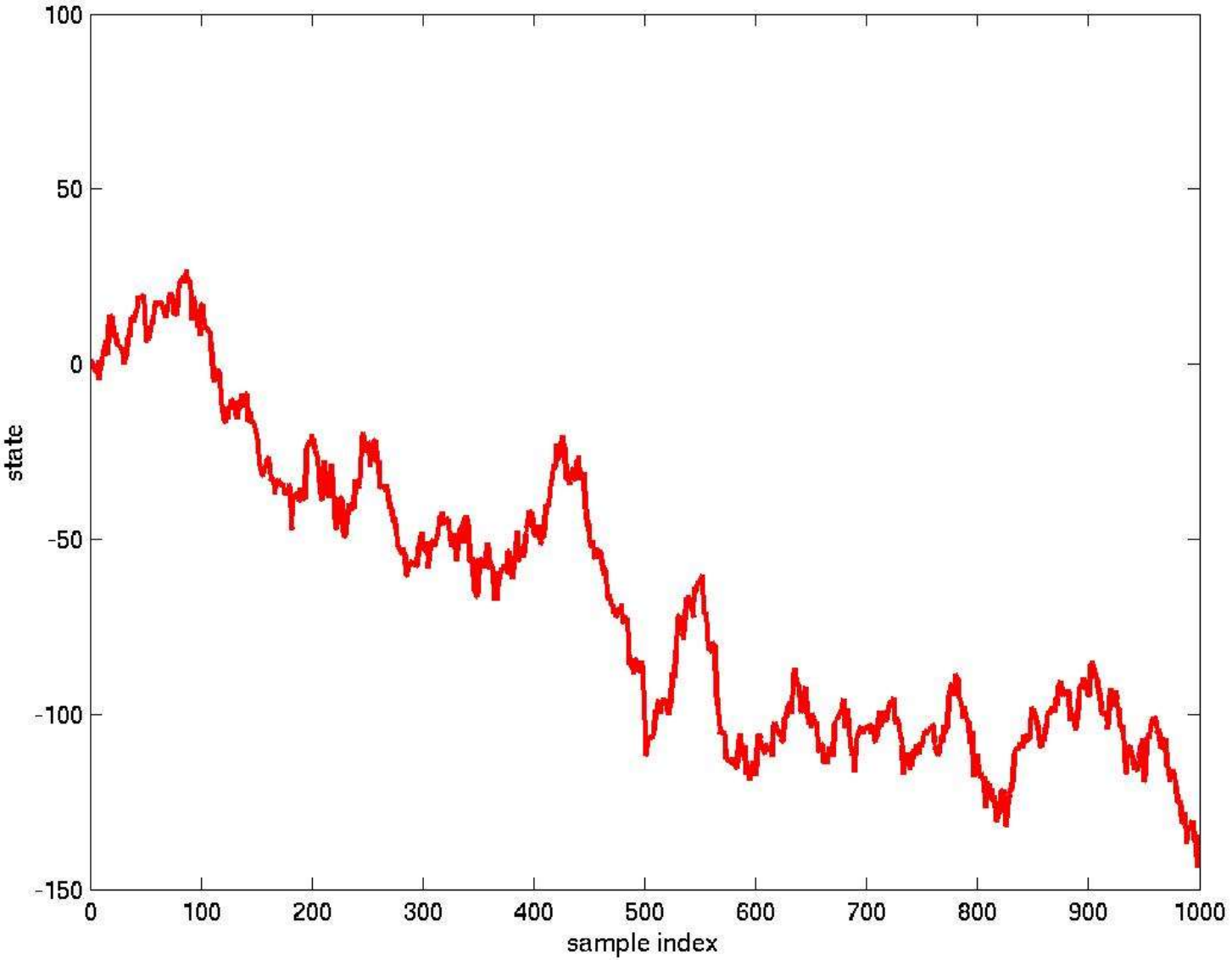


Updated Probs: Top View

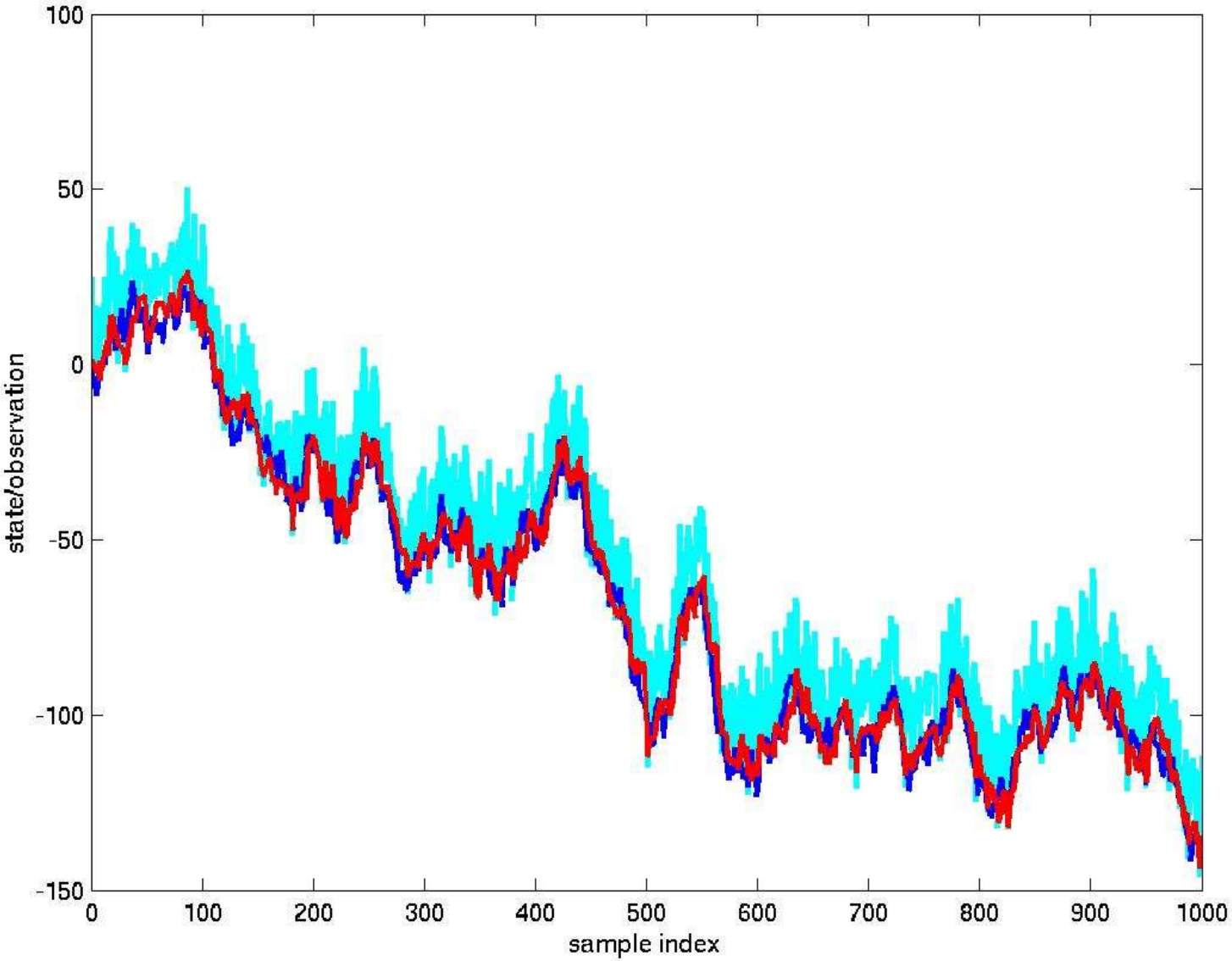
update, t <= 1000



ESTIMATED STATE



Observation, True States, Estimate



Particle Filtering

- Generally quite effective in scenarios where EKF/UKF may not be applicable
  - Potential applications include tracking and edge detection in images!
  - Not very commonly used however
- Highly dependent on sampling
  - A large number of samples required for accurate representation
  - Samples may not represent mode of distribution
  - Some distributions are not amenable to sampling
    - Use importance sampling instead: Sample a Gaussian and assign non-uniform weights to samples

Prediction filters

- HMMs
- Continuous state systems
  - Linear Gaussian: Kalman
  - Nonlinear Gaussian: Extended Kalman
  - Non-Gaussian: Particle filtering

- EKFs are the most commonly used kalman filters

- Accurate predictions with non-Gaussian models need particle-filters or other sampling based methods

The Abrupt Stop