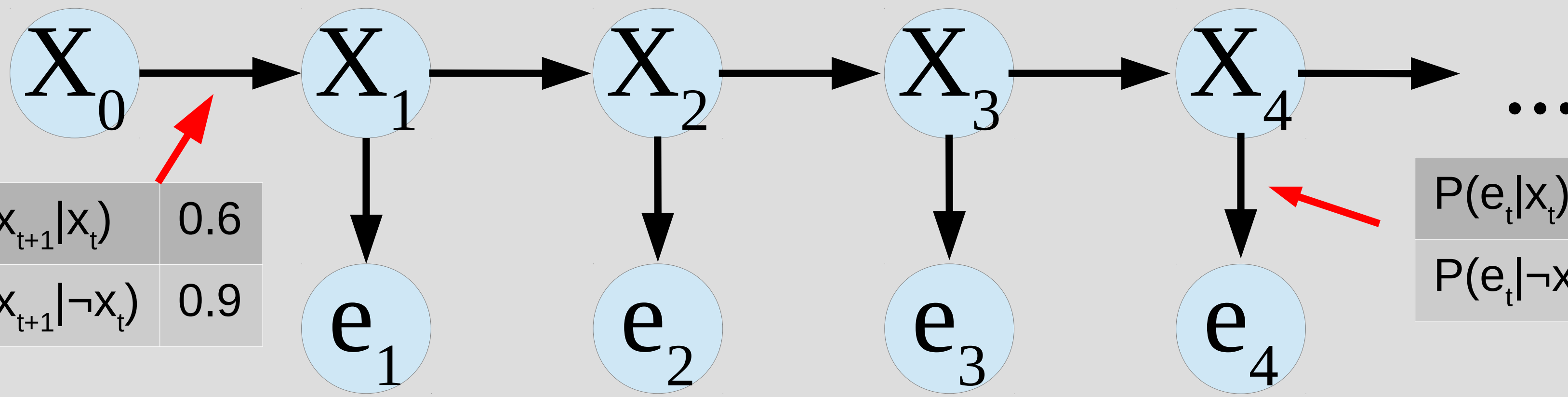
Kalman Filter
(Ch. 15)

HMMs and Matrices



$P(x_{t+1}\|x_t)$	0.6
$P(x_{t+1}\|\neg x_t)$	0.9

$P(e_t\|x_t)$	0.3
$P(e_t\|\neg x_t)$	0.8

We can represent this Bayes net with matrices:

$$P(x_{t+1}|x_t) = T = \begin{bmatrix} 0.6 & 0.4 \\ 0.9 & 0.1 \end{bmatrix}$$

The evidence matrices are more complicated:

$$P(e_t|x_t) = E_t = \begin{bmatrix} 0.3 & 0 \\ 0 & 0.8 \end{bmatrix} \qquad P(\neg e_t|x_t) = E_t = \begin{bmatrix} 0.7 & 0 \\ 0 & 0.2 \end{bmatrix}$$

both just called E_t, which depends on whether e_t or $\neg e_t$

HMMs and Matrices

This allows us to represent our filtering eq:

$$f(t) = \alpha P(e_t|x_t) \sum_{x_{t-1}} \Big(P(x_t|x_{t-1}) f(t-1) \Big)$$

... with matrices:

$$F_{t+1} = \alpha E_{t+1} T^T F_t$$

... why?

(1) Gets rid of sum (matrix mult. does this)

(2) More easily to “reverse” messages

$$F_t = \alpha' (T^T)^{-1} E_{t+1}^{-1} F_{t+1}$$

HMMs and Matrices

This actually gives rise to a smoothing alg. with constant memory (we did with linear):

Smooth (constant mem):
-1. Compute filtering from 1 to t
-2. Loop: i=t to 1
-2.1. Smooth X_i (have f(i) and backwards(i))
-2.2. Compute backwards(i-1) in normal way
-2.3. Compute f(i-1) using previous slide

HMMs and Matrices

Smoothing actually has issues with “online” algorithms, where you need results mid-alg.

The stock market is an example as you have historical info and need choose trades today

But tomorrow we will have the info for today as well... need alg to not compute “from scratch”

HMMs and Matrices

With smoothing, the “forwards” message is fine, since we start it at f(0) and go to f(t)

We can then compute the “next day” easily as f(t+1) is based off f(t) in our equations

This is not the case for the “backwards” message, as this starts b(t) to get b(t-1)

$$b(k) = \sum_{x_{k+1}} P(e_{k+1}|x_{k+1})b(k+1)P(x_{k+1}|x_k)$$

As matrix: $B_k = TE_{k+1}B_{k+1}$

HMMs and Matrices

The naive way would be to restart the "backwards" message from scratch

I will switch to the book's notation of $B_{1:t}$ as the backward message that uses e_1 to e_t (slightly different as B_k uses e_{k+1} to e_t)

Thus we would want some way to compute $B_{j:t+1}$ from $B_{k:t}$ without doing it from scrath

HMMs and Matrices

So we have:

$$B_{1:1} = TE_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad B_{1:2} = TE_1TE_2 \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad B_{2:2} = TE_2 \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

In general: $B_{j:k} = \left(\prod_{i=j}^{k} TE_i \right) \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

This $[1,1]^T$ matrix is in the way, so let's store:

$$\hat{B}_{j:k} = \left(\prod_{i=j}^{k} TE_i \right)$$

... then: $\hat{B}_{2:2} = E_1^{-1} T^{-1} \hat{B}_{1:1} T E_2$

i starts large,
then decreases:
for(i=j-1; i>=k; i--)

... or generally if j>k: $\hat{B}_{j:t+1} = \left(\prod_{i=j-1}^{k} E_i^{-1} T^{-1} \right) \hat{B}_{k:t} T E_{t+1}$

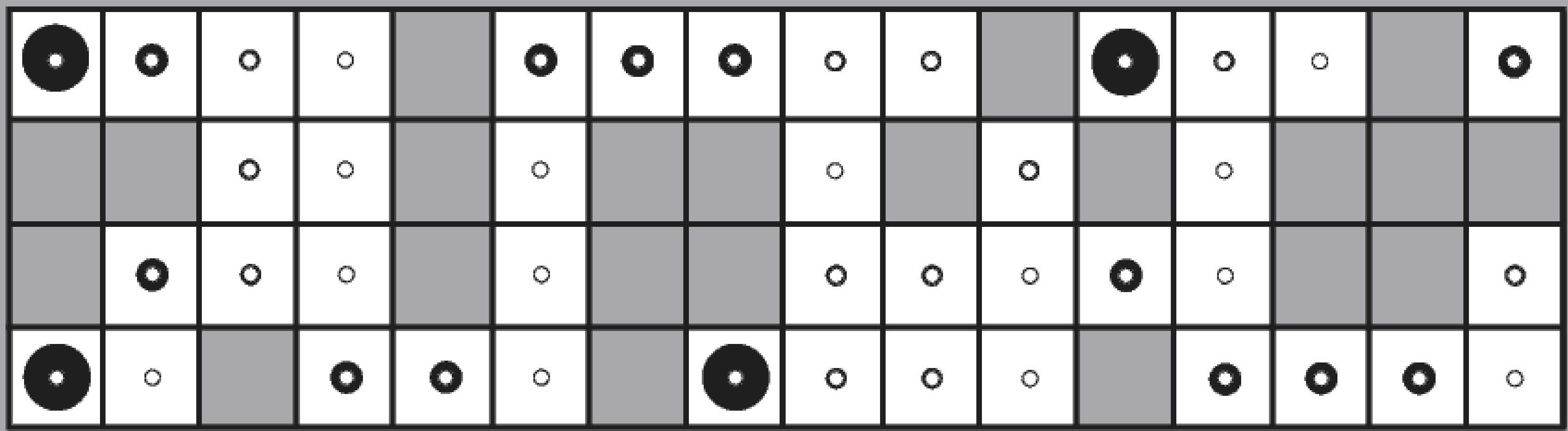
HMMs in Practice

One common place this filtering is used is in position tracking (radar)

The book gives a nice example that is more complex than we have done:

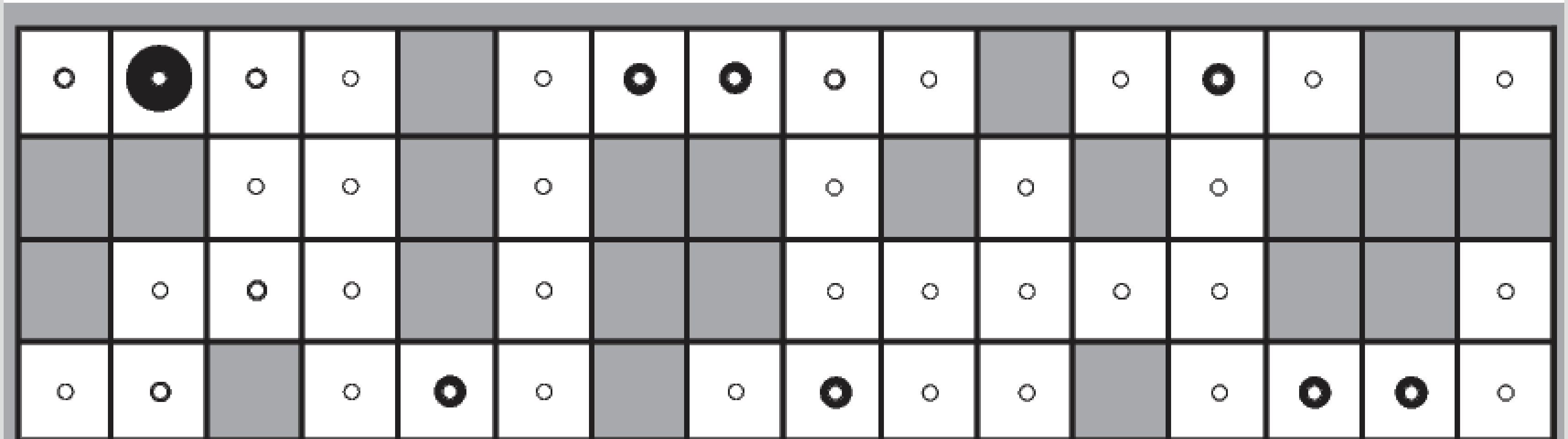
A robot is dropped in a maze (it has a map), but it does not know where...
... additionally, the sensors on the robot does not work well... where is the robot?

HMMs in Practice



(a) Posterior distribution over robot location after $E_1 = NSW$

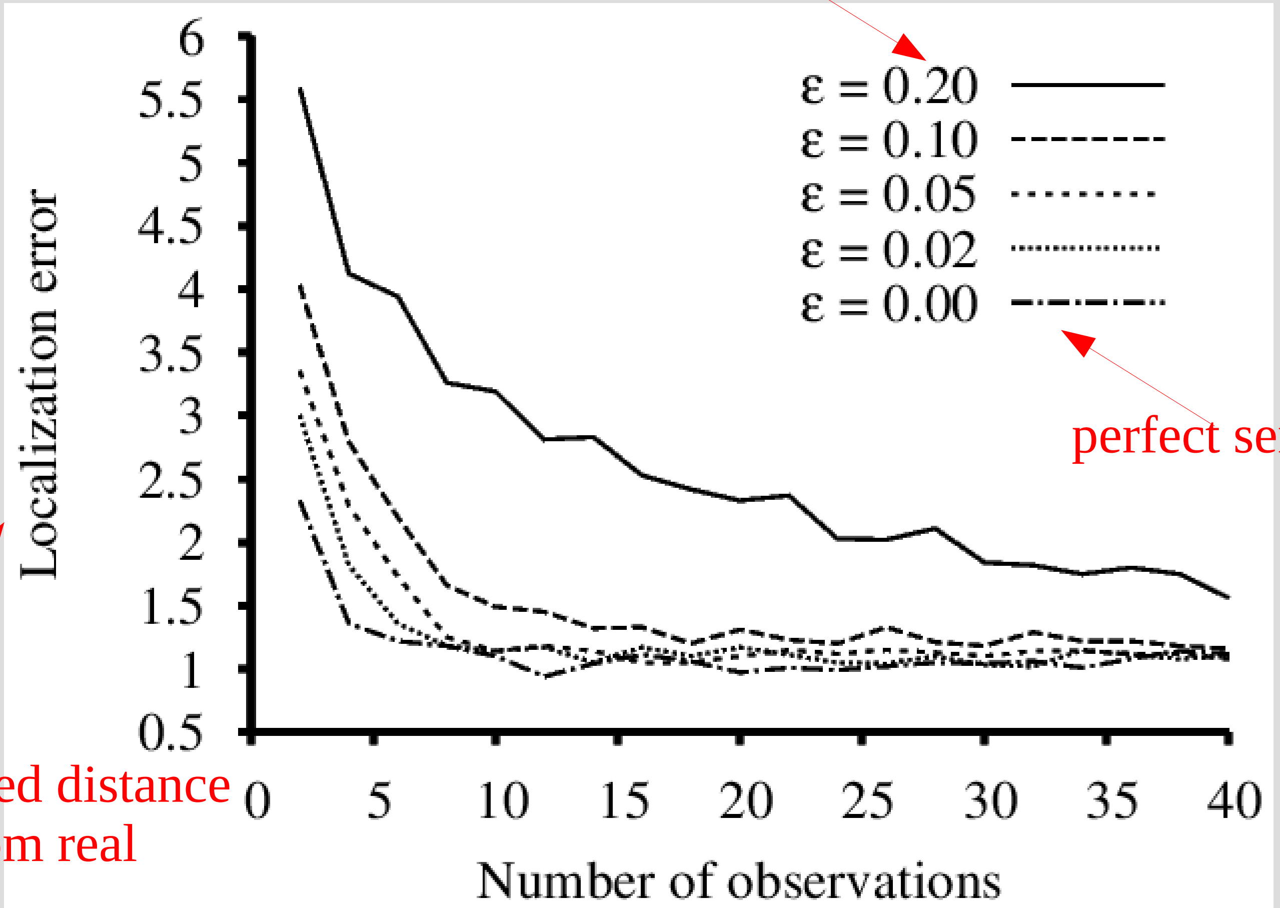
where walls are



(b) Posterior distribution over robot location after $E_1 = NSW, E_2 = NS$

HMMs in Practice

20% error per direction
(1-.8^4) = 59% at least one error



perfect sensors

Average expected distance
(Manhattan) from real

Kalman Filters

How does all of this relate to Kalman filters?

This is just “filtering” (in HMM/Bayes net), except with continuous variables

This heavily use the Gaussian distribution:

$$N(\mu,\sigma^2)(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-1}{2}\frac{(x-\mu)^2}{\sigma^2}} = \alpha e^{\frac{-1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

thank you alpha!

Kalman Filters

Why the preferential treatment for Gaussians?

A key benefit is that when you do our normal operations (add and multiply), if you start with a Gaussian as input, you get Gaussian out

In fact, if you input a linear Gaussian input, you get a Gaussian out: (linear=matrix mult)

More on this later, let's start simple

Kalman Filters

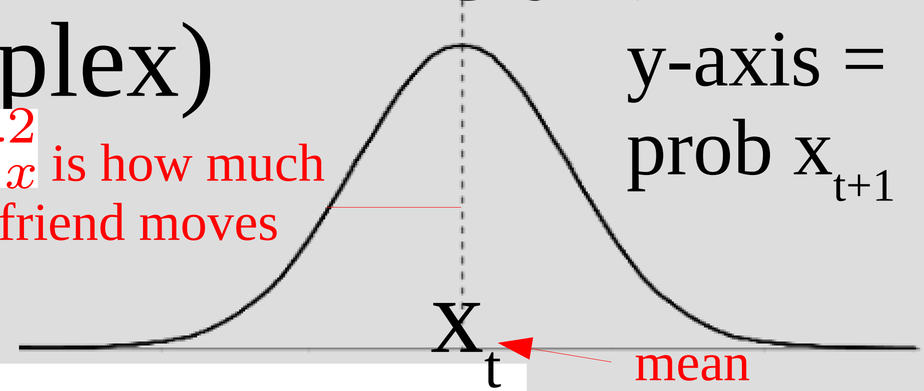
As an example, let's say you are playing Frisbee at night

1. Can't see exactly where friend is

2. Friend will move slightly to catch Frisbee

Kalman Filters

Unfortunately... the math is a bit ugly (as Gaussians are a bit complex)



Here we assume:

$$P(x_{t+1}|x_t) = \alpha e^{\frac{-1}{2}\frac{(x_{t+1}-x_t)^2}{\sigma_x^2}} = N(x_t, \sigma_x^2)(x_{t+1})$$

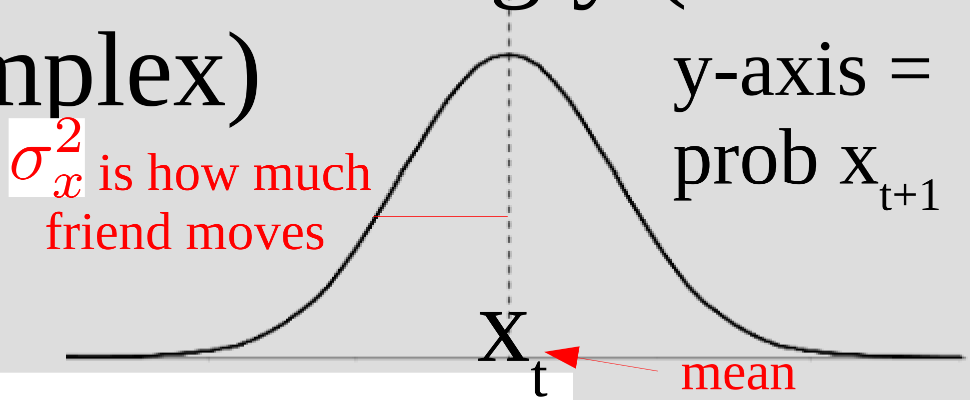
$$P(e_t|x_t) = \alpha e^{\frac{-1}{2}\frac{(e_t-x_t)^2}{\sigma_e^2}} = N(x_t, \sigma_e^2)(e_t)$$

variance is “can’t see well”

How do we compute the filtering “forward” messages (in our efficient non-recursive way)?

Kalman Filters

Unfortunately... the math is a bit ugly (as Gaussians are a bit complex)



Here we assume:

$$P(x_{t+1}|x_t) = \alpha e^{\frac{-1}{2}\frac{(x_{t+1}-x_t)^2}{\sigma_x^2}} = N(x_t, \sigma_x^2)(x_{t+1})$$

$$P(e_t|x_t) = \alpha e^{\frac{-1}{2}\frac{(e_t-x_t)^2}{\sigma_e^2}} = N(x_t, \sigma_e^2)(e_t)$$

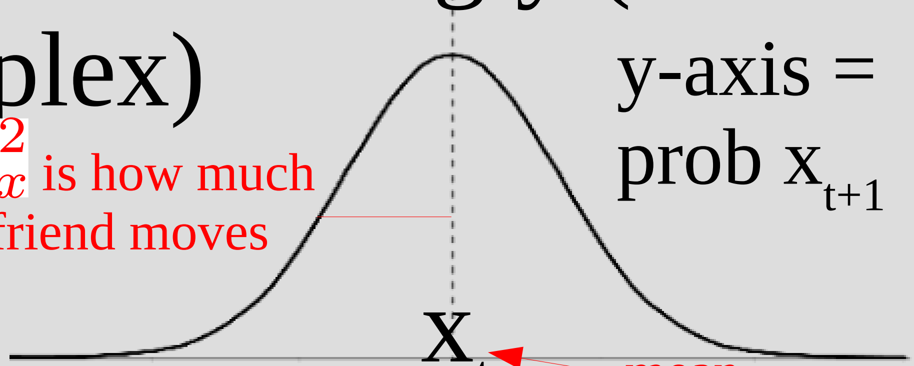
variance is “can’t see well”

erm... let’s change variable names

How do we compute the filtering “forward” messages (in our efficient non-recursive way)?

Kalman Filters

Unfortunately... the math is a bit ugly (as Gaussians are a bit complex)



Here we assume:

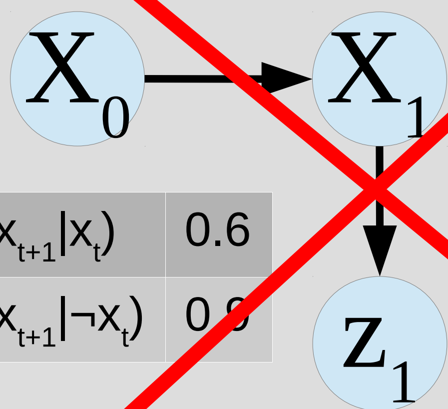
$$P(x_{t+1}|x_t) = \alpha e^{\frac{-1}{2}\frac{(x_{t+1}-x_t)^2}{\sigma_x^2}} = N(x_t, \sigma_x^2)(x_{t+1})$$

$$P(z_t|x_t) = \alpha e^{\frac{-1}{2}\frac{(z_t-x_t)^2}{\sigma_z^2}} = N(x_t, \sigma_z^2)(z_t)$$

variance is “can’t see well”

How do we compute the filtering “forward” messages (in our efficient non-recursive way)?

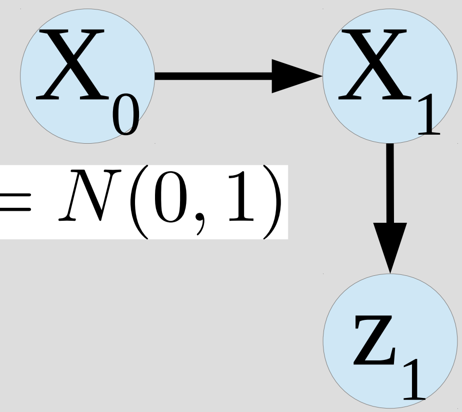
Kalman Filters



$P(x_{t+1}\|x_t)$	0.6
$P(x_{t+1}\|\neg x_t)$	[illegible]

$P(e_t\|x_t)$	0.3
$P(e_t\|\neg x_t)$	0.8

$$P(x_{t+1}|x_t) = N(x_t, \sigma_x^2)(x_{t+1})$$



$$P(x_0) = N(0,1)$$

$$P(z_t|x_t) = N(x_t, \sigma_z^2)(z_t)$$

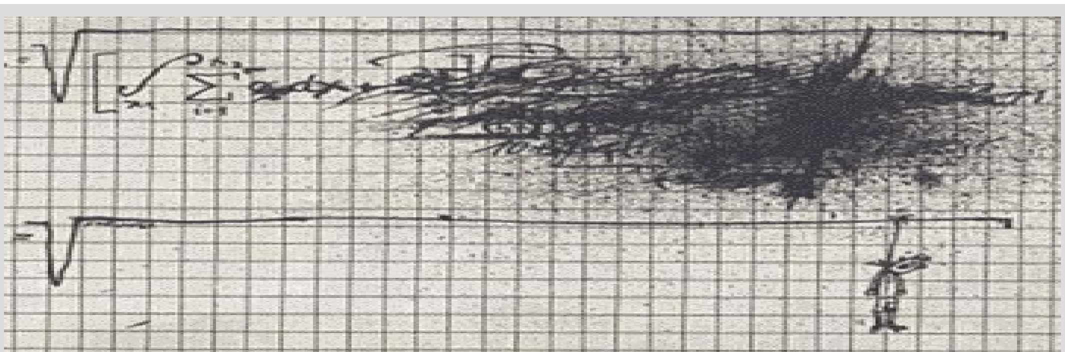
The same? Sorta... but we have to integrate

$$\cancel{F_1 = P(z_1|x_1) \sum_{x_0} P(x_1|x_0)P(x_0)}$$

$$F_1 = P(z_1|x_1) \int_{-\infty}^{\infty} P(x_1|x_0)P(x_0)dx_0$$

Kalman Filters

$$
\begin{aligned}
F_1 &= P(z_1|x_1)\int_{-\infty}^{\infty} P(x_1|x_0)P(x_0)dx_0 \\
&= N(x_1,\sigma_z^2)(z_1)\int_{-\infty}^{\infty} N(x_0,\sigma_x^2)(x_1)\cdot N(0,1)(x_0)dx_0 \\
&= N(x_1,\sigma_z^2)(z_1)\int_{-\infty}^{\infty} \alpha e^{\frac{-1}{2}\frac{(x_1-x_0)^2}{\sigma_x^2}}\cdot \alpha' e^{\frac{-1}{2}x_0^2}dx_0 \\
&= \hat{\alpha}N(x_1,\sigma_z^2)(z_1)\int_{-\infty}^{\infty} e^{\frac{-1}{2}\frac{(x_1-x_0)^2+\sigma_x^2x_0^2}{\sigma_x^2}}dx_0 \\
&= \hat{\alpha}N(x_1,\sigma_z^2)(z_1)\int_{-\infty}^{\infty} e^{\frac{-1}{2}\left(\frac{(1+\sigma_x^2)}{\sigma_x^2}x_0^2+\frac{(-2x_1)}{\sigma_x^2}x_0+\frac{x_1^2}{\sigma_x^2}\right)}dx_0
\end{aligned}
$$



Kalman Filters

But wait! There's hope!

