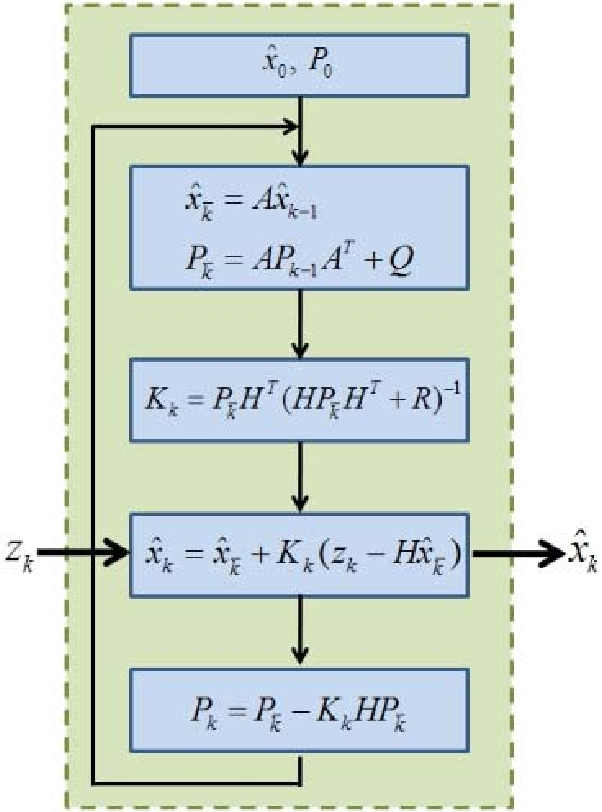
Unscented Kalman filter

2019.04.08

차례

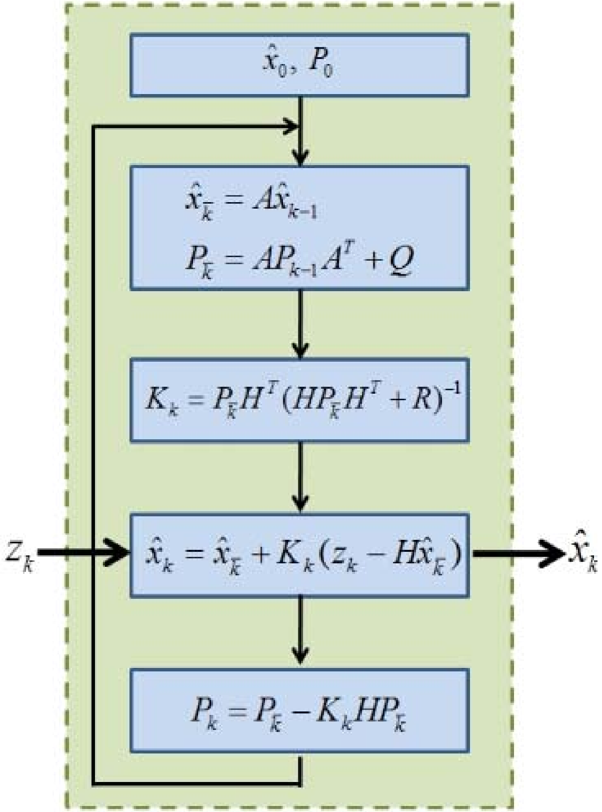
1. Unscented Kalman Filter

1. Unscented Kalman Filter

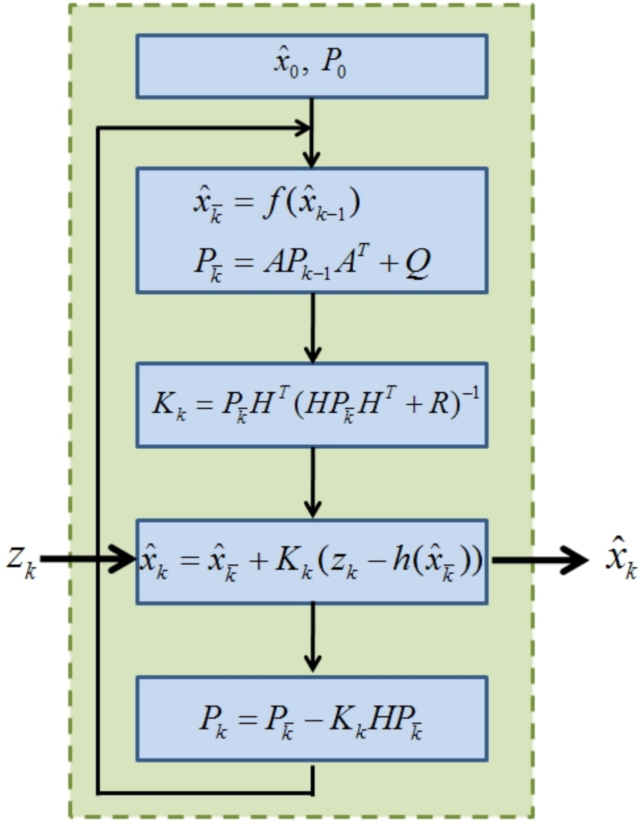


Kalman Filter algorithm

1. Unscented Kalman Filter

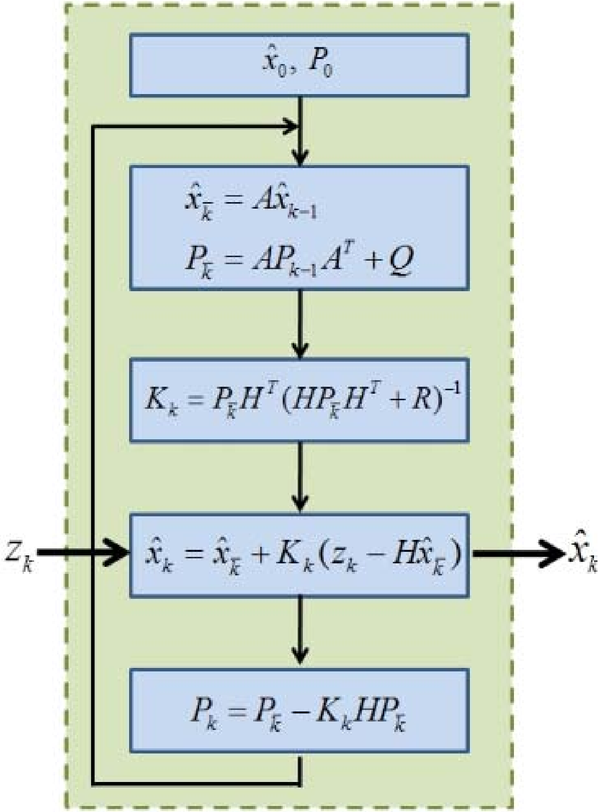


