State Estimation for Mobile Robotics

Michal Reinštein

Czech Technical University in Prague
Faculty of Electrical Engineering, Department of Cybernetics
Center for Machine Perception
`http://cmp.felk.cvut.cz/~reinsmic,`
`reinstein.michal@fel.cvut.cz`

Acknowledgement: P. Newman, SLAM Summer School 2006, Oxford

Outline of the lecture:

- Probability rules & Bayes Theorem
- MLE, MAP, MMSE, RBE, LSQ
- Linear Kalman Filter (LKF)
- Example: