# Kalman Filter
# Particle Filter

Lab3

# Kalman Filter

When will we use it?

1) When we have some model of the world.
2) When we can gather some observations which will tell us something about the real world.

Example:

1) A car is driving at a constant speed: $x = x0+v * t$
2) Measurement of inaccurate GPS signal

# Kalman Filter and Particle Filter

When will we use it?

1) We want to treat a problem where information is presented over time, not all at once.

State (x) - What we want

Measurement (z) - What we have

Kalman - assumes linear relation between the two and Gaussian all around

Particle - Relaxes the Gaussian assumption (but still assumes linear transition matrix between states)

# Kalman Filter

When will we use it?

1) When we have some model of the world.
2) When we can gather some observations which will tell us something about the real world.

Example:

1) A car is driving at a constant speed: x = x0+v * t
2) Measurement of inaccurate GPS signal

**Noise in the model is modeled by a matrix Q**

$$x_{k+1} = \Phi x_k + w_k$$

$$Q = E\left[w_k w_k^T\right]$$

# Kalman Filter

When will we use it?

1) When we have some model of the world.
2) When we can gather some observations which will tell us something about the real world.

Example:

1) A car is driving at a constant speed: x = x0+v * t
2) Measurement of inaccurate GPS signal

**Measurement noise is modeled by matrix R**

$$R = E\left[v_k v_k^T\right] \leftarrow z_k = Hx_k + v_k$$

# Kalman Filter

When will we use it?

1) When we have some model of the world.
2) When we can gather some observations which will tell us something about the real world.

Example:

1) A car is driving at a constant speed: $x = x0+v * t$
2) Measurement of inaccurate GPS signal

What do we obtain? An estimate of the state.

# Kalman Filter - What’s a state anyway?

Goal: We're interested in the state: $x_k$

Kalman Filter outputs: $\hat{x}_k$

# Kalman formulation

## The state:

We are interested in estimating some $x_k$.

We know (or assume / model) that the $x_k$'s progress in time is:

$$\boxed{x_{k+1} = \Phi \cdot x_k + w_k}$$

## The measurements:

We observe: $z_k$ - These are our measurements.

The measurements are related to the state via:

$$\boxed{z_k = H \cdot x_k + v_k}$$

## The goal accroding to Kalman's formulation:

Find: $\hat{x}_k$ which minimizes:

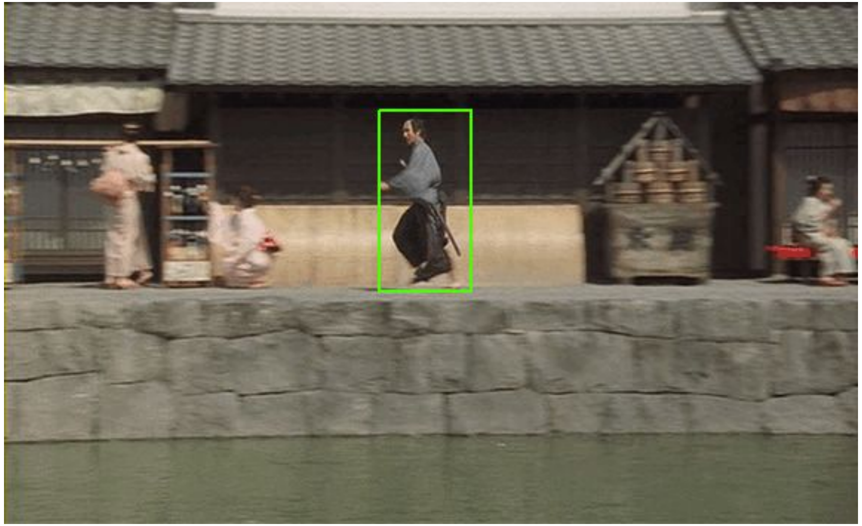
$$\boxed{f(e_k) = (x_k - \hat{x}_k)^2}$$

for every time step.

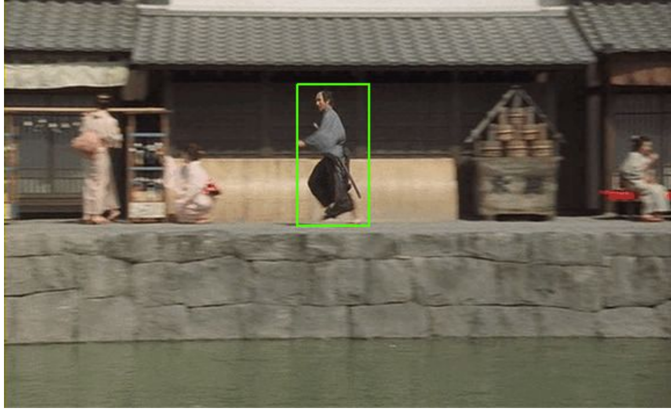
# Example

We're interested in the location of the subject in the

$$\rightarrow x_k = [x_c, y_c, w, h, v_x, v_y]$$



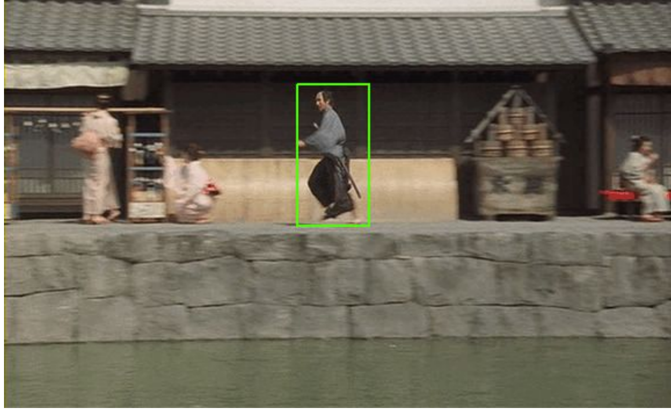
# Kalman Filter Checklist



We need to answer the following questions:

1. What is the state that we're interested in?
2. What can we measure?
3. How well do we measure it?
4. What can we assume about the physics of the problem?
5. How well do we model the physics of the problem?

# Kalman Filter Checklist

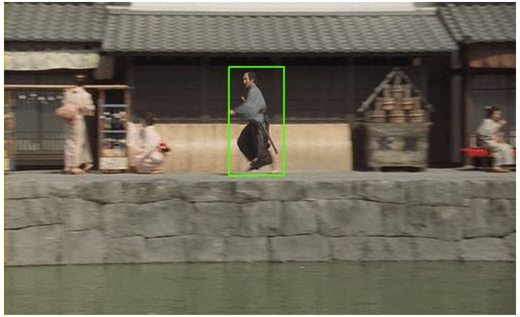


We need to answer the following questions:

1. What is the state that we're interested in? $\rightarrow\ x_k$
2. What can we measure? $\rightarrow\ z_k$
3. How well do we measure it? $\rightarrow\ v_k$ Modeled by: $R = E[v_k v_k^T]$
4. What can we assume about the physics of the problem? $\rightarrow\ \Phi$
5. How well do we model the physics of the problem? $\rightarrow\ w_k$ Modeled by: $Q = E[w_k w_k^T]$

# Kalman Filter Checklist

1. What is the state that we're interested in? $\rightarrow$ $x_k = [x_c, y_c, w, h, v_x, v_y]$
2. What can we measure? $\rightarrow$ $z_k = SSD(template, candidate)$
3. How well do we measure it? $\rightarrow$ $v_k$ Modeled by: $R = E[v_k v_k^T]$
4. What can we assume about the physics of the problem? $\rightarrow$ $\Phi$ s.t:



$$x_{k+1}[0] = x_k[0] + x_k[4] \cdot 1$$

and:

$$x_{k+1}[1] = x_k[1] + x_k[5] \cdot 1$$

5. How well do we model the physics of the problem? $\rightarrow$ $w_k$ Modeled by: $Q = E[w_k w_k^T]$

# Kalman's Goal:

Goal:

Find: $\hat{x}_k$ which minimizes:

$$f(e_k) = (x_k - \hat{x}_k)^2$$

for every time step.