Kalman Filter algorithm

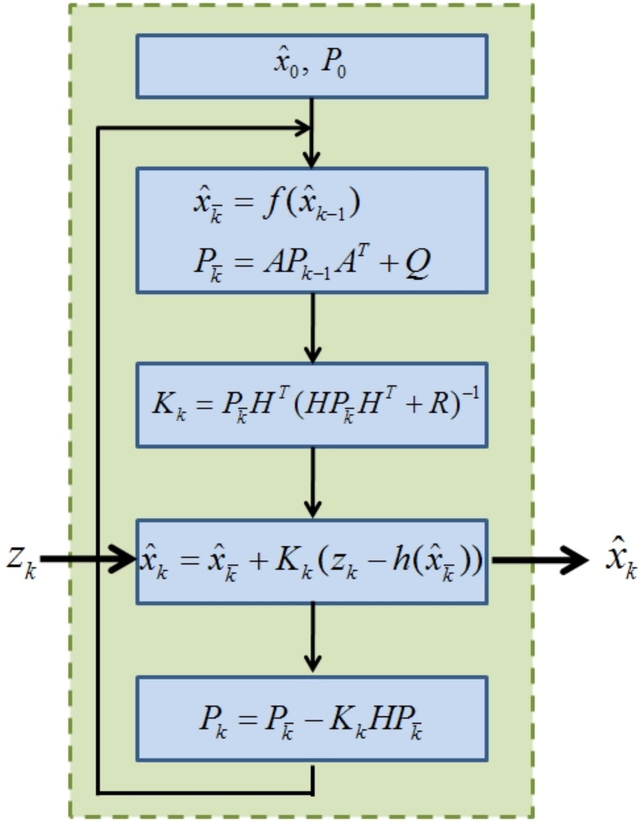


EKF algorithm

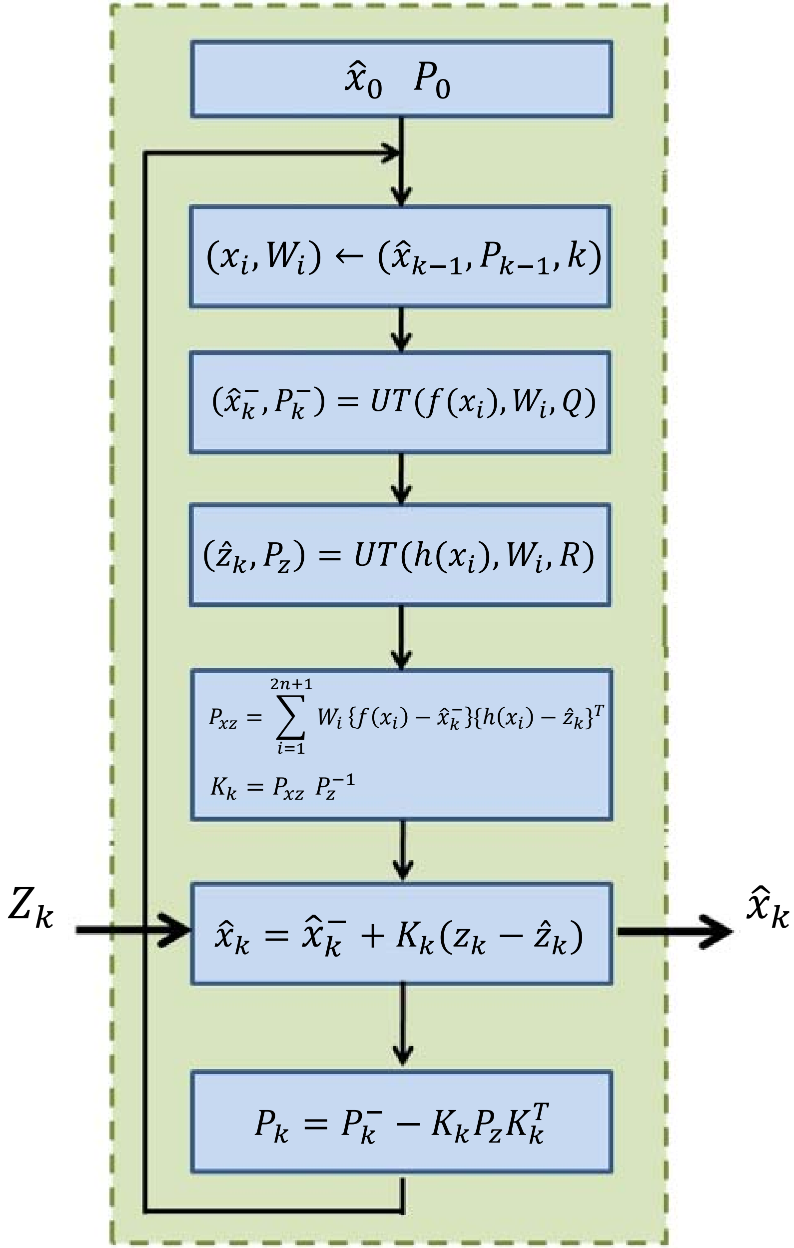
1. Unscented Kalman Filter



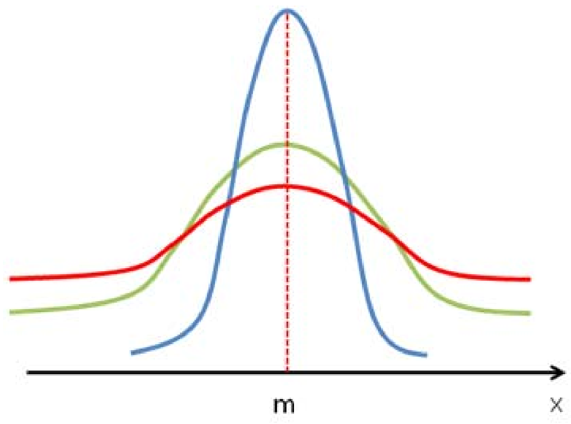
Kalman Filter algorithm



EKF algorithm



1. Unscented Kalman Filter



평균이 x_m, 공분산은 P_x인 정규분포를 따르는 상태변수 x를 고려한다.

