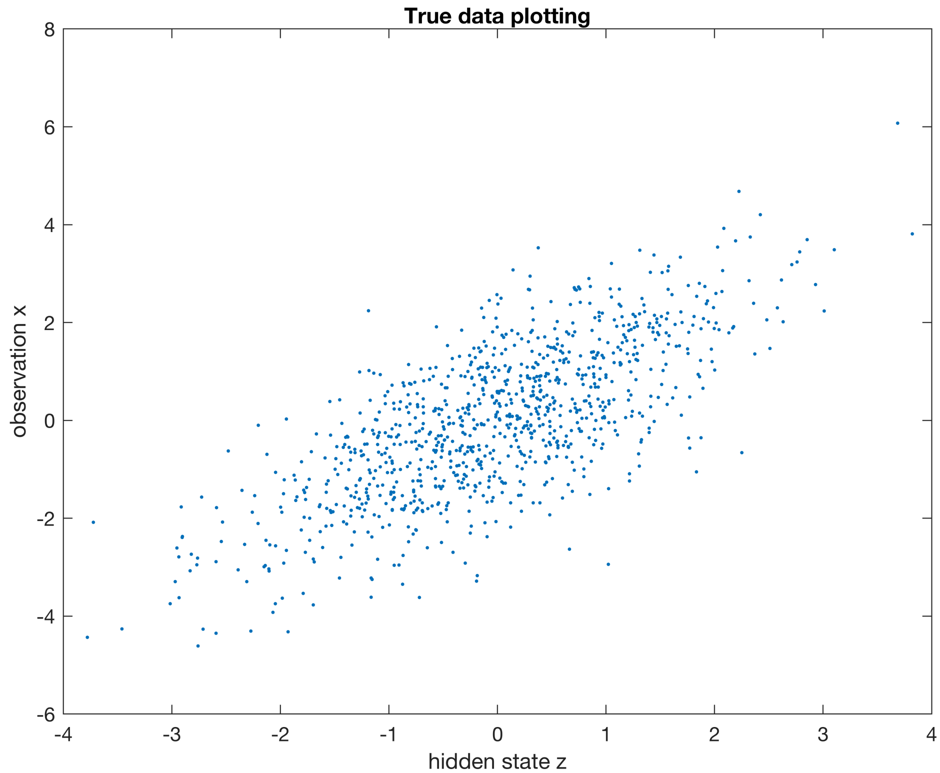
```
clear
clc
```

Generate the true state sequence z's and the observation sequence x's

```
% true underlying state sequence z
z0 = 0;
gamma = 0.5;
delta = normrnd(0,1,[1,1000]);
num_samples = 1000;

true_z = zeros(1, num_samples) ;
true_z(1) = gamma*z0 + delta(1);

for i=2:num_samples
    true_z(i) = gamma*true_z(i-1) + delta(i);
end


% generate x
x = zeros(1, num_samples);
for i=1:num_samples
    x(i) = normrnd(true_z(i), 1);
end

% visualize the original data
figure(1)
plot(true_z, x,'.')
title("True data plotting")
xlabel('hidden state z')
ylabel('observation x')
```



Initialization: Given x, initialize (\alpha, \beta), (\gamma, \delta), with \alpha, \gamma != 0

```
gamma = rand;

% gamma is non zero
while(gamma == 0)
    gamma = rand;
end

% z = zeros(1, num_samples);
% z(1) = rand;
% for i=2:num_samples
%     z(i) = gamma*z(i-1);
% end
```

Repeat until covergence:

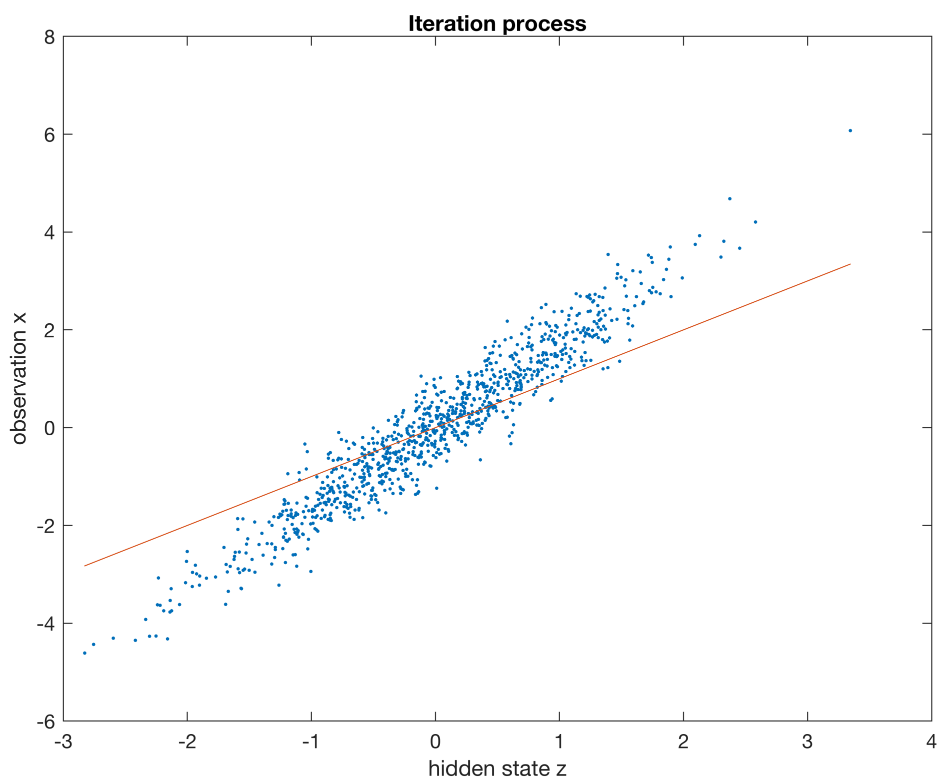
Step 1: Given x, z from initialization (if first iteration) or derived from step 2, using the poor man procedure, we adjust \gamma to minimize mean of y and w.

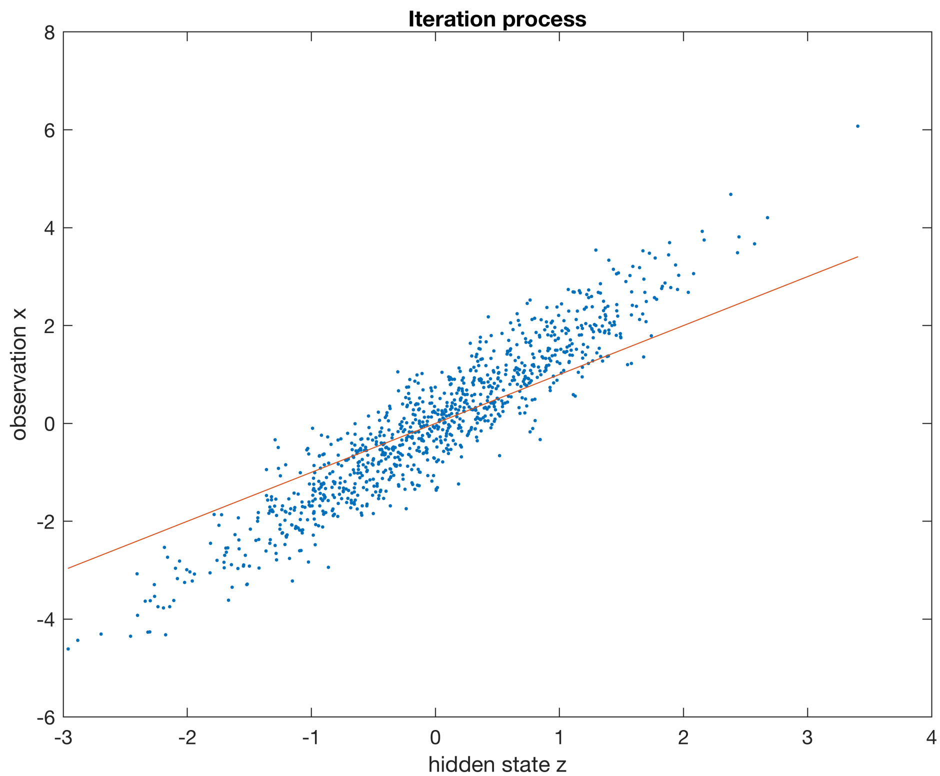
Step 2: Given z, z_0 = 0 and \gamma that are previously found in step 1, we adjust z to minimize the variance of y and w