This can be also formulated as the minimization of the trace of the matrix:

$$\min trace(P_k) = \min trace(E[e_k e_k^T]) =$$
$$\min trace(E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T])$$

# Kalman's Solution:

## Prior Estimate:

The prior estimate of $\hat{x}_k$ is denoted $\hat{x}'_k$.

The prior estimate = was gained by knowledge of the system (system = model = $\Phi$).

---

## The posterior estimate of $x_k$ is $\hat{x}_k$

According to Kalman's derivation, the posterior is given by:

$$\hat{x}_k = \hat{x}'_k + K_k \cdot (z_k - H \cdot \hat{x}'_k)$$

This eqation ties together:

1. The prior: $\hat{x}'_k$
2. The error between the measurement ($z_k$) and the prior state estimate, translated to measurement coordinates: $H \cdot \hat{x}'_k$

# Kalman's Solution:

You've seen in class that it all boils down to the following scheme:

Iterate:

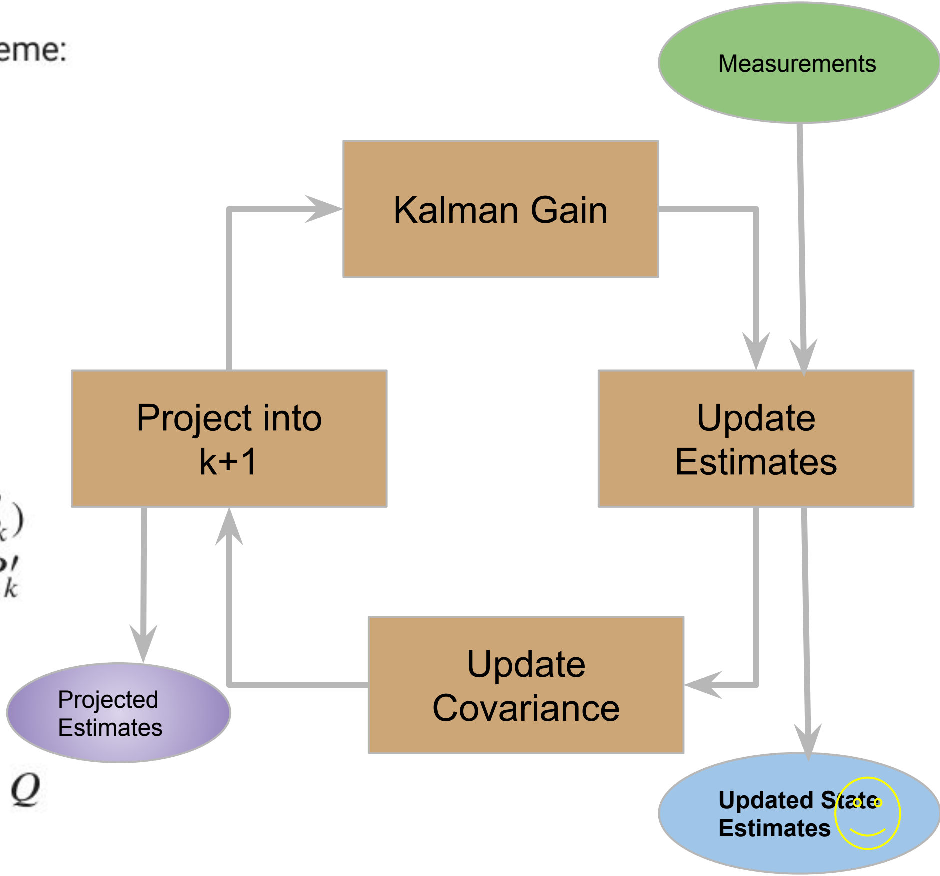
1. Update
2. Project

The Update Step:

1. Compute the Kalman Gain:
   $K_k = P'_k \cdot H^T \cdot (H \cdot P'_k \cdot H^T + R)^{-1}$
2. The posterior estimate: $\hat{x}_k = \hat{x}'_k + K_k \cdot (z_k - H \cdot \hat{x}'_k)$
3. Update the Covariance Matrix: $P_k = (I - K_k \cdot H) \cdot P'_k$

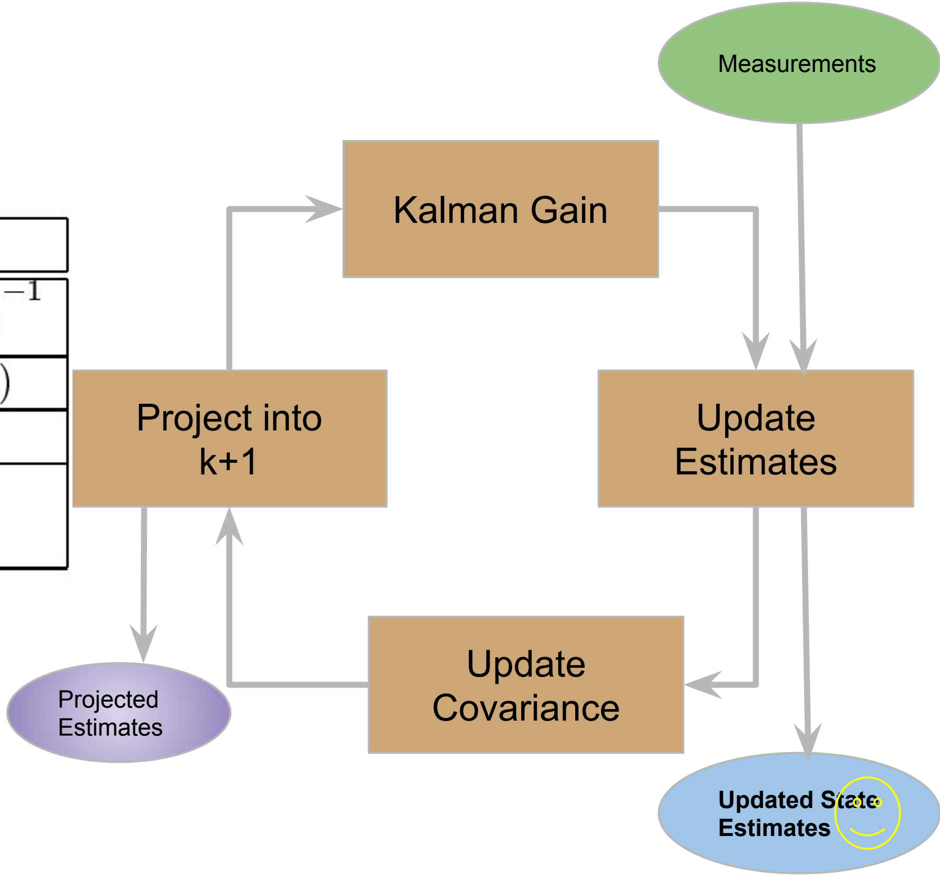
The Project Step:

1. State Projection: $\hat{x}'_{k+1} = \Phi \hat{x}_k$
2. Covariance Matrix Projectyion: $P'_{k+1} = \Phi \cdot P_k \cdot \Phi^T + Q$