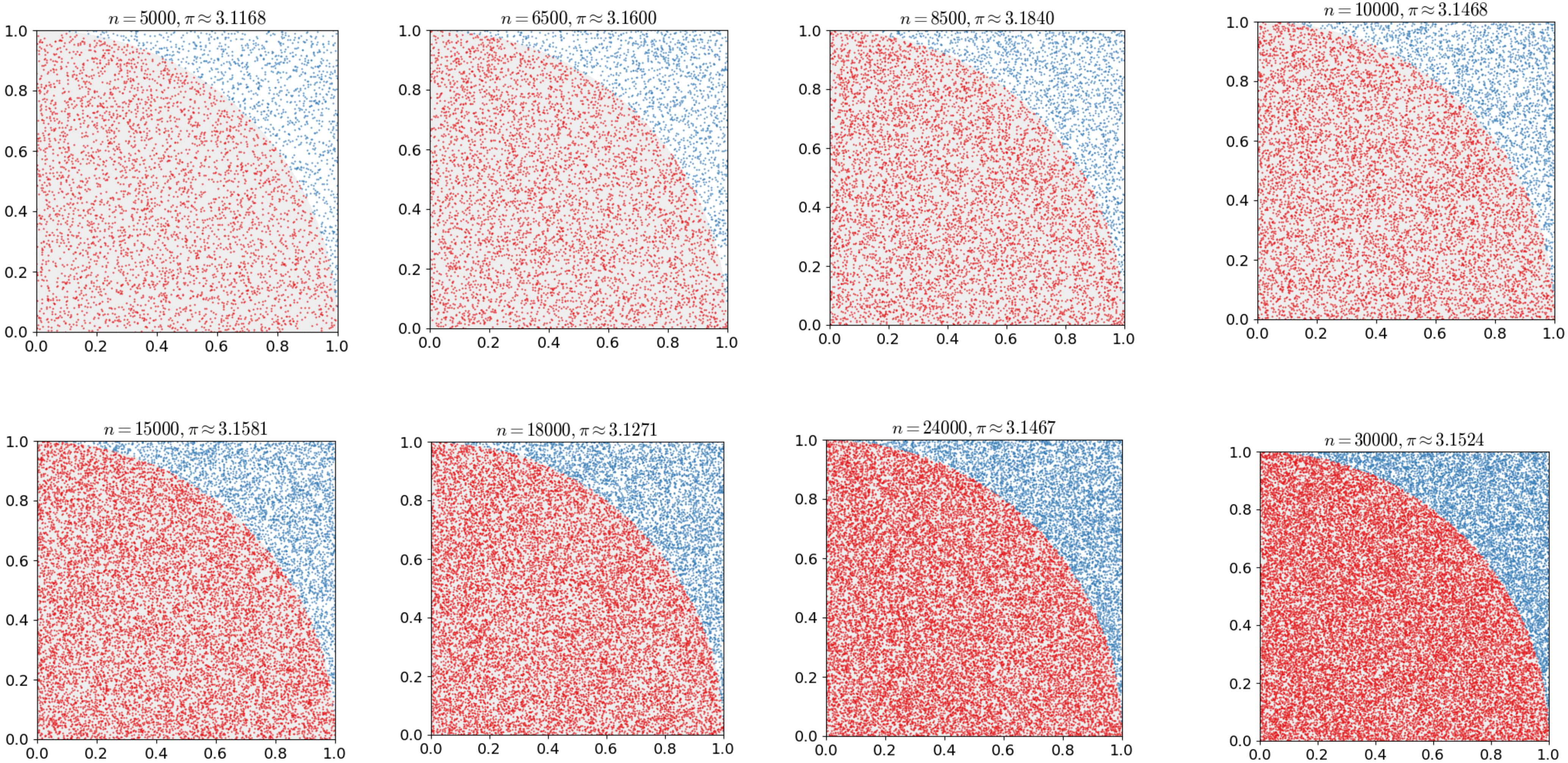
$$x \sim N(x_m, P_k)$$

x를 임의의 함수 f(-)로 변환하면, f(x)의 평균과 공분산은 어떻게 될 것인가?

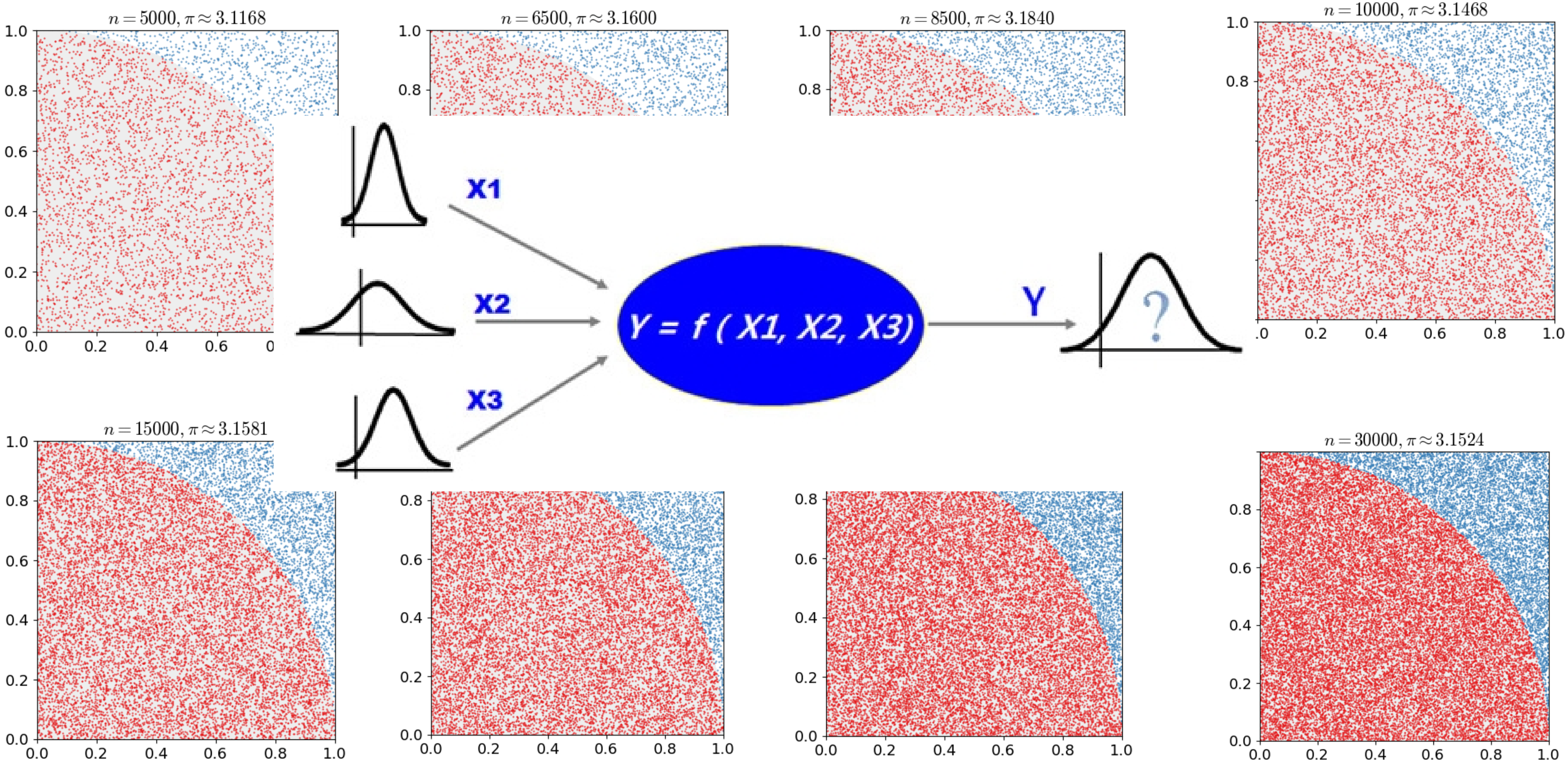
f(x)~?