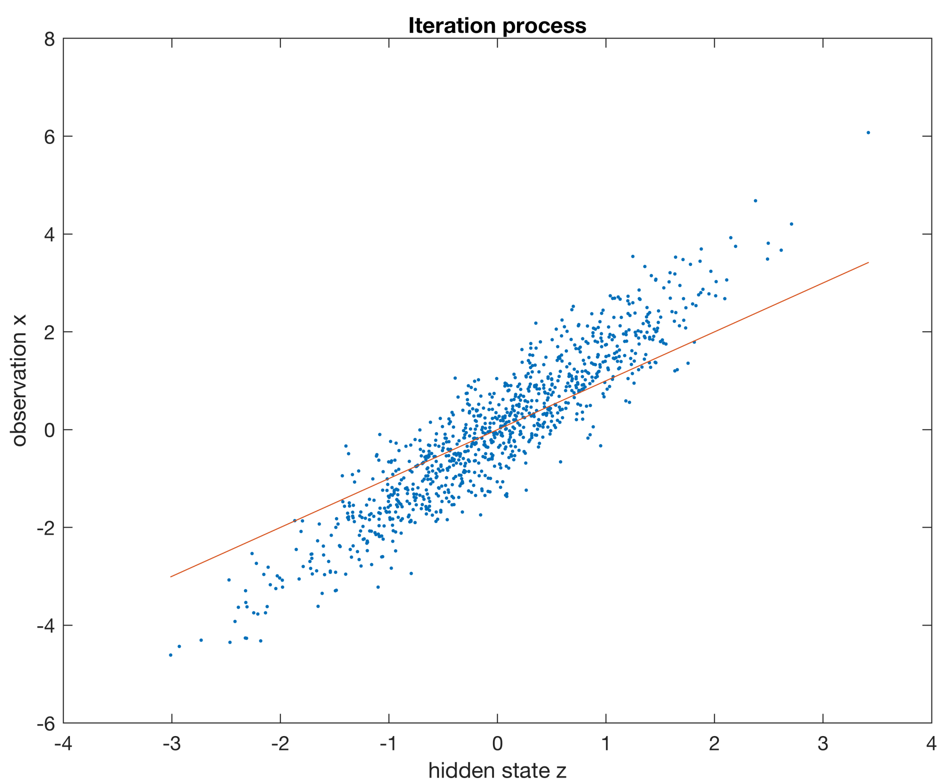
```
threshold = 0.000000001;
```

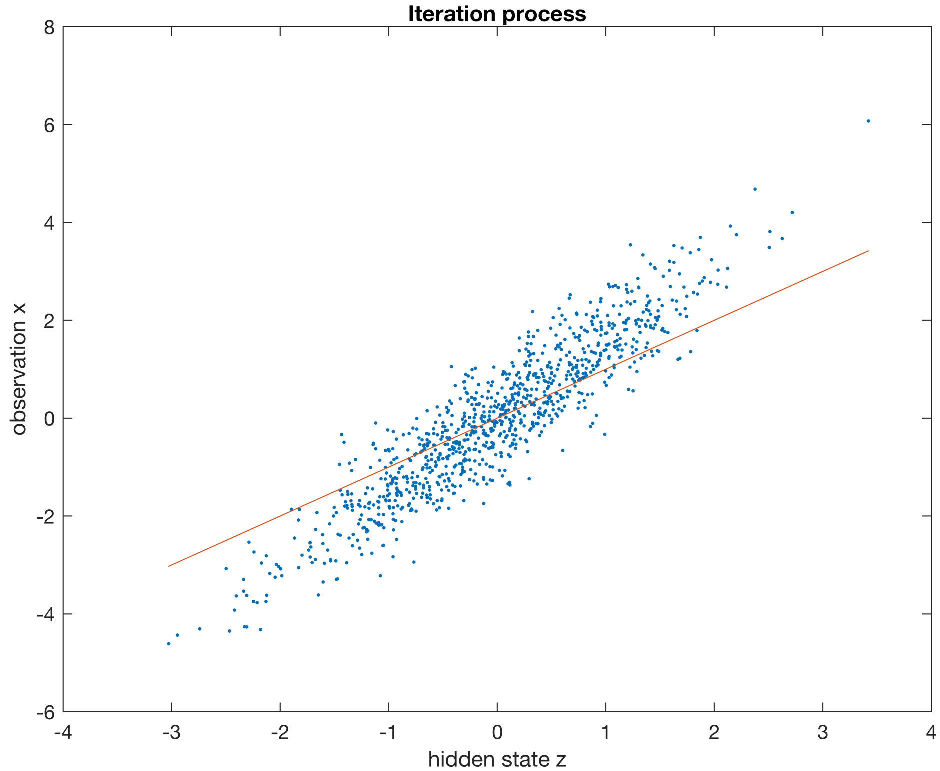
```
all_MSE = zeros(1, 0);
cnt = 0;
while (true)

    cnt = cnt+1;

    % STEP 1
    % calculate A, c in Az = c
    u = -2 * gamma;
    v = 4+ 2 * gamma^2;
    c = zeros(1, num_samples);
    for i=1:num_samples
        c(i) = 2*x(i);
    end


    % calculate LU decomposition of A
    r = zeros(1, num_samples);
    t = zeros(1, num_samples);

    t(1) = v;
    for i=2:num_samples
        r(i) = u/t(i-1);
        if i < num_samples
            t(i) = v - r(i)*u ;
        else t(i) = v + u - r(i)*u ;
        end
    end


    % calculate z
    % Az = LUz = C. Let Uz = D => LD = C
    % Calculate D
    d = zeros(1, num_samples);
    d(1) = c(1);

    for i=2:num_samples
        d(i) = c(i) - r(i)*d(i-1);
    end
    % Calculate Z
    curr_z = zeros(1, num_samples);
    curr_z(num_samples) = d(num_samples)/t(num_samples);
    for i = num_samples-1:-1:1
       curr_z(i) = (d(i) - u*curr_z(i+1))/t(i);
    end

    if cnt > 1
        error = abs(z-curr_z).^2;
        MSE = sum(error(:)) / num_samples;
        all_MSE = [all_MSE MSE];
        z = curr_z;

        if (MSE < threshold)
            break
```

```
        end
    end

    z = curr_z;

    % STEP 2
    % calcualte gamma
    z_prev = zeros(1, num_samples);
    z_prev(1) = z0;
    for i=2:num_samples
        z_prev(i) = z(i-1);
    end

    gamma = pinv(z_prev * z_prev') * (z_prev * z');

    % visualize the original data
    figure(cnt)
    x_hat = z';
    plot(z, x,'.')
    title("Iteration process")
    xlabel('hidden state z')
    ylabel('observation x')
    hold on
    % Plot the best fit line.
    plot(z, x_hat)

end
```