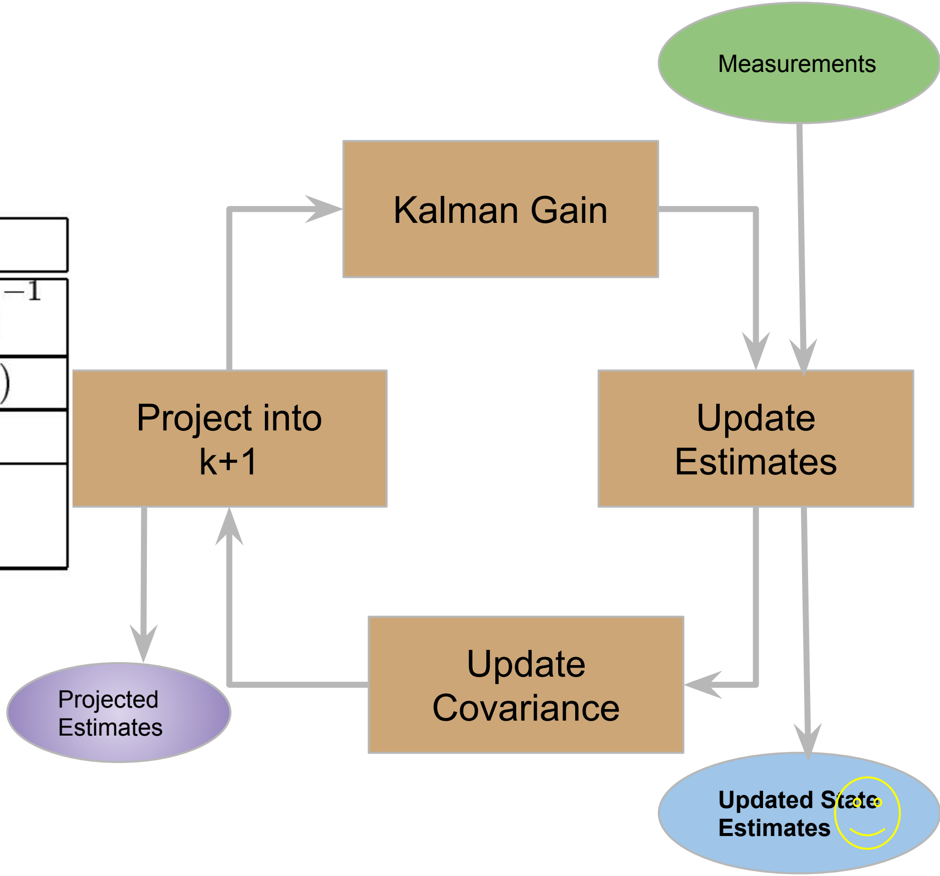
# Kalman's Solution:

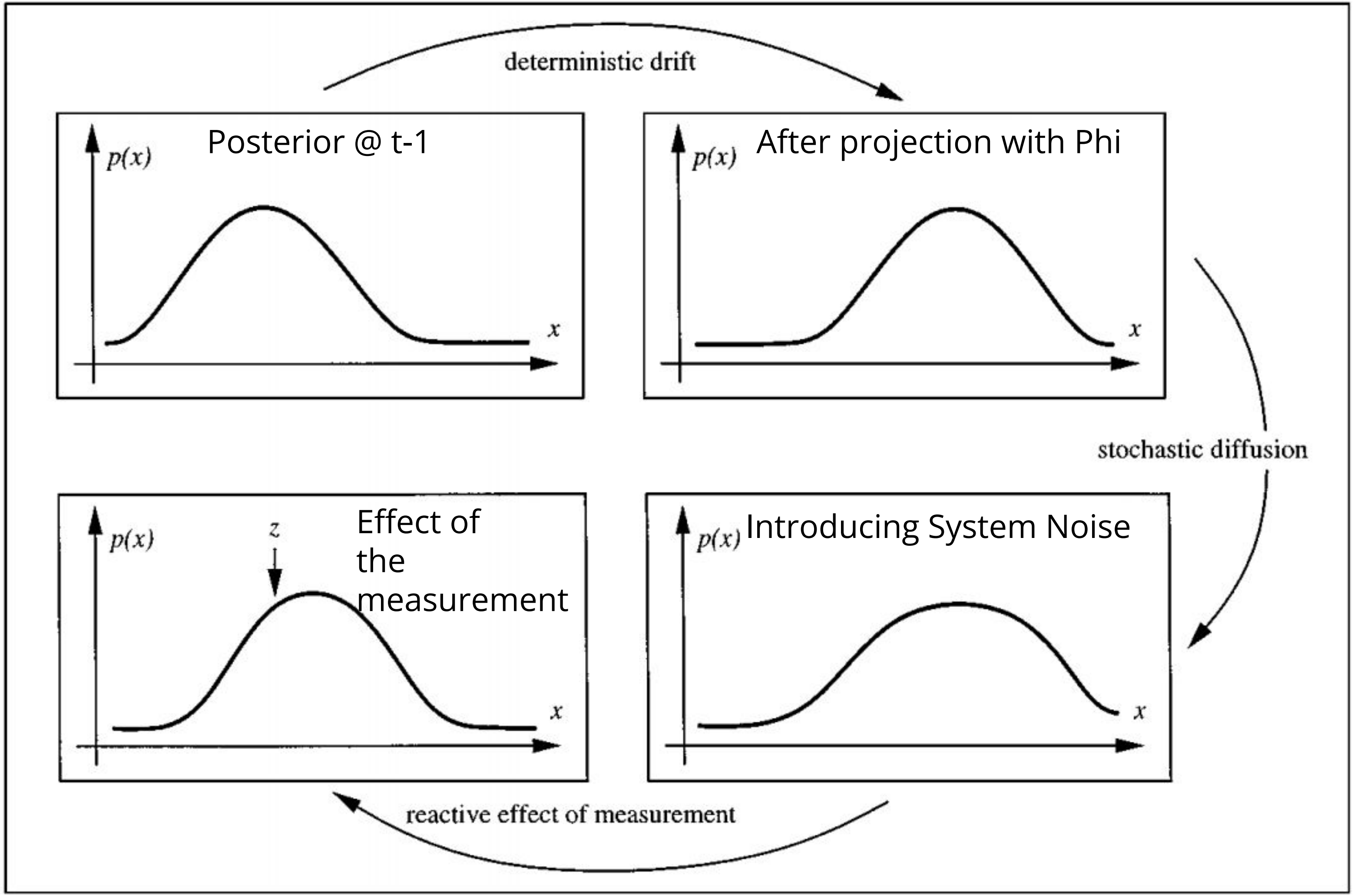
| Description | Equation |
|---|---|
| Kalman Gain | $K_k = P'_k H^T \left(H P'_k H^T + R\right)^{-1}$ |
| Update Estimate | $\hat{x}_k = \hat{x}'_k + K_k \left(z_k - H \hat{x}'_k\right)$ |
| Update Covariance | $P_k = \left(I - K_k H\right) P'_k$ |
| Project into $k+1$ | $\hat{x}'_{k+1} = \Phi \hat{x}_k$<br>$P_{k+1} = \Phi P_k \Phi^T + Q$ |