몬테카를로 시뮬레이션은 계산 양이 많다… 또한, 무작위로 샘플을 추출하는 몬테카를로 시뮬레이션과 달리, Unscented 변환은 샘플과 각 샘플의 가중치를 정교하게 선정한다. 그러므로 몬테카를로 보다 훨씬 적은 수의 샘플만 있으면 된다.

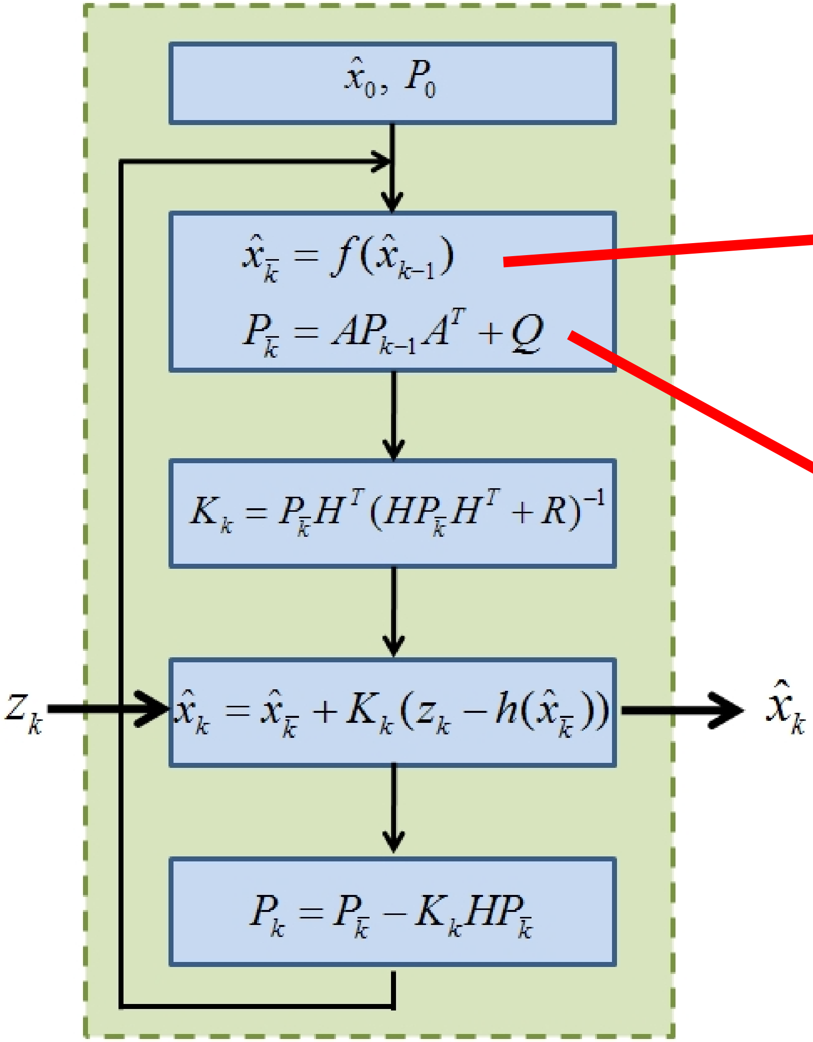
1. Unscented Kalman Filter



1. Unscented Kalman Filter



1. Unscented Kalman Filter



EKF algorithm

확장 칼만

$$\hat{x}_{k+1}^{-} = \boxed{f(\hat{x}_{k-1})}$$

선형 칼만

$$\hat{x}_{k+1}^{-} = \boxed{A\hat{x}_{k-1}}$$

$$f(\hat{x}_{k-1}) \Leftrightarrow A\hat{x}_{k-1}$$

$$\boxed{A \equiv \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_k}}$$

$$P_{k+1}^{-} = AP_kA^T + Q$$

1. Unscented Kalman Filter

먼저, x에 대해 다음과 같은 시그마 포인트 x_i와 가중치 W_i 를 정의한다.

$$x_1 = x_m$$
$$x_{i+1} = x_m + u_i \qquad i = 1,2,\dots,n$$
$$x_{i+n+1} = x_m - u_i \qquad i = 1,2,\dots,n$$

$$W_1 = \frac{k}{n+k} \qquad W_{i+n+1} = \frac{k}{2(n+k)} \qquad i = 1,2,\dots,n$$
$$W_{i+1} = \frac{k}{2(n+k)}$$

여기서, u_i는 다음과 같은 행렬 U의 행벡터이고 k는 임의의 상수이다.

$$U^T U = (n+k)P_k$$

그러면 함수 y=f(x)의 평균과 공분산은 다음과 같이 구할 수 있다.

$$y_m = \sum_{i=1}^{2n+1} W_i f(x_i) \qquad P_y = \sum_{i=1}^{2n+1} W_i \{f(x_i) - y_m\}\{f(x_i) - y_m\}^T$$

여기서 x_i는 Unscented 변환의 시그마 포인트이고 W_i는 가중치이다.