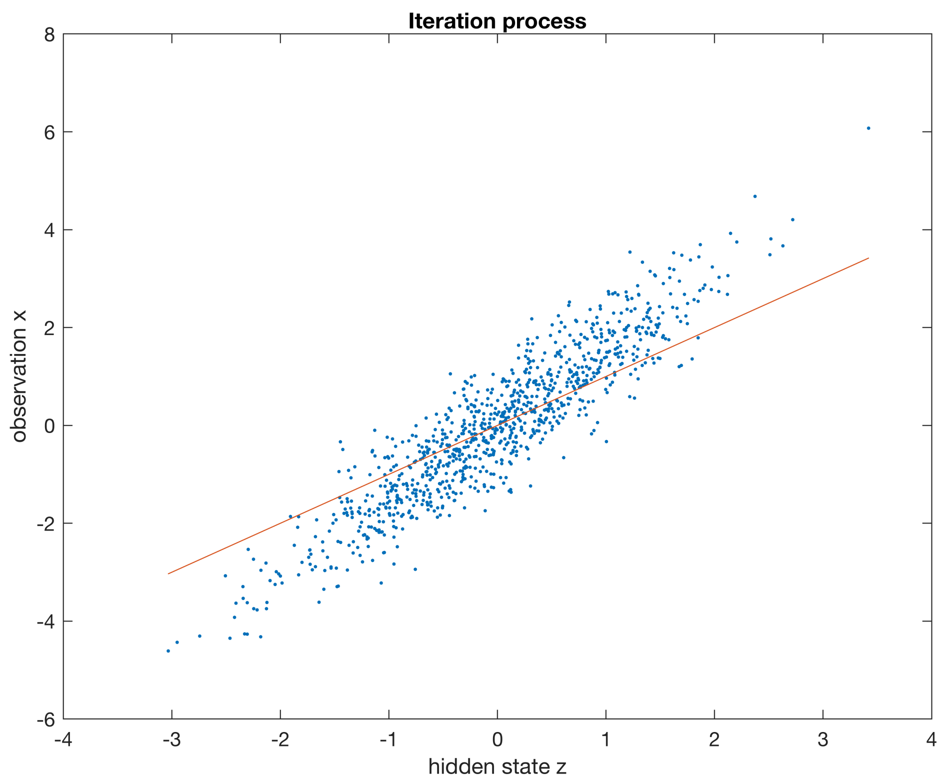



Iteration process
observation x
hidden state z
8
6
4
2
0
-2
-4
-6
-3
-2
-1
0
1
2
3
4

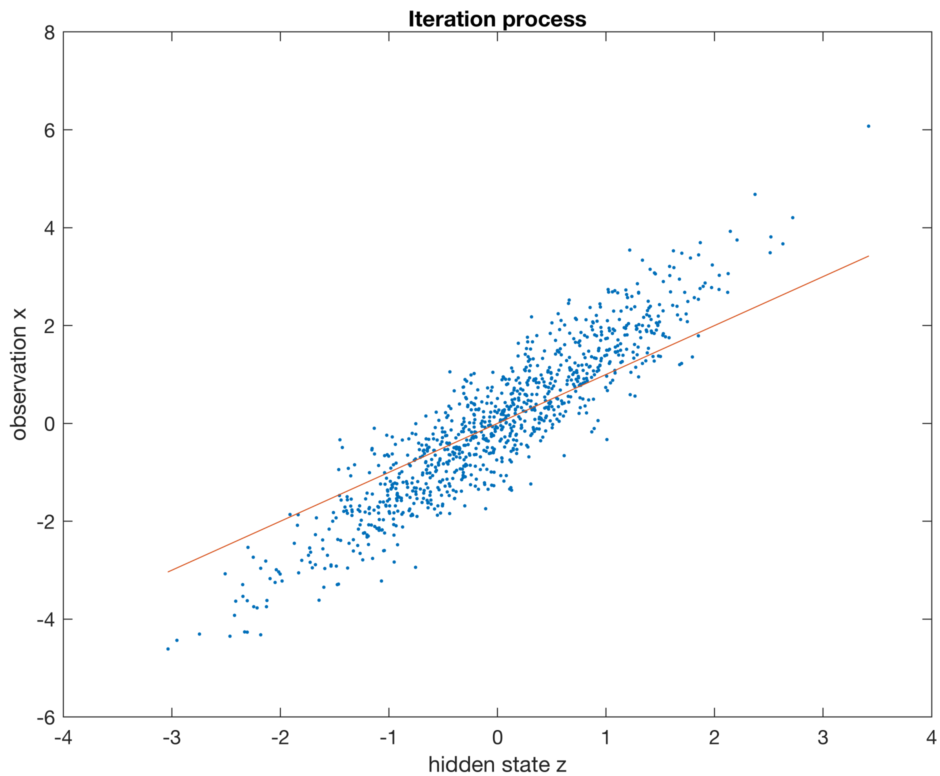

Iteration process
observation x
hidden state z
8
6
4
2
0
-2
-4
-6
-4
-3
-2
-1
0
1
2
3
4