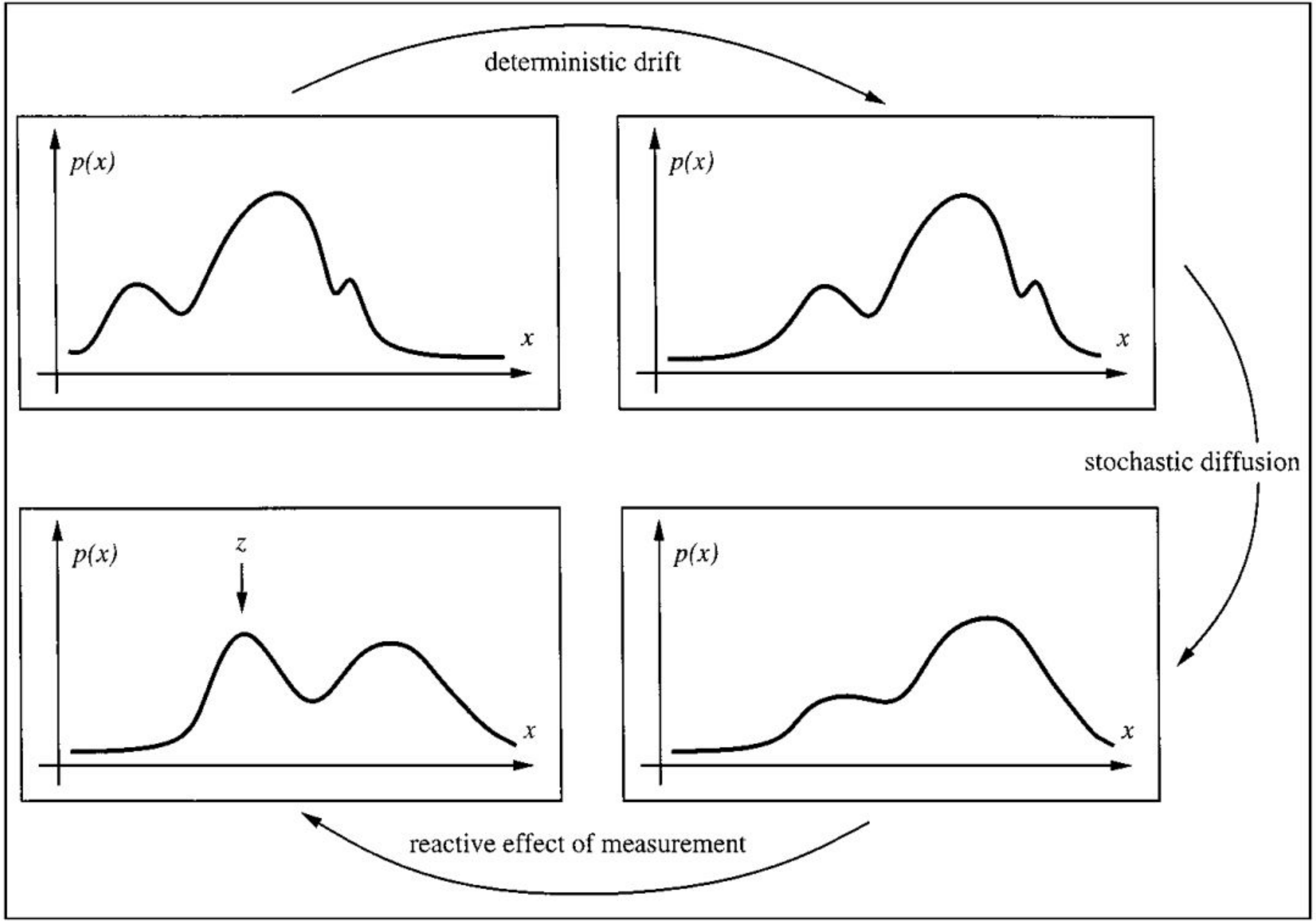
# Kalman's Solution:

| Description | Equation |
|---|---|
| Kalman Gain | $K_k = P'_k H^T \left( H P'_k H^T + R \right)^{-1}$ |
| Update Estimate | $\hat{x}_k = \hat{x}'_k + K_k \left( z_k - H \hat{x}'_k \right)$ |
| Update Covariance | $P_k = \left( I - K_k H \right) P'_k$ |
| Project into $k+1$ | $\hat{x}'_{k+1} = \Phi \hat{x}_k$ <br> $P_{k+1} = \Phi P_k \Phi^T + Q$ |




deterministic drift
Posterior @ t-1
p(x)
x
After projection with Phi
p(x)
x
stochastic diffusion
Introducing System Noise
p(x)
x
reactive effect of measurement
Effect of the measurement
p(x)
z
x

# Particle filter


deterministic drift
p(x)
x
p(x)
x
stochastic diffusion
p(x)
z
x
p(x)
x
reactive effect of measurement

# Particle Filter - Goal

Given the previous State - estimate the new state.

Input:

$$\{S_{t-1}^{(n)}, \pi_{t-1}^{(n)}, C_{t-1}^{(n)}\}_{n=1}^{N} = \{State, Weight, Slack\}$$

Output:

$$\{S_{t}^{(n)}, \pi_{t}^{(n)}, C_{t}^{(n)}\}_{n=1}^{N} = \{State, Weight, Slack\}$$

# Particle Filter - Partial Algorithm:

## Input:

$$\{S_{t-1}^{(n)}, \pi_{t-1}^{(n)}, C_{t-1}^{(n)}\}_{n=1}^{N}$$

## Output:

$$\{S_{t}^{(n)}, \pi_{t}^{(n)}, C_{t}^{(n)}\}_{n=1}^{N}$$

## The Algorithm:

1. Select $S_t'^{(n)}$ as follows:

   a) Generate r at random

   b) Find smallest j such that $C_{t-1}^{(j)} >= r$

   c) Set $S_t'^{(n)} = S_{t-1}^{(j)}$

2. Predict:

   $p(x_t | x_{t-1} = s_t'^{(n)})$

   i.e:

   $S_t^{(n)} = A \cdot S_t'^{(n)} + B \cdot w_t^{(n)}$

3. Measure: $\pi_t^{(n)} = p(z_t | x_t = s_t^{(n)}$

Return: $\sum_{n=1}^{N} \pi_t^{(n)} f(s_t^{(n)})$

# Particle Filter - Partial Algorithm:

Prior = Before measurement observed

## The Algorithm:

1. Select $S_t^{\prime(n)}$ as follows:

   a) Generate r at random

   b) Find smallest j such that $C_{t-1}^{(j)} >= r$

   c) Set $S_t^{\prime(n)} = S_{t-1}^{(j)}$

2. Predict:

   $p(x_t | x_{t-1} = s_t^{\prime(n)})$

   i.e:

   $S_t^{(n)} = A \cdot S_t^{\prime(n)} + B \cdot w_t^{(n)}$

3. Measure: $\pi_t^{(n)} = p(z_t | x_t = s_t^{(n)})$

Return: $\sum_{n=1}^{N} \pi_t^{(n)} f(s_t^{(n)})$

# Particle Filter - Partial Algorithm:

Prior = Before measurement observed

Apply deterministic drift according to the "physics"

## The Algorithm:

1. Select $S_t'^{(n)}$ as follows:
   a) Generate r at random
   b) Find smallest j such that $C_{t-1}^{(j)} >= r$
   c) Set $S_t'^{(n)} = S_{t-1}^{(j)}$
2. Predict:

   $p(x_t | x_{t-1} = s_t'^{(n)})$

   i.e:

   $S_t^{(n)} = A \cdot S_t'^{(n)} + B \cdot w_t^{(n)}$
3. Measure: $\pi_t^{(n)} = p(z_t | x_t = s_t^{(n)})$

Return: $\sum_{n=1}^{N} \pi_t^{(n)} f(s_t^{(n)})$

# Particle Filter - Partial Algorithm:

Prior = Before measurement observed

Apply deterministic drift according to the “physics”

System noise / stochastic diffusion

## The Algorithm:

1. Select $S_t'^{(n)}$ as follows:

   a) Generate r at random

   b) Find smallest j such that $C_{t-1}^{(j)} >= r$

   c) Set $S_t'^{(n)} = S_{t-1}^{(j)}$

2. Predict:

   $p(x_t | x_{t-1} = s_t'^{(n)})$

   i.e:

   $S_t^{(n)} = A \cdot S_t'^{(n)} + B \cdot w_t^{(n)}$

3. Measure: $\pi_t^{(n)} = p(z_t | x_t = s_t^{(n)})$

Return: $\sum_{n=1}^{N} \pi_t^{(n)} f(s_t^{(n)})$

# Particle Filter - Partial Algorithm:

Prior = Before measurement observed

Apply deterministic drift according to the "physics"

System noise / stochastic diffusion

Given that the state vector is indeed at $s_t^n$ how well does it describe the measurement

## The Algorithm:

1. Select $S_t'^{(n)}$ as follows:

   a) Generate r at random

   b) Find smallest j such that $C_{t-1}^{(j)} >= r$

   c) Set $S_t'^{(n)} = S_{t-1}^{(j)}$

2. Predict:

   $p(x_t | x_{t-1} = s_t'^{(n)})$

   i.e:

   $S_t^{(n)} = A \cdot S_t'^{(n)} + B \cdot w_t^{(n)}$

3. Measure: $\pi_t^{(n)} = p(z_t | x_t = s_t^{(n)})$

Return: $\sum_{n=1}^{N} \pi_t^{(n)} f(s_t^{(n)})$

# Particle Filter - Partial Algorithm:

Prior = Before measurement observed

Apply deterministic drift according to the "physics"

System noise / stochastic diffusion

Given that the state vector is indeed at $s_t^n$ how well does it describe the measurement

## The Algorithm:

1. Select $S_t'^{(n)}$ as follows:

   a) Generate r at random

   b) Find smallest j such that $C_{t-1}^{(j)} >= r$

   c) Set $S_t'^{(n)} = S_{t-1}^{(j)}$

2. Predict:

   $p(x_t | x_{t-1} = s_t'^{(n)})$

   i.e:

   $S_t^{(n)} = A \cdot S_t'^{(n)} + B \cdot w_t^{(n)}$

3. Measure: $\pi_t^{(n)} = p(z_t | x_t = s_t^{(n)})$

Return: $\sum_{n=1}^{N} \pi_t^{(n)} f(s_t^{(n)})$

Use template here

# Particle Filter - Partial Algorithm:

Prior = Before measurement observed

Apply deterministic drift according to the "physics"

System noise / stochastic diffusion

This is the score per particle n at time t.

## The Algorithm:

1. Select $S_t'^{(n)}$ as follows:

   a) Generate r at random

   b) Find smallest j such that $C_{t-1}^{(j)} >= r$

   c) Set $S_t'^{(n)} = S_{t-1}^{(j)}$

2. Predict:

   $p(x_t | x_{t-1} = s_t'^{(n)})$

   i.e:

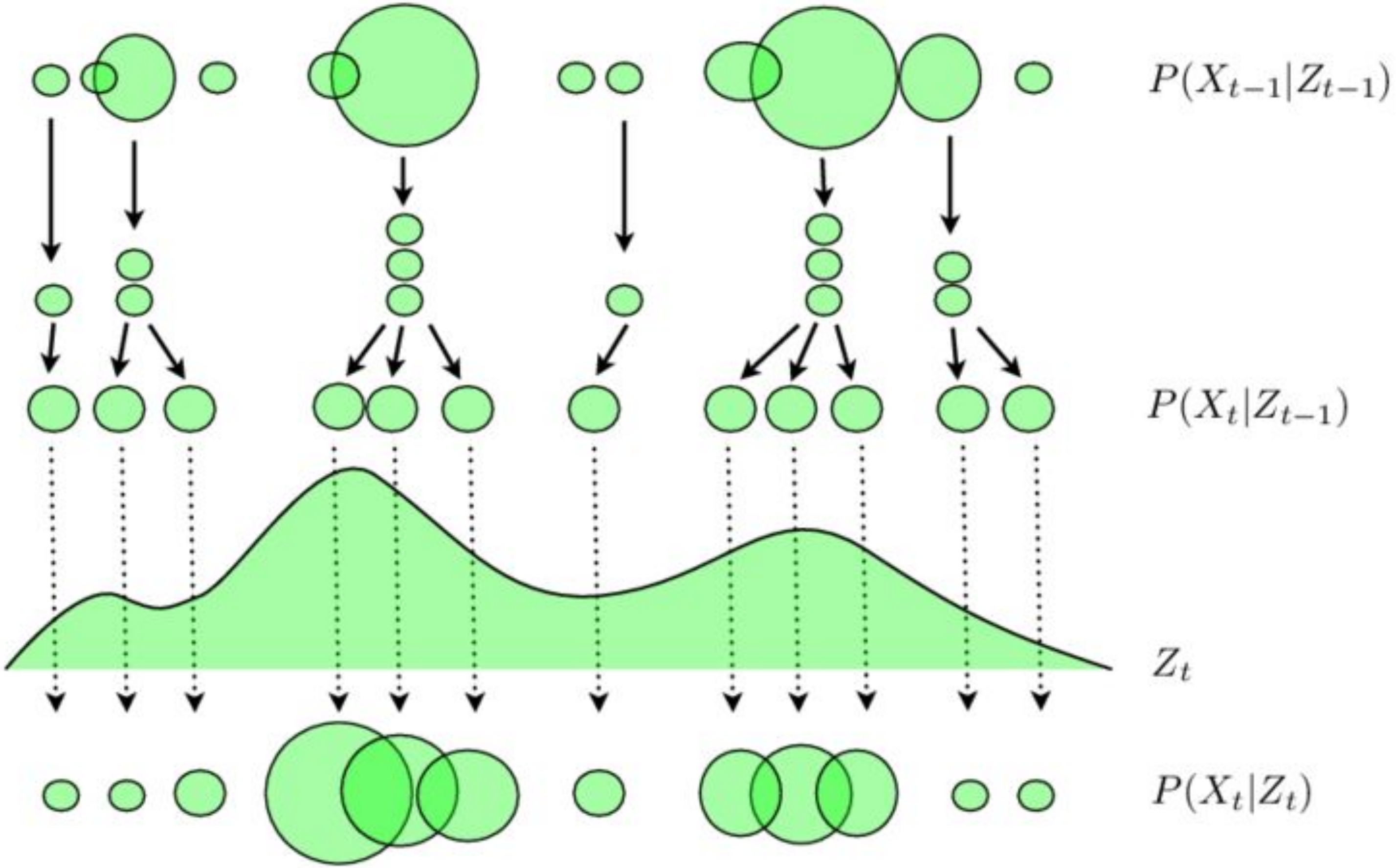
   $S_t^{(n)} = A \cdot S_t'^{(n)} + B \cdot w_t^{(n)}$