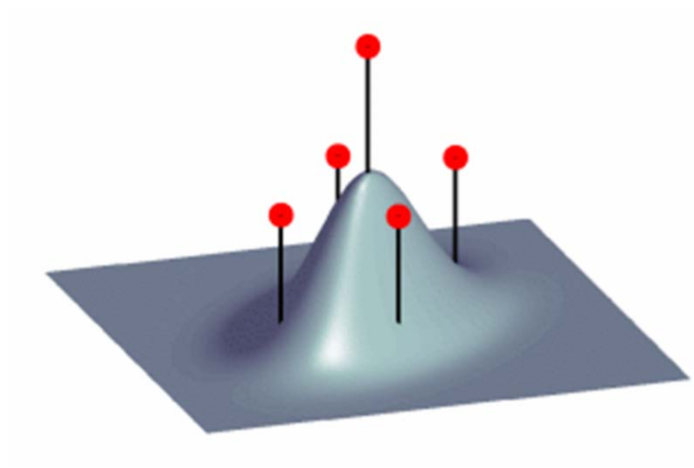
1. Unscented Kalman Filter

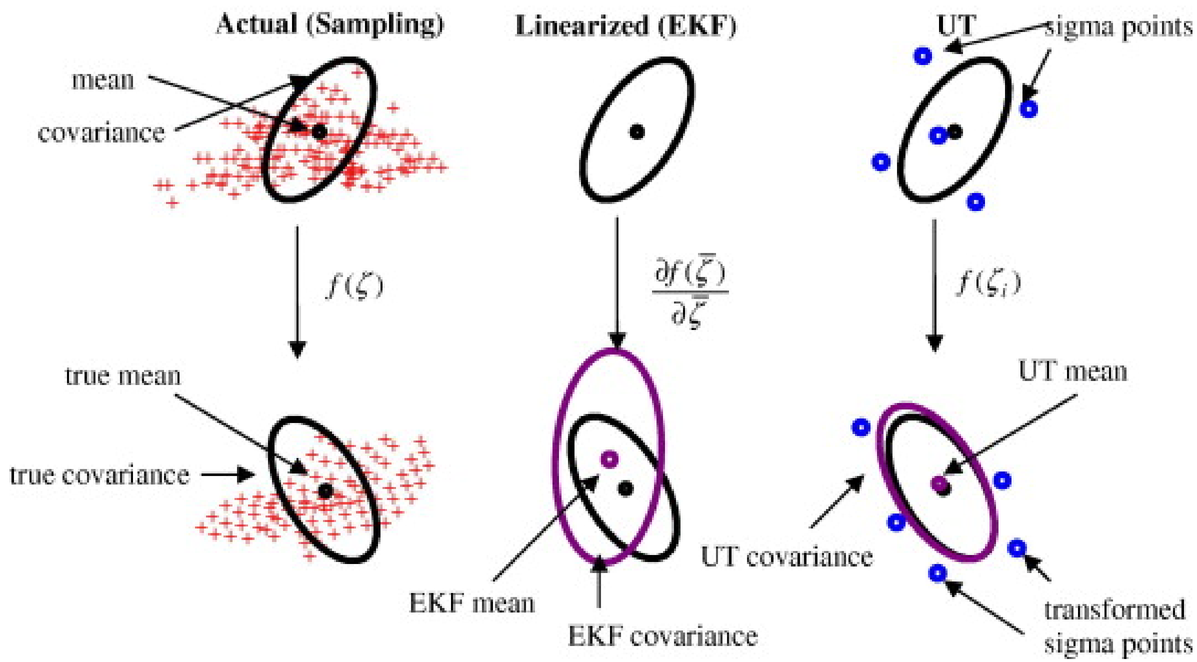
시그마 포인트의 가중 평균

$$x_m = \sum_{i=1}^{2n+1} W_i x_i$$

시그마 포인트의 가중 공분산

$$P_x = \sum_{i=1}^{2n+1} W_i \{x_i - x_m\}\{x_i - x_m\}^T$$





1. Unscented Kalman Filter

```
function [Xi W] = SigmaPoints(xm, P, kappa)
%
n  = numel(xm);
Xi = zeros(n, 2*n+1);                  % sigma points = col of Xi
W  = zeros(n, 1);

Xi(:, 1) = xm;
W(1)     = kappa / (n + kappa);

U = chol((n+kappa)*P);                 % U'*U = (n+kappa)*P

for k=1:n
  Xi(:, k+1) = xm + U(k, :)';          % row of U
  W(k+1)     = 1 / (2*(n+kappa));
end

for k=1:n
  Xi(:, n+k+1) = xm - U(k, :)';
  W(n+k+1)     = 1 / (2*(n+kappa));
end
```

1. Unscented Kalman Filter

```
function [xm xcov] = UT(Xi, W)
%
%
[n, kmax] = size(Xi);

xm = 0;
for k=1:kmax
  xm = xm + W(k)*Xi(:, k);
end

xcov = zeros(n, n);
for k=1:kmax
  xcov = xcov + W(k)*(Xi(:, k) - xm)*(Xi(:, k) - xm)';
end
xcov = xcov;
```