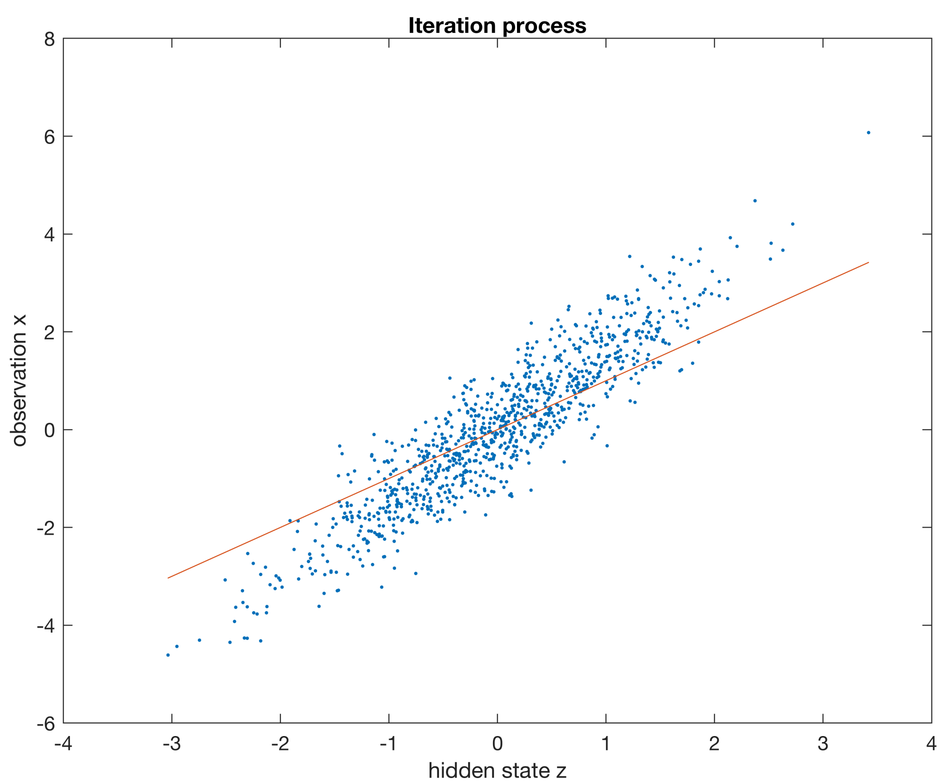


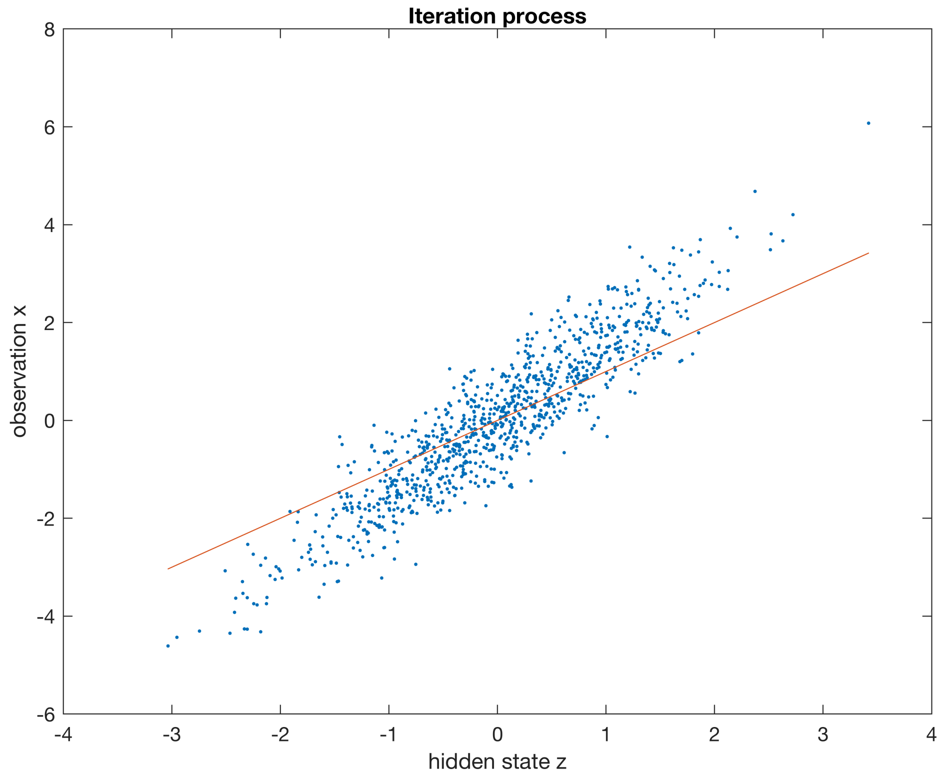
Iteration process
observation x
hidden state z