3. Measure: $\pi_t^{(n)} = p(z_t | x_t = s_t^{(n)})$

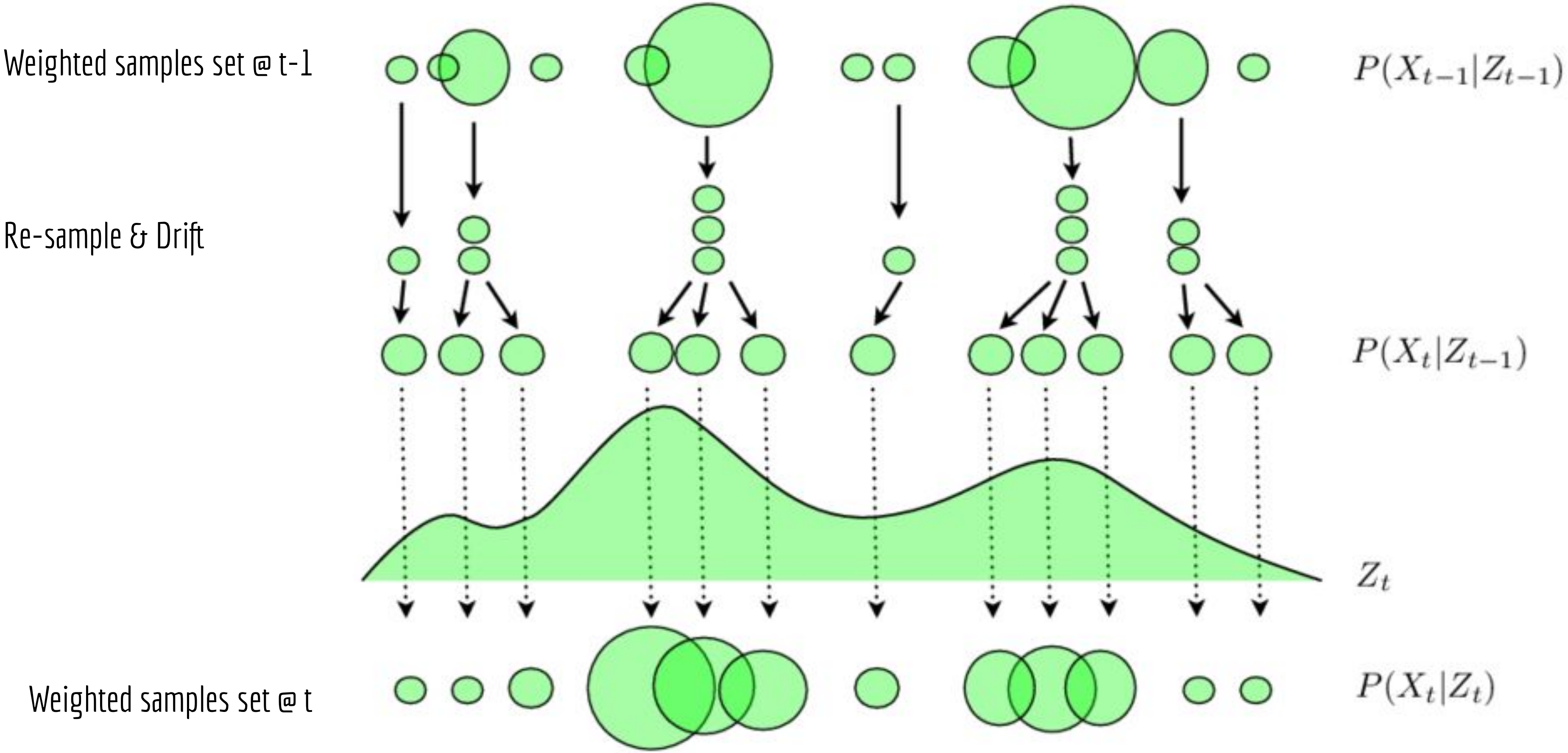
Return: $\sum_{n=1}^{N} \pi_t^{(n)} f(s_t^{(n)})$

Use template here

Weighted samples set @ t-1



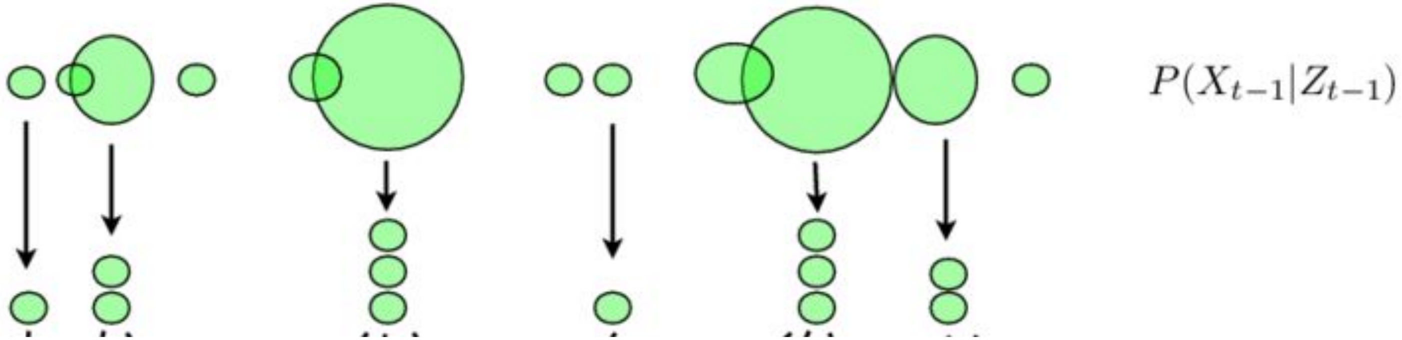

Weighted samples set @ t



Weighted samples set @ t-1
$P(X_{t-1}|Z_{t-1})$
Re-sample & Drift
$P(X_t|Z_{t-1})$
$Z_t$
Weighted samples set @ t
$P(X_t|Z_t)$