$$x_m = \sum_{i=1}^{2n+1} W_i x_i$$

$$P_x = \sum_{i=1}^{2n+1} W_i \{x_i - x_m\}\{x_i - x_m\}^T$$

1. Unscented Kalman Filter

선형 시스템 모델

$$x_{k+1} = \boxed{Ax_k} + w_k$$

$$z_k = \boxed{Hx_k} + v_k$$

비선형 시스템 모델

$$x_{k+1} = \boxed{f(x_k)} + w_k$$

$$z_k = \boxed{h(x_k)} + v_k$$

x_k는 상태 변수

z_k는 측정값

A는 상태전이행렬

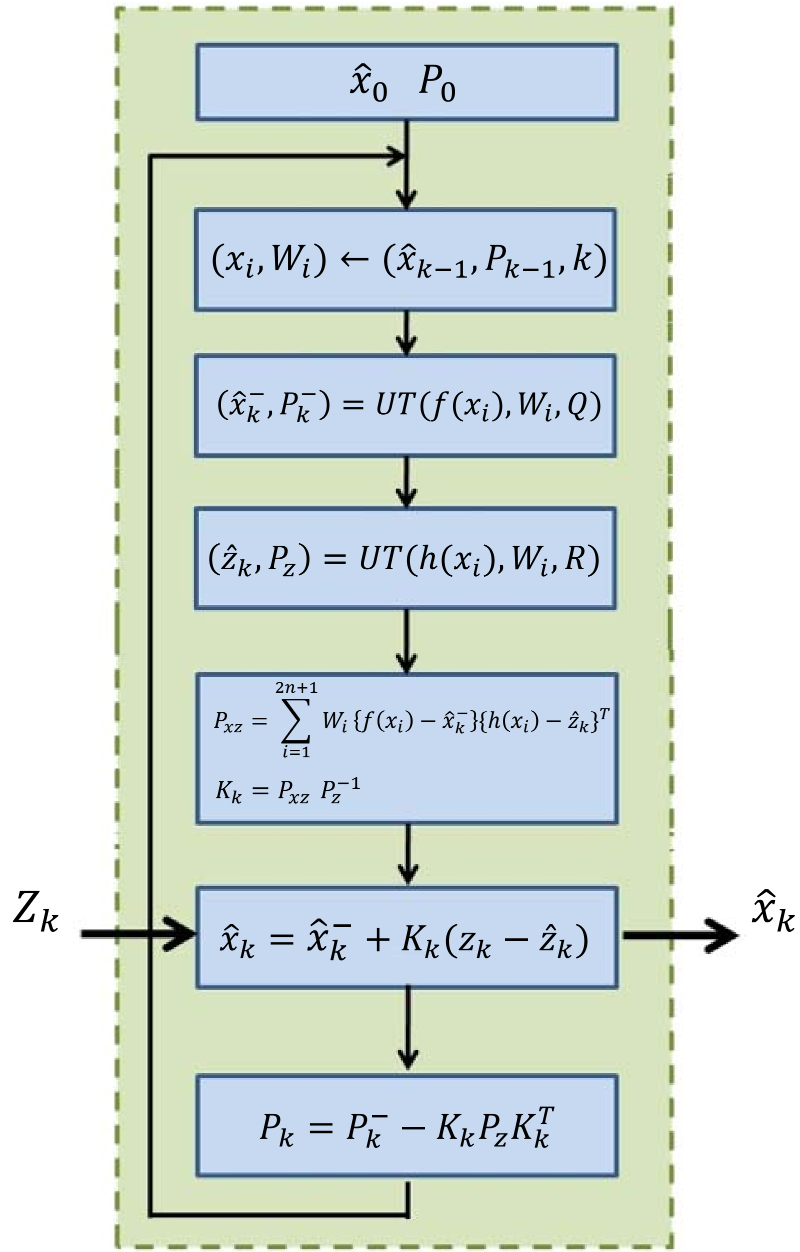
H는 mXn 행렬

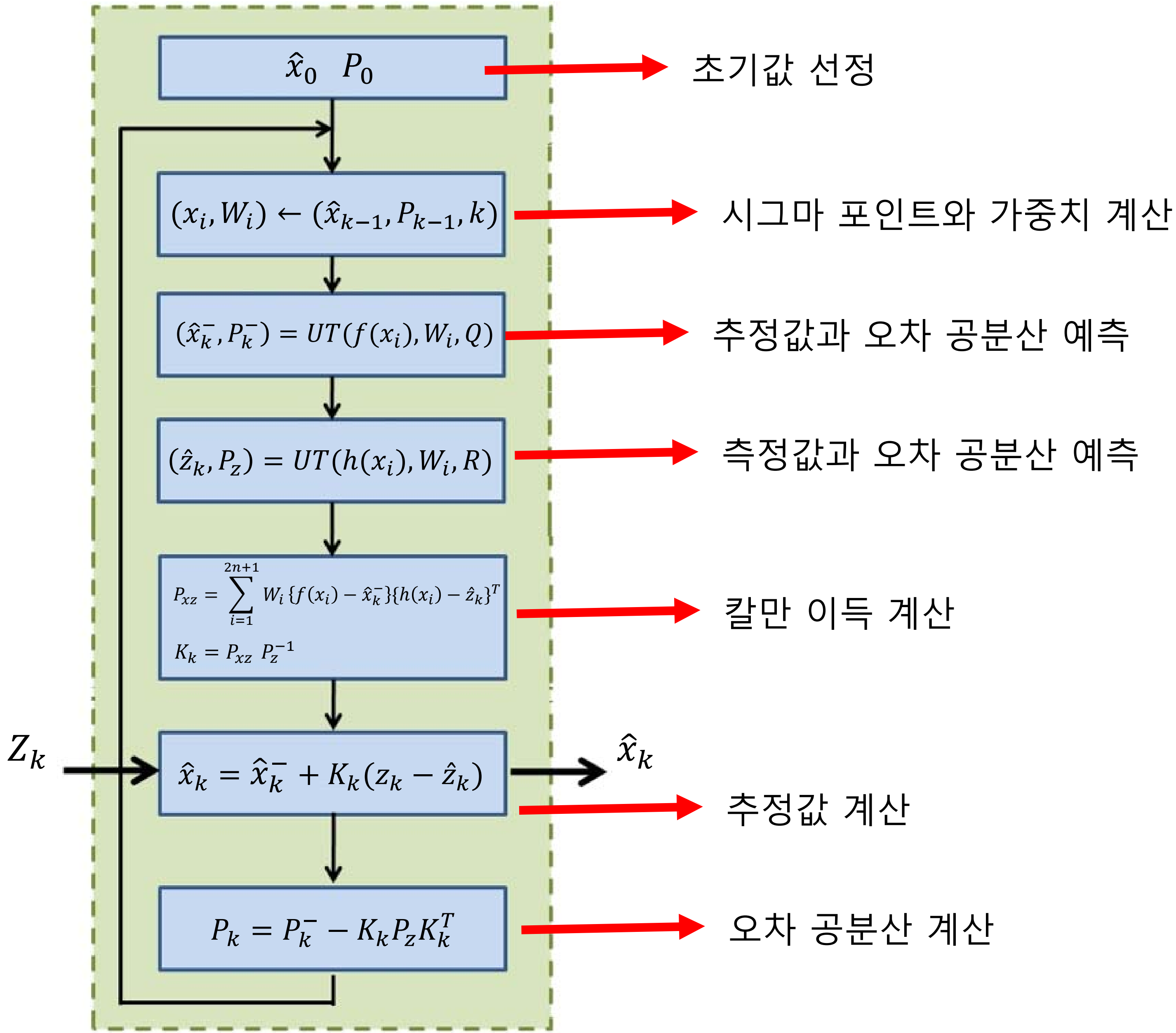
w_k는 잡음

w_k는 측정 잡음

$Q = w_k$의 공분산 행렬

$R = v_k$의 공분산 행렬




$\hat{x}_0 \quad P_0$
초기값 선정
$(x_i, W_i) \leftarrow (\hat{x}_{k-1}, P_{k-1}, k)$
시그마 포인트와 가중치 계산
$(\hat{x}_k^-, P_k^-) = UT(f(x_i), W_i, Q)$
추정값과 오차 공분산 예측
$(\hat{z}_k, P_z) = UT(h(x_i), W_i, R)$
측정값과 오차 공분산 예측
$P_{xz} = \sum_{i=1}^{2n+1} W_i \{f(x_i) - \hat{x}_k^-\}\{h(x_i) - \hat{z}_k\}^T$
$K_k = P_{xz} P_z^{-1}$
칼만 이득 계산
Z_k
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - \hat{z}_k)$
$\hat{x}_k$
추정값 계산
$P_k = P_k^- - K_k P_z K_k^T$
오차 공분산 계산