We can use a little fact that:

$$ax^2 + bx + c = a(x - \frac{-b}{2a})^2 + \underbrace{(c - \frac{b^2}{4a})}_{\text{does not contain } x}$$

$$\begin{aligned}
F_1 &= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) \int_{-\infty}^{\infty} e^{\frac{-1}{2}(\frac{(1+\sigma_x^2)}{\sigma_x^2} x_0^2 + \frac{(-2x_1)}{\sigma_x^2} x_0 + \frac{x_1^2}{\sigma_x^2})} \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) \int_{-\infty}^{\infty} e^{\frac{-1}{2}(a(x_0 - \frac{-b}{2a})^2 + (c - \frac{b^2}{4a})} dx_0 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) \int_{-\infty}^{\infty} e^{\frac{-1}{2}(a(x_0 - \frac{-b}{2a})^2} e^{\frac{-1}{2}(c - \frac{b^2}{4a})} dx_0 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}(c - \frac{b^2}{4a})} \int_{-\infty}^{\infty} e^{\frac{-1}{2}(a(x_0 - \frac{-b}{2a})^2} dx_0
\end{aligned}$$

Kalman Filters

But wait! There's hope!

We can use a little fact that:

$$ax^2 + bx + c = a(x - \frac{-b}{2a})^2 + \underbrace{(c - \frac{b^2}{4a})}_{\text{does not contain } x}$$

$$\begin{aligned}
F_1 &= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) \int_{-\infty}^{\infty} e^{\frac{-1}{2}\left(\frac{(1+\sigma_x^2)}{\sigma_x^2} x_0^2 + \frac{(-2x_1)}{\sigma_x^2} x_0 + \frac{x_1^2}{\sigma_x^2}\right)} \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) \int_{-\infty}^{\infty} e^{\frac{-1}{2}(a(x_0 - \frac{-b}{2a})^2 + (c - \frac{b^2}{4a})} dx_0 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) \int_{-\infty}^{\infty} e^{\frac{-1}{2}(a(x_0 - \frac{-b}{2a})^2} e^{\frac{-1}{2}(c - \frac{b^2}{4a})} dx_0 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}(c - \frac{b^2}{4a})} \int_{-\infty}^{\infty} \boxed{e^{\frac{-1}{2}(a(x_0 - \frac{-b}{2a})^2}} dx_0
\end{aligned}$$

This is just:

$$N(\frac{-b}{2a}, \frac{1}{a})$$

Kalman Filters

$$a = \frac{1+\sigma_x^2}{\sigma_x^2}, b = \frac{-2x_1}{\sigma_x^2}, c = \frac{x_1^2}{\sigma_x^2}$$

$$\begin{aligned}
F_1 &= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}(c - \frac{b^2}{4a})} \int_{-\infty}^{\infty} e^{\frac{-1}{2}(a(x_0 - \frac{-b}{2a})^2} dx_0 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}(c - \frac{b^2}{4a})} \int_{-\infty}^{\infty} N(\frac{-b}{2a}, \frac{1}{a})(x_0) dx_0 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}(c - \frac{b^2}{4a})} \cdot 1 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}\left(\frac{x_1^2}{\sigma_x^2} - \frac{(\frac{-2x_1}{\sigma_x^2})^2}{4\frac{1+\sigma_x^2}{\sigma_x^2}}\right)} \cdot 1 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}\left(\frac{x_1^2}{\sigma_x^2} - \frac{4x_1^2/\sigma_x^4}{4(1+\sigma_x^2)/\sigma_x^2}\right)} \cdot 1 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}\left(\frac{x_1^2(1+\sigma_x^2)}{\sigma_x^2(1+\sigma_x^2)} - \frac{x_1^2}{(1+\sigma_x^2)\sigma_x^2}\right)} \cdot 1 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}\left(\frac{x_1^2+\sigma_x^2 x_1^2 - x_1^2}{(1+\sigma_x^2)\sigma_x^2}\right)} \cdot 1 \\
&= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}\left(\frac{x_1^2}{1+\sigma_x^2}\right)} \cdot 1
\end{aligned}$$

area under all of normal distribution adds up to 1

Kalman Filters

$$
\begin{aligned}
F_1 &= \hat{\alpha} N(x_1, \sigma_z^2)(z_1) e^{\frac{-1}{2}(\frac{x_1^2}{1+\sigma_x^2})} \\
&= \hat{\alpha} e^{\frac{-1}{2}(\frac{(z_1-x_1)^2}{\sigma_z^2})} e^{\frac{-1}{2}(\frac{x_1^2}{1+\sigma_x^2})} \\
&= \hat{\alpha} e^{\frac{-1}{2}(\frac{(z_1-x_1)^2}{\sigma_z^2}+\frac{x_1^2}{1+\sigma_x^2})} \\
&= \hat{\alpha} e^{\frac{-1}{2}(\frac{(z_1-x_1)^2(1+\sigma_x^2)+\sigma_z^2 x_1^2}{\sigma_z^2(1+\sigma_x^2)})} \\
&= \hat{\alpha} e^{\frac{-1}{2}\left((\frac{1+\sigma_x^2+\sigma_z^2}{\sigma_z^2(1+\sigma_x^2)})x_1^2+(-2z_1/\sigma_z^2)x_1+z_1^2/\sigma_z^2\right)} \\
&= \hat{\alpha} e^{\frac{-1}{2}a(x_1-\frac{-b}{2a})^2+(c-\frac{b^2}{4a})} \\
&= \hat{\alpha} e^{\frac{-1}{2}(c-\frac{b^2}{4a})} e^{\frac{-1}{2}a(x_1-\frac{-b}{2a})^2} \\
&= \hat{\alpha}' e^{\frac{-1}{2}a(x_1-\frac{-b}{2a})^2}
\end{aligned}
$$

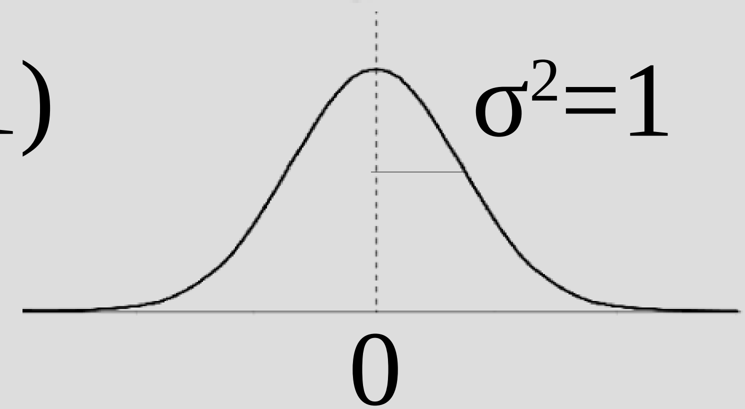
$$ax^2 + bx + c = a(x - \frac{-b}{2a})^2 + (c - \frac{b^2}{4a})$$

gross after plugging in a,b,c (see book)

Kalman: Frisbee in the Dark

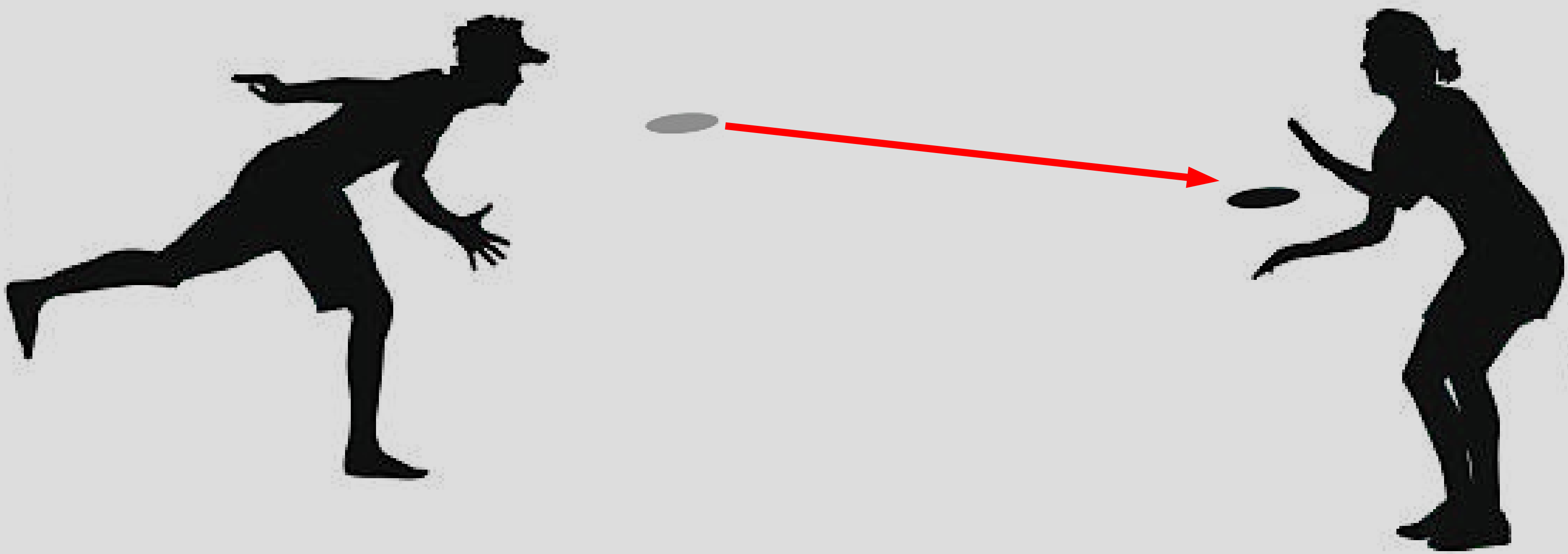
$$a = \frac{1+\sigma_x^2+\sigma_z^2}{\sigma_z^2(1+\sigma_x^2)}, b = -2z_1/\sigma_z^2, c = z_1^2/\sigma_z^2$$

Initially your friend is N(0,1)



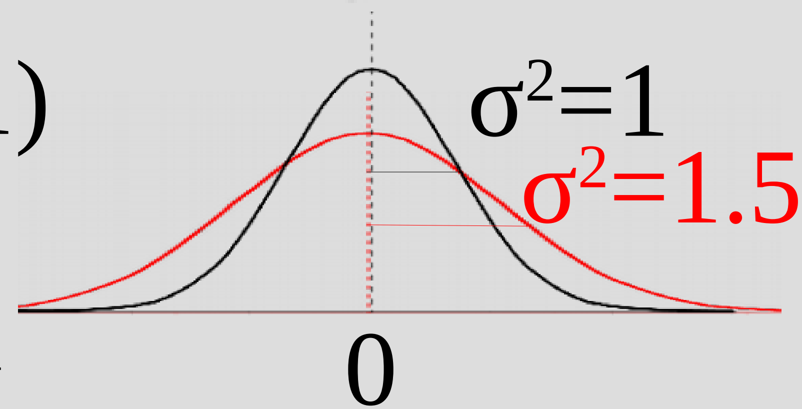
Kalman: Frisbee in the Dark

$$a = \frac{1+\sigma_x^2+\sigma_z^2}{\sigma_z^2(1+\sigma_x^2)}, b = -2z_1/\sigma_z^2, c = z_1^2/\sigma_z^2$$



Initially your friend is N(0,1)

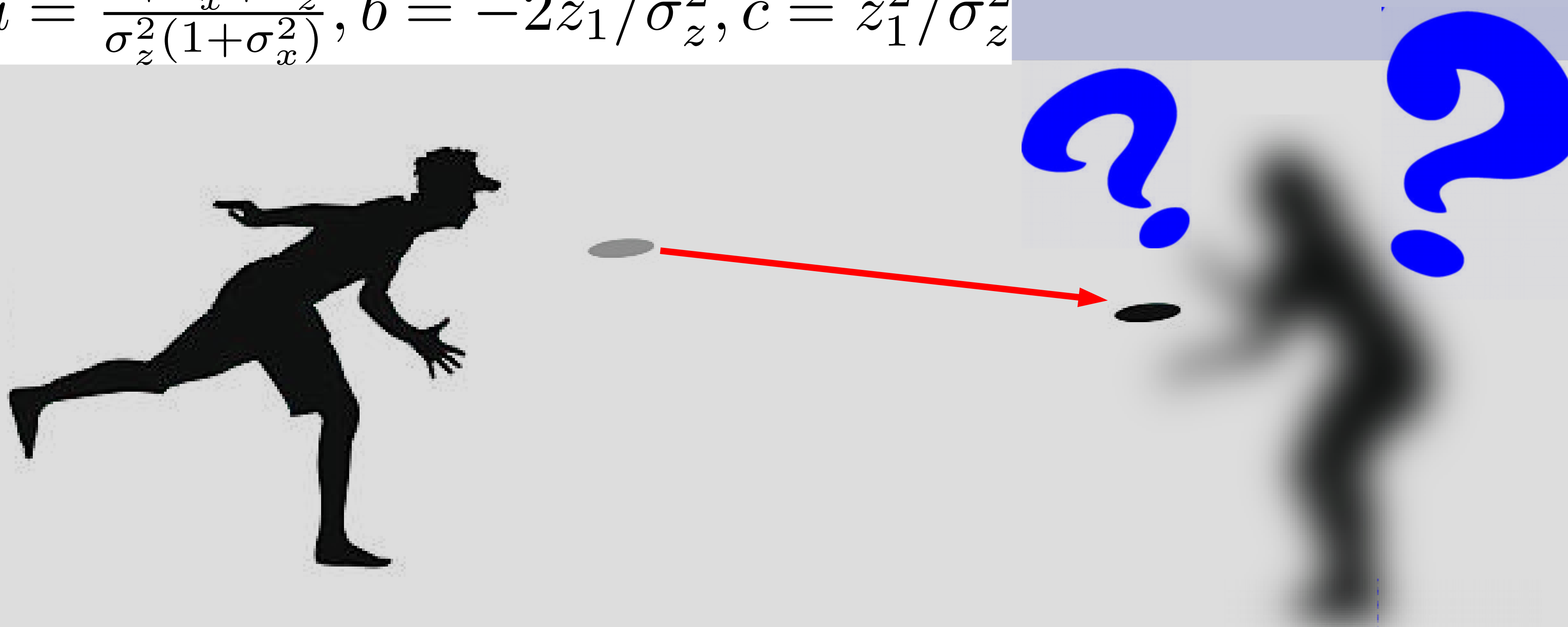
Throw not perfect, so friend has to move N(0,1.5)



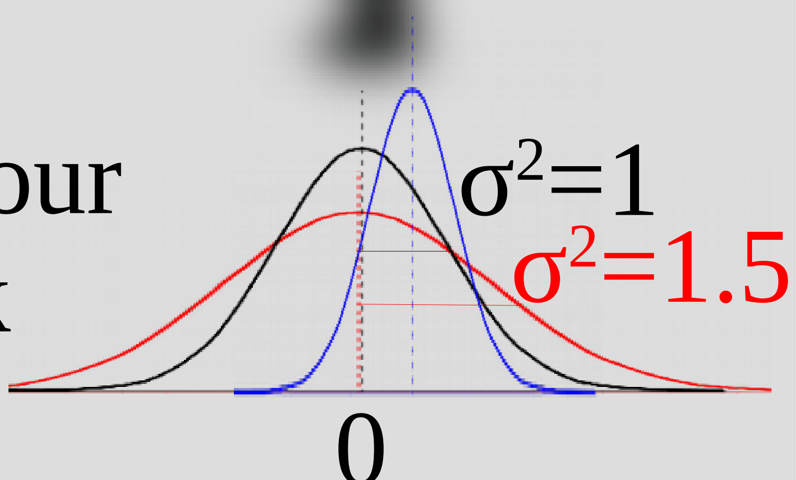
(i.e. move from black to red)

Kalman: Frisbee in the Dark

$$a = \frac{1+\sigma_x^2+\sigma_z^2}{\sigma_z^2(1+\sigma_x^2)}, b = -2z_1/\sigma_z^2, c = z_1^2/\sigma_z^2$$



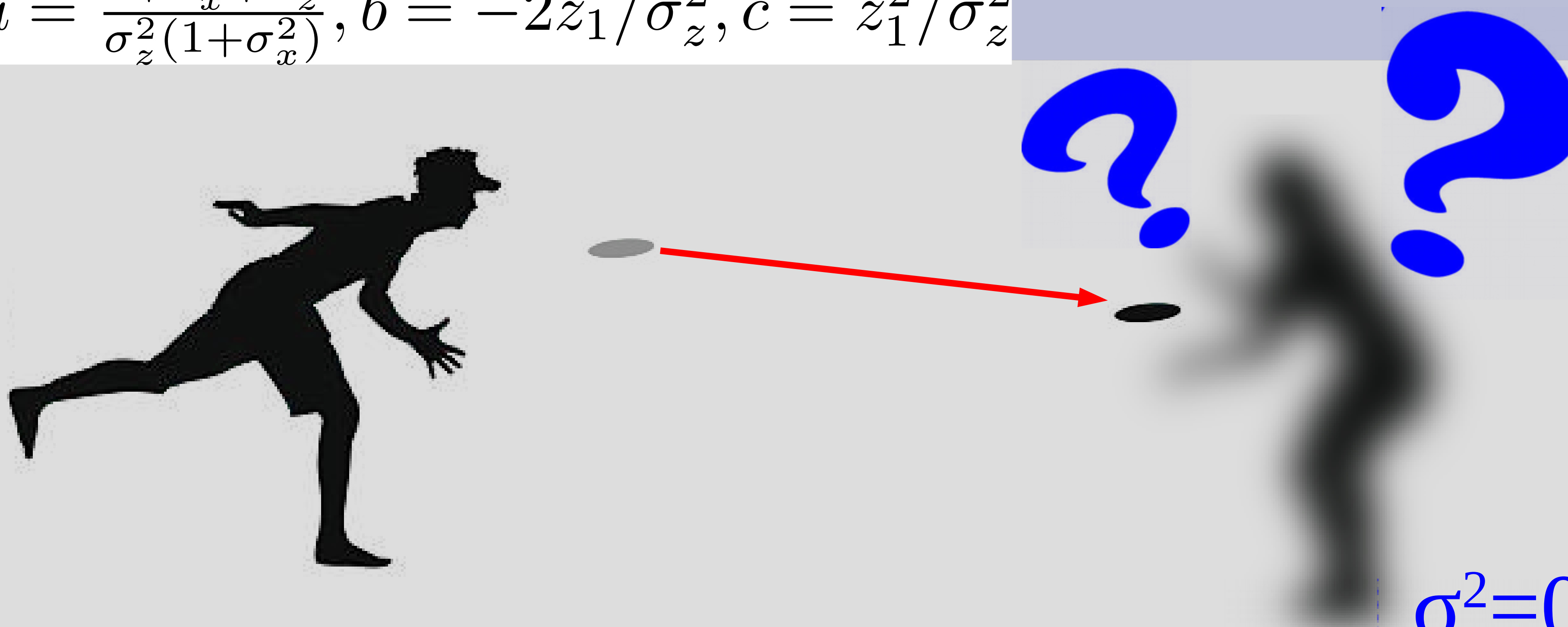
But you can’t actually see your friend too clearly in the dark



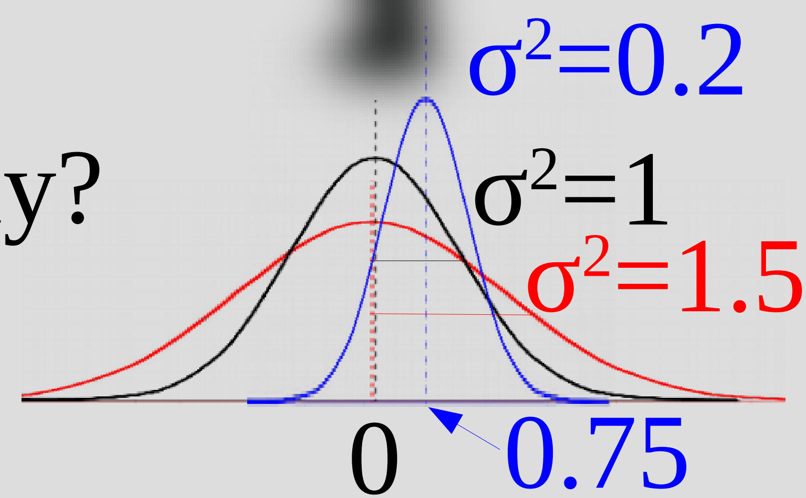
You thought you saw them at 0.75 (σ^2=0.2)

Kalman: Frisbee in the Dark

$$a = \frac{1+\sigma_x^2+\sigma_z^2}{\sigma_z^2(1+\sigma_x^2)}, b = -2z_1/\sigma_z^2, c = z_1^2/\sigma_z^2$$



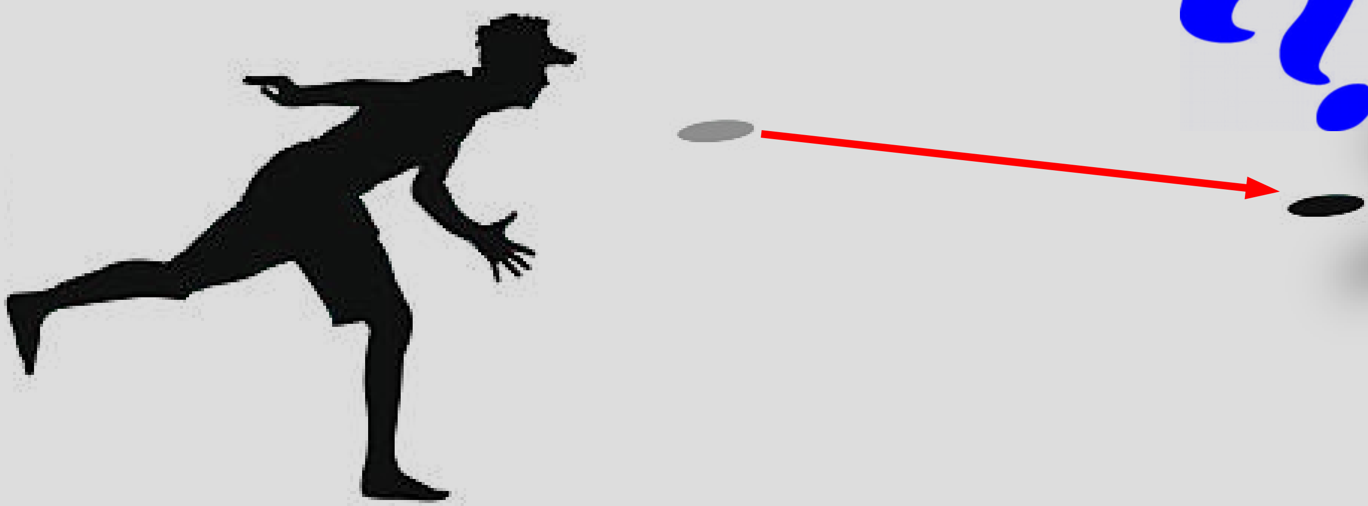
Where is your friend actually?

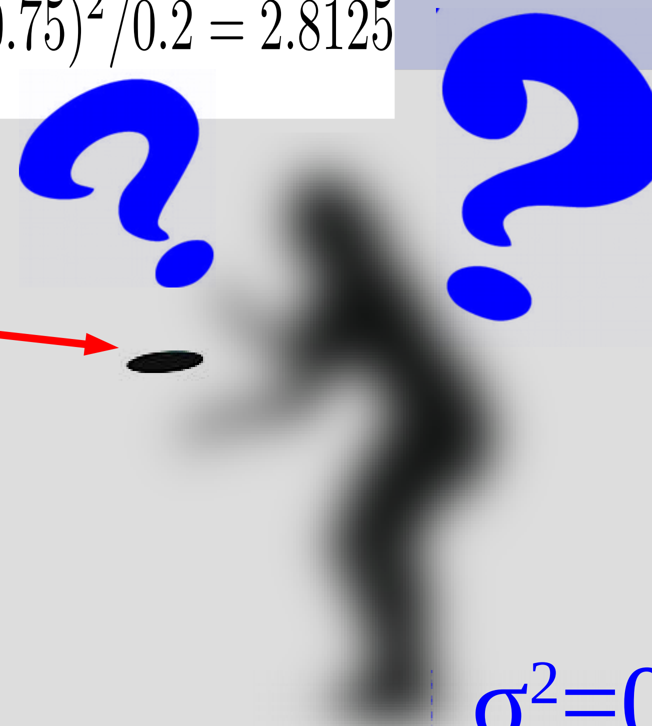


$$F_1 = \hat{\alpha}' e^{\frac{-1}{2} a (x_1 - \frac{-b}{2a})^2}$$

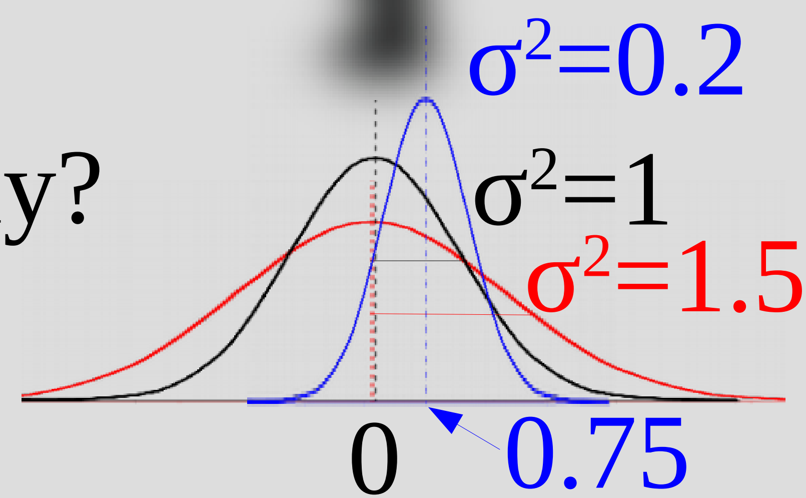
Kalman: Frisbee in the Dark

$$a = \frac{1+1.5+0.2}{0.2(1+1.5)} = 5.4, b = -2(0.75)/0.2 = -7.5, c = (0.75)^2/0.2 = 2.8125$$





Where is your friend actually?



$$\begin{aligned} F_1 &= \hat{\alpha}' e^{\frac{-1}{2}a(x_1 - \frac{-b}{2a})^2} \\ &= N(\frac{-b}{2a}, \frac{1}{a}) \\ &= N(0.694, 0.185) \end{aligned}$$

Probably 0.05

“left” of where you “saw” them

Kalman Filters

So the filtered “forward” message for throw 1 is: $N(0.694, 0.185)$

To find the filtered “forward” message for throw 2, use $N(0.694, 0.185)$ instead of $N(0, 1)$ (this does change the equations as you need to involve a μ for the old $N(0, 1)$)

The book gives you the full messy equations:

$$\mu_{t+1} = \frac{(\sigma_t^2 + \sigma_x^2) z_{t+1} + \sigma_z^2 \mu_t}{\sigma_t^2 + \sigma_x^2 + \sigma_z^2} \qquad \sigma_{t+1}^2 = \frac{(\sigma_t^2 + \sigma_x^2) \sigma_z^2}{\sigma_t^2 + \sigma_x^2 + \sigma_z^2}$$

Kalman Filters

So the filtered "forward" message for throw 1 is: $N(0.694, 0.185)$

To find the filtered "forward" message for throw 2, use $N(\boxed{0.694,}\,0.185)$ instead of $N(0, 1)$ (this does change the equations as you need to involve a μ for the old $N(\boxed{0,}\,1)$)

The book gives you the full messy equations:

$$\boxed{\mu_{t+1}} = \frac{(\sigma_t^2+\sigma_x^2)z_{t+1}+\sigma_z^2\boxed{\mu_t}}{\sigma_t^2+\sigma_x^2+\sigma_z^2} \qquad \sigma_{t+1}^2 = \frac{(\sigma_t^2+\sigma_x^2)\sigma_z^2}{\sigma_t^2+\sigma_x^2+\sigma_z^2}$$

Kalman Filters

So the filtered “forward” message for throw 1 is: $N(0.694, 0.185)$

To find the filtered “forward” message for throw 2, use $N(0.694, \boxed{0.185})$ instead of $N(0, 1)$ (this does change the equations as you need to involve a μ for the old $N(0, \boxed{1})$)

The book gives you the full messy equations:

$$\mu_{t+1} = \frac{(\boxed{\sigma_t^2} + \sigma_x^2) z_{t+1} + \sigma_z^2 \mu_t}{\sigma_t^2 + \sigma_x^2 + \sigma_z^2} \qquad \boxed{\sigma_{t+1}^2} = \frac{(\boxed{\sigma_t^2} + \sigma_x^2) \sigma_z^2}{\sigma_t^2 + \sigma_x^2 + \sigma_z^2}$$

Kalman Filters

The full Kalman filter is done with multiple numbers (matrices)

covariance matrix

Here a Gaussian is: $N(\mu,\Sigma) = \alpha e^{-\frac{1}{2}\left((x-\mu)^T\Sigma^{-1}(x-\mu)\right)}$

Bayes net is: (F and H are “linear” matrix)

$$P(x_{t+1}|x_t) = N(Fx_t,\Sigma_x)(x_{t+1})$$

$$P(z_t|x_t) = N(Hx_t,\Sigma_z)(z_t)$$

identity matrix

Then filter update is: $\Sigma_{t+1} = (I - K_{t+1}H)(F\Sigma_tF^T + \Sigma_x)$

$$\mu_{t+1} = F\mu_t + K_{t+1}(z_{t+1} - HF\mu_t)$$

yikes...

$$K_{t+1} = (F\Sigma_tF^T + \Sigma_x)H^T(H(F\Sigma_tF^T + \Sigma_x)H^T + \Sigma_z)^{-1}$$

Kalman Filters

Often we use $\begin{bmatrix} x \\ v_x \end{bmatrix}$ for a 1-dimensional problem with both position and velocity