Weighted samples set @ t-1

Re-sample & Drift

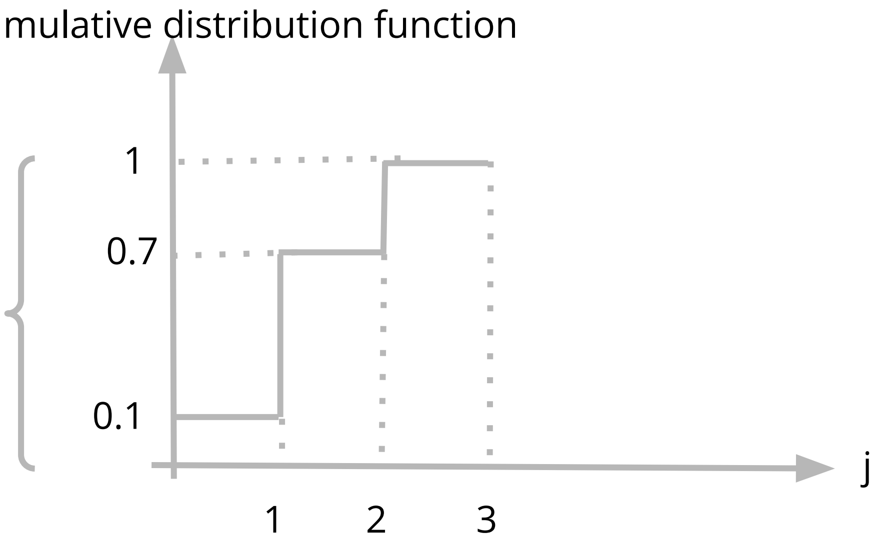


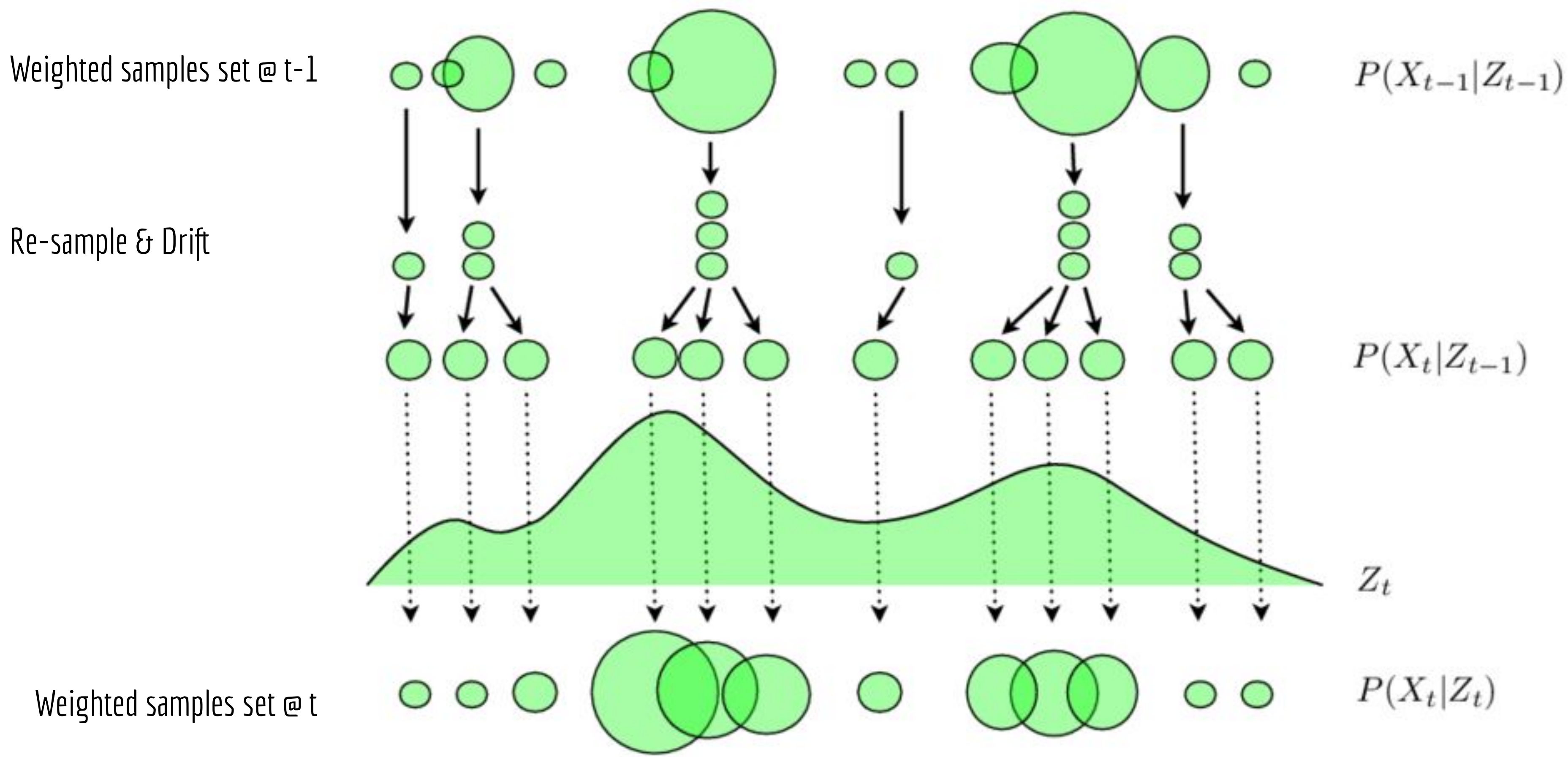

$$\pi_1 = 0.1$$

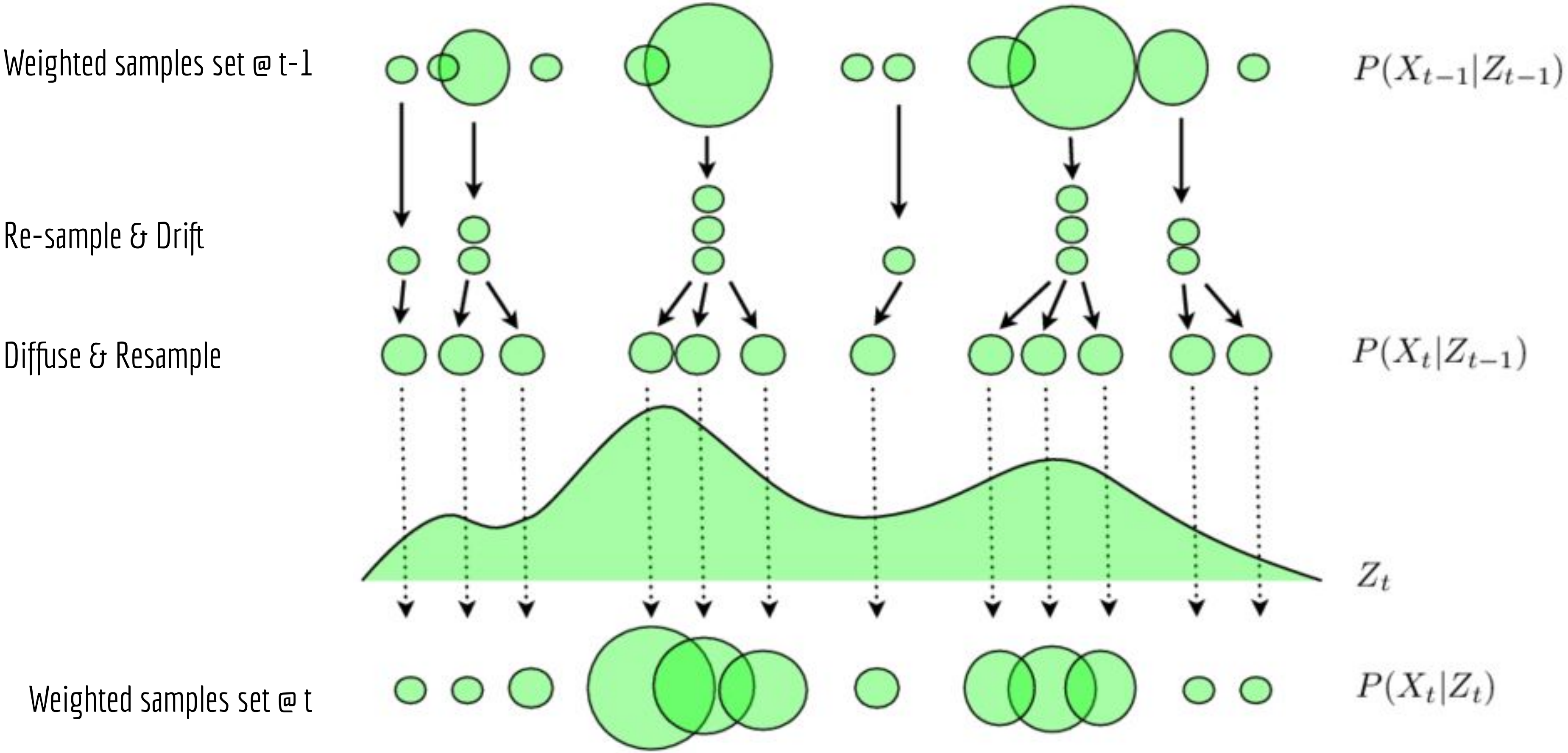
$$\pi_2 = 0.6$$

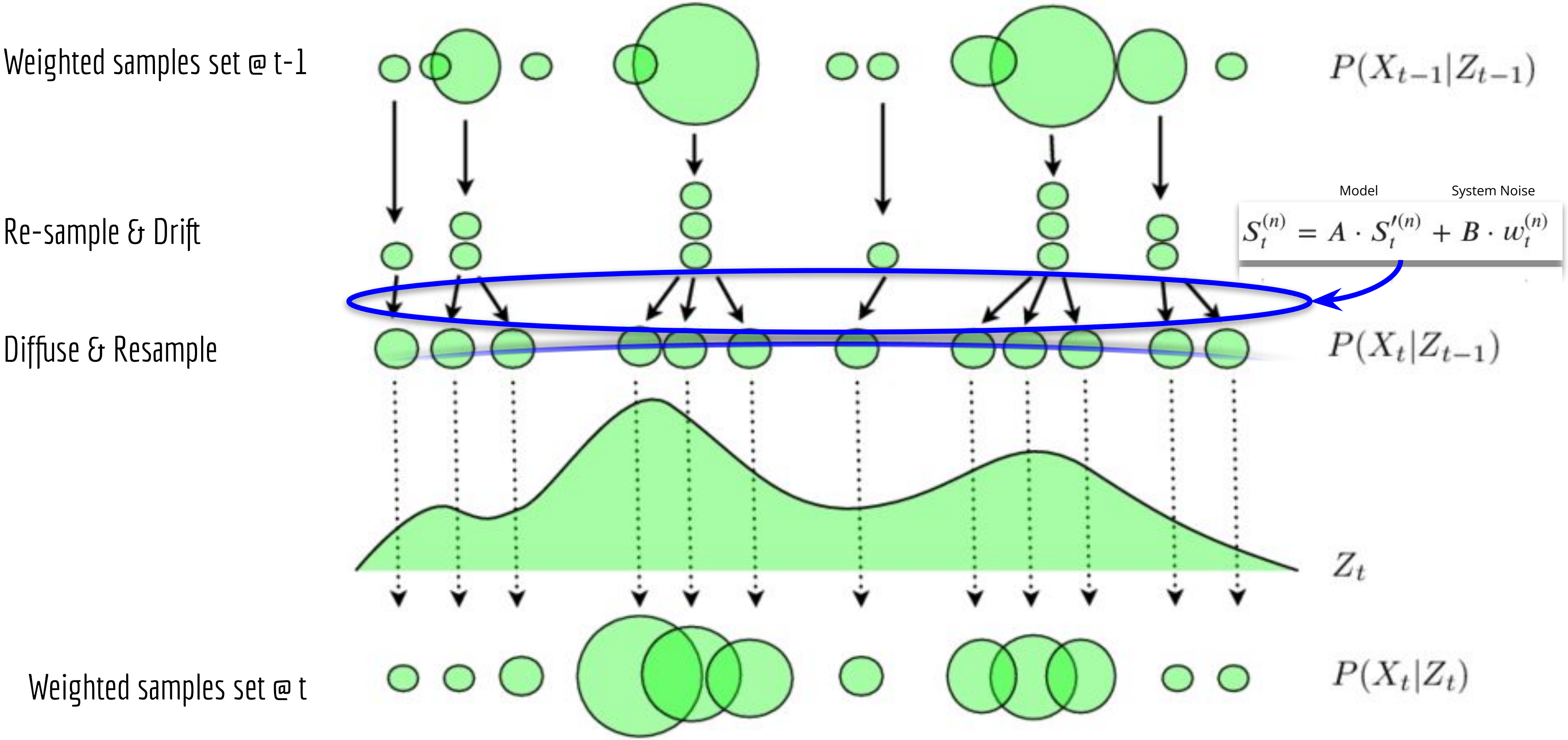
$$\pi_3 = 0.3$$

Sample r
between 0
and 1.

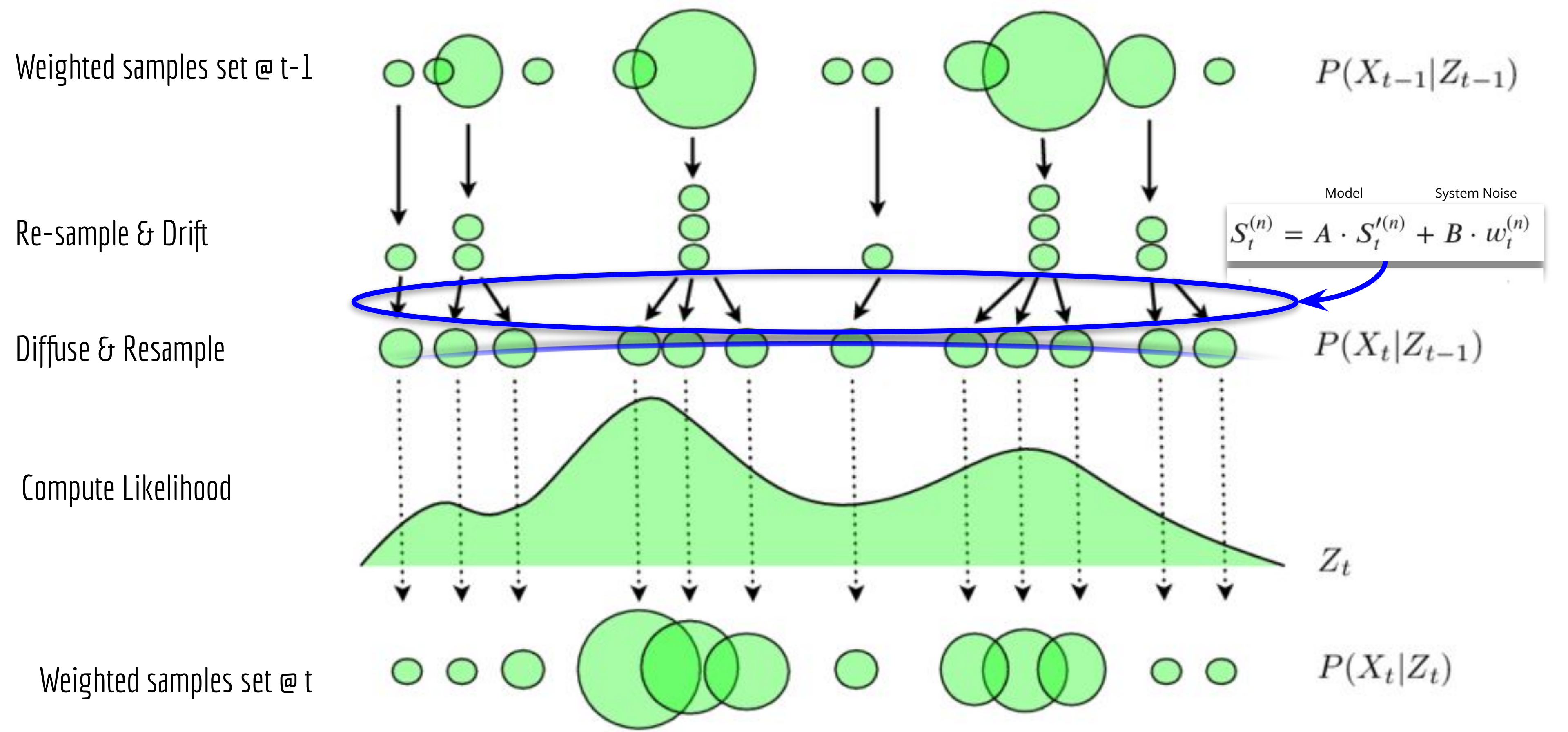



Weighted samples set @ t-1
Re-sample & Drift
Weighted samples set @ t
$P(X_{t-1}|Z_{t-1})$
$P(X_t|Z_{t-1})$
$Z_t$
$P(X_t|Z_t)$


Weighted samples set @ t-1
$P(X_{t-1}|Z_{t-1})$
Re-sample & Drift
Diffuse & Resample
$P(X_t|Z_{t-1})$
$Z_t$
Weighted samples set @ t
$P(X_t|Z_t)$



Weighted samples set @ t-1
$P(X_{t-1}|Z_{t-1})$
Model
System Noise
$S_t^{(n)} = A \cdot S_t^{\prime(n)} + B \cdot w_t^{(n)}$
Re-sample & Drift
Diffuse & Resample
$P(X_t|Z_{t-1})$
$Z_t$
Weighted samples set @ t
$P(X_t|Z_t)$


Weighted samples set @ t-1
$P(X_{t-1}|Z_{t-1})$
Model
System Noise
$S_t^{(n)} = A \cdot S_t'^{(n)} + B \cdot w_t^{(n)}$
Re-sample & Drift
Diffuse & Resample
$P(X_t|Z_{t-1})$
Compute Likelihood
$Z_t$
Weighted samples set @ t
$P(X_t|Z_t)$

# Let’s code it...