2. RADAR Program using UKF

시스템의 상태변수는 다음과 같이 정의한다.

$$x = \begin{Bmatrix} \text{수평거리} \\ \text{이동속도} \\ \text{고도} \end{Bmatrix} \equiv \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix}$$

물체의 이동 속도와 고도는 일정하므로 시스템의 운동 방정식은 다음과 같다.

$$\begin{Bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{Bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} + \begin{Bmatrix} 0 \\ w_1 \\ w_2 \end{Bmatrix} \equiv Ax + w$$

수평거리 미분 $\dot{x}_1 = x_2$

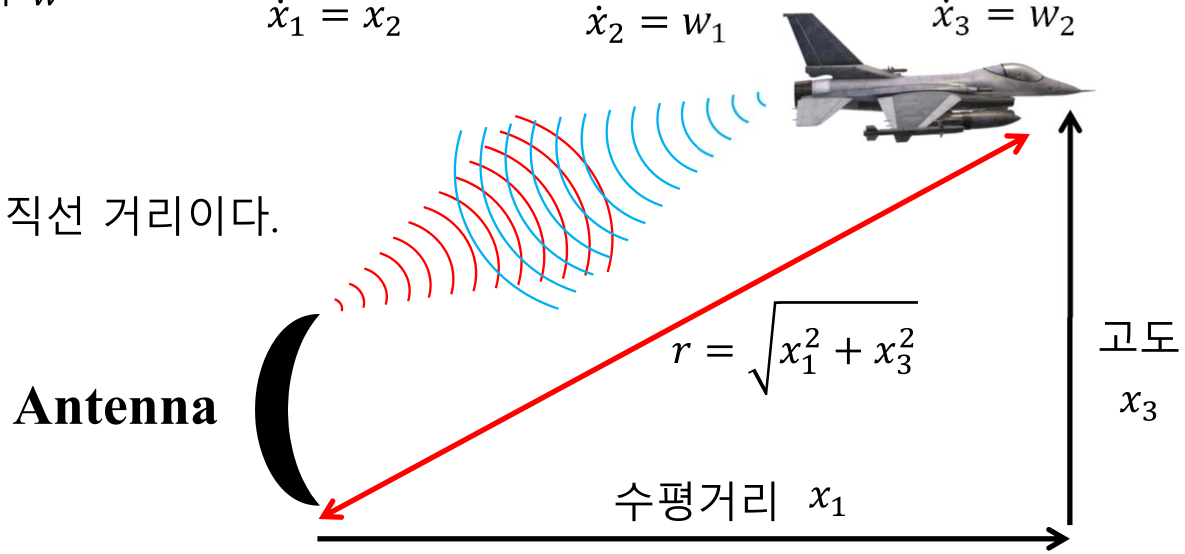
이동속도 미분 $\dot{x}_2 = w_1$

고도 미분 $\dot{x}_3 = w_2$

측정 모델 설계

- 레이더가 측정하는 값은 이동 물체까지의 직선 거리이다.