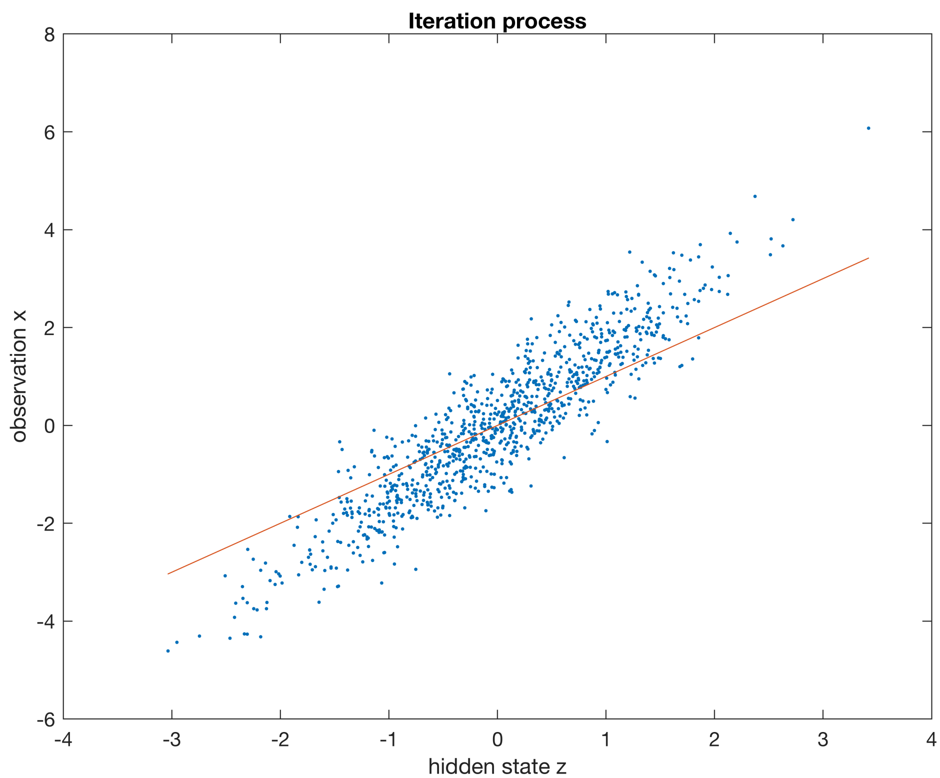


Iteration process
observation x
hidden state z


Iteration process
observation x
hidden state z
8
6
4
2
0
-2
-4
-6
-4
-3
-2
-1
0
1
2
3
4


Iteration process
observation x
hidden state z
8
6
4
2
0
-2
-4
-6
-4
-3
-2
-1
0
1
2
3
4






```
% Compare with the true z
real_error = abs(z-true_z).^2;
```

```
real_MSE = sum(real_error(:)) / num_samples
```

```
real_MSE = 0.5224
```