To update x_{t+1}, we would want: $x_{t+1} = x_t + v_x$

In matrix form:

$$P(x_{t+1}|x_t) = N(Fx_t, \Sigma_x)(x_{t+1})$$

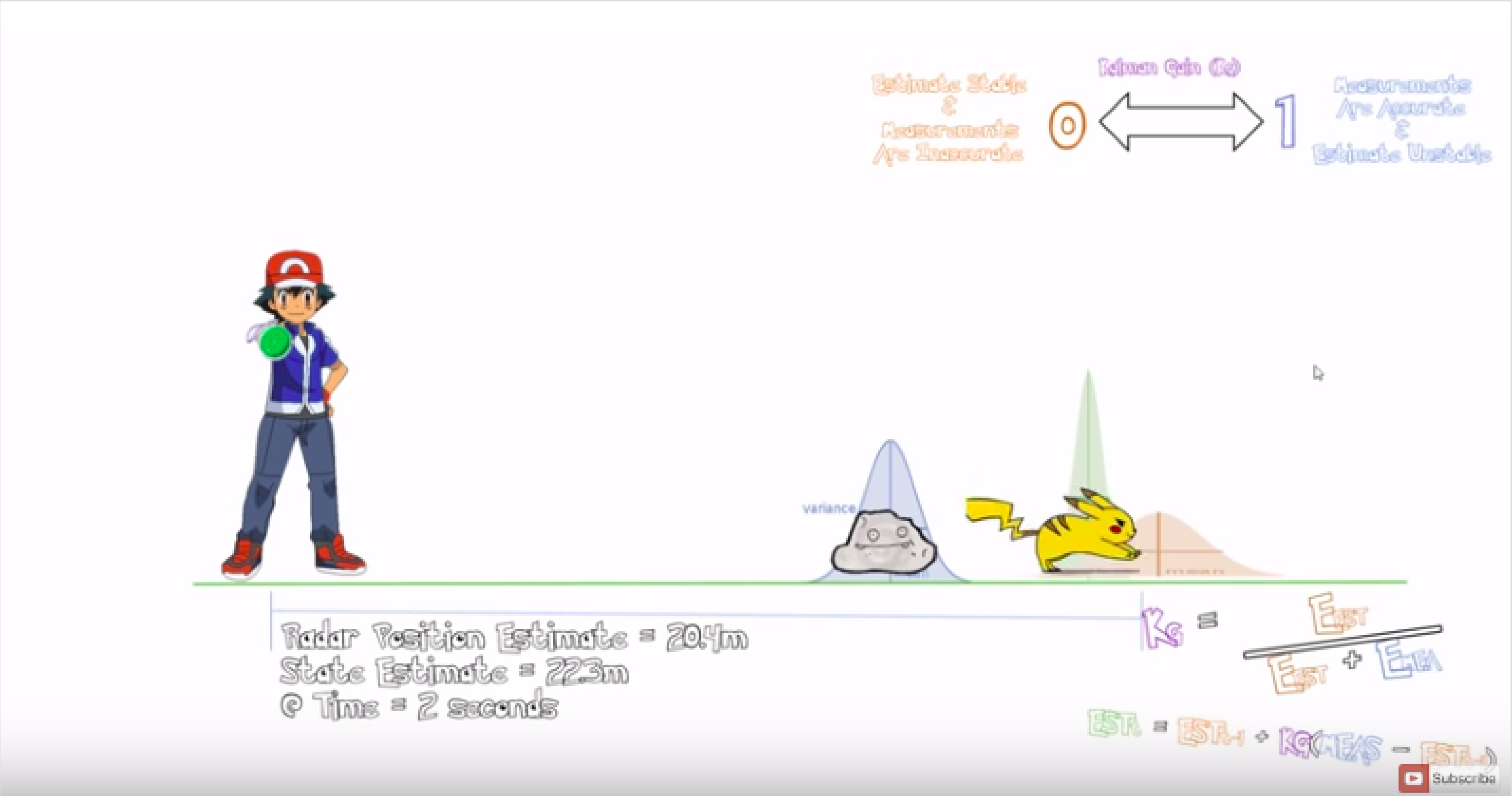
$$F = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \quad \text{so:} \quad Fx_t = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ v_x \end{bmatrix} = \begin{bmatrix} x + v_x \\ v_x \end{bmatrix}$$

So our "mean" at t+1 is [our position at $x+v_x$]

Kalman Filters

Here’s a Pokemon example (not technical)

https://www.youtube.com/watch?v=bm3cwEP2nUo



Kalman Filters

Downsides?

In order to get “simple” equations, we are limited to the linear Gaussian assumption

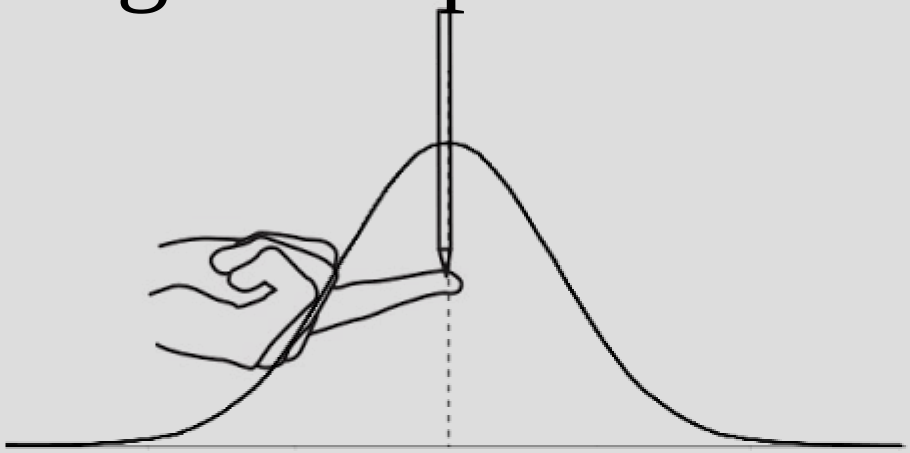
However, there are some cases when this assumption does not work very well at all

Kalman Filters

Consider the example of balancing a pencil on your finger

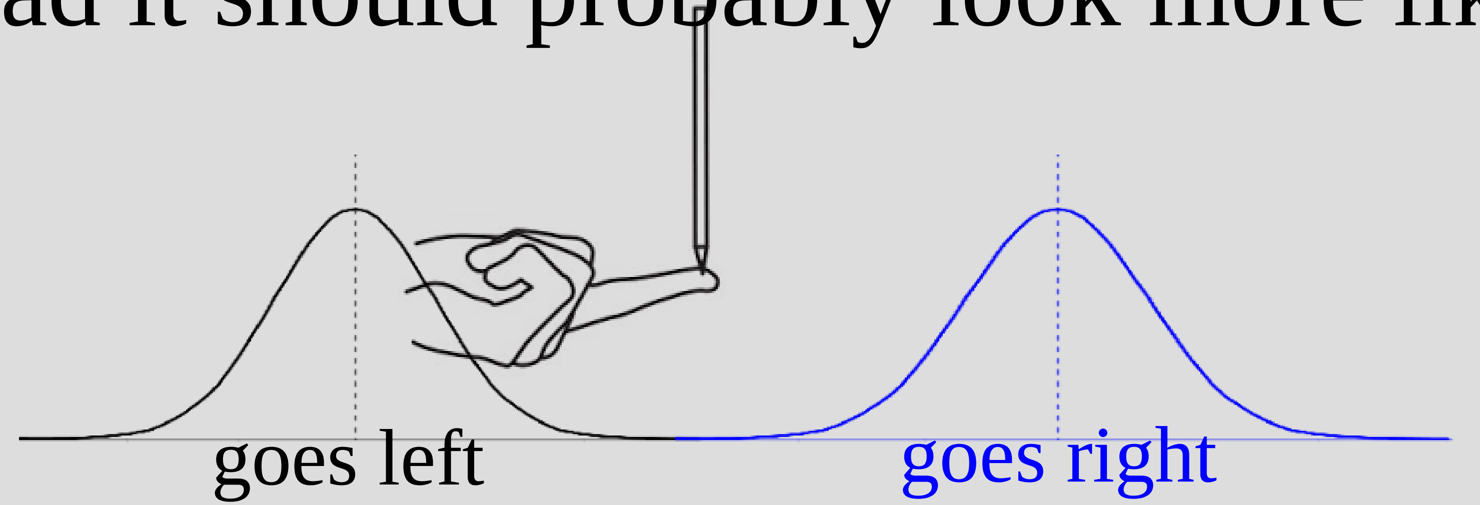
How far to the left/right will the pencil fall?

Below is not a good representation:



Kalman Filters

Instead it should probably look more like:



... where you are deciding between two options, but you are not sure which one

The Kalman filter can handle this as well (just keep 2 sets of equations and use more likely)

Kalman Filters

Unfortunately if you repeat this “pencil balance” on the new spot... you would need 4 sets of equations

3rd attempt: 8 equations
4th attempt: 16 equations
... this exponential amount of work/memory cannot be done for a large HMM