$$r = \sqrt{x_1^2 + x_3^2} + v \quad \equiv h(x) + v$$



2. RADAR Program using UKF

$$x_{k+1} = f(x_k) + w_k \qquad Q = w_k\text{의 공분산 행렬}$$

$$z_k = h(x_k) + v_k \qquad R = v_k\text{의 공분산 행렬}$$

$$\boxed{f(\hat{x}_{k-1}) \Leftrightarrow A\hat{x}_{k-1}}$$

$$\boxed{h(\hat{x}_k^-) \Leftrightarrow H\hat{x}_k^-}$$

x_k는 상태 변수

z_k는 측정값

A는 상태전이행렬

H는 mXn 행렬

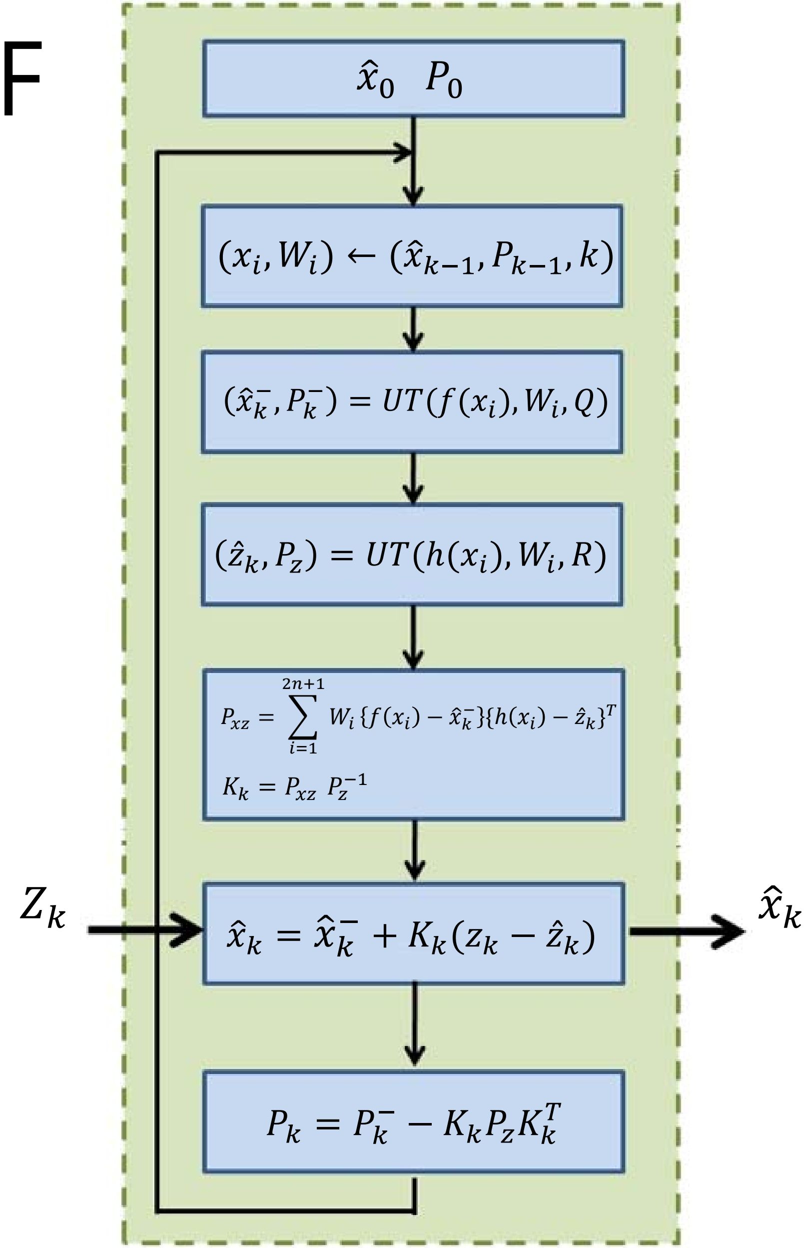
w_k는 잡음

w_k는 측정 잡음

$$\begin{Bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{Bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} + \begin{Bmatrix} 0 \\ w_1 \\ w_2 \end{Bmatrix} \equiv Ax + w$$

$$A = I + \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \times dt$$

$$r = \sqrt{x_1^2 + x_3^2} + v \quad \equiv h(x) + v$$



2. RADAR Program using UKF

```
function [pos vel alt] = RadarUKF(z, dt)
%
if isempty(firstRun)
  Q = [ 0.01   0       0;
        0      0.01    0;
        0      0       0.01 ];

  R = 100;
  x = [0 90 1100]';
  P = 100*eye(3);
  n = 3;
  m = 1;

  firstRun = 1;
end

[Xi W] = SigmaPoints(x, P, 0);

fXi = zeros(n, 2*n+1);
for k = 1:2*n+1
  fXi(:, k) = fx(Xi(:,k), dt);
end

[xp Pp] = UT(fXi, W, Q);
%norm(xp - fx(x, dt))

hXi = zeros(m, 2*n+1);
for k = 1:2*n+1
  hXi(:, k) = hx(fXi(:,k));
end
```

측정값의 영향을 덜 받고 변화가 완만한 추정값을 얻고 싶다면 행렬 R을 키우고 Q를 줄여야한다.

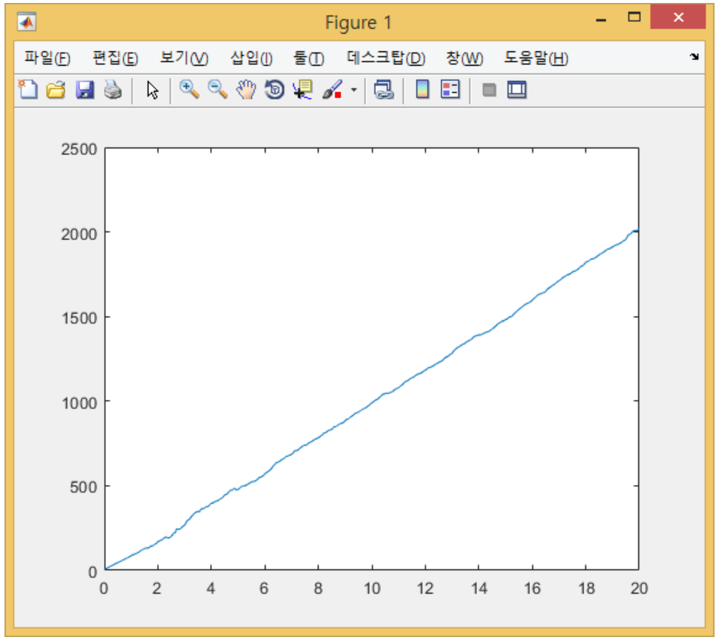
```
[zp Pz] = UT(hXi, W, R);

Pxz = zeros(n, m);
for k = 1:2*n+1
  Pxz = Pxz + W(k)*(fXi(:, k)-xp)*(hXi(:, k) - zp)';
end

K = Pxz*inv(Pz);
x = xp + K*(z - zp);
P = Pp - K*Pz*K';


pos = x(1);
vel = x(2);
alt = x(3);

%------------------------------
function xp = fx(x, dt)
%
A = eye(3) + dt*[ 0 1 0;
                  0 0 0;
                  0 0 0 ];

xp = A*x;

%------------------------------
function yp = hx(x)
%
yp = sqrt(x(1)^2 + x(3)^2);
```

$$A = I + \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \times dt$$

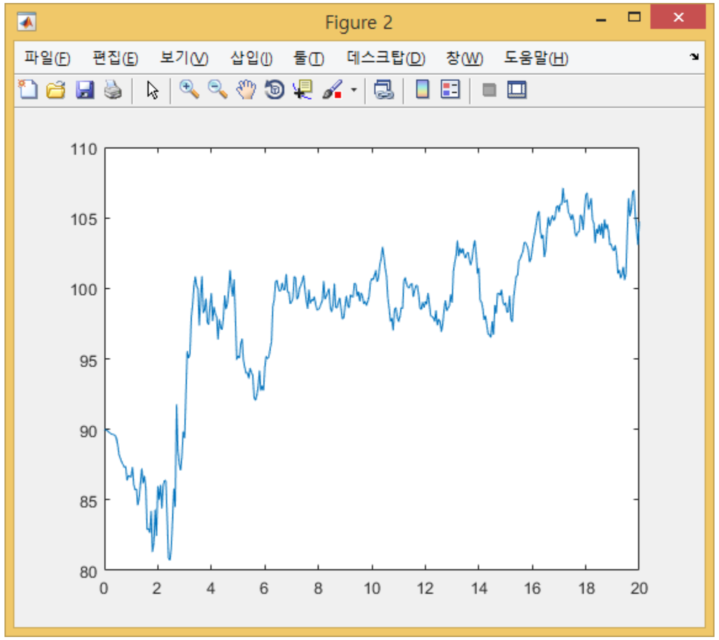
$$r = \sqrt{x_1^2 + x_3^2}$$

2. RADAR Program using UKF

거리



속도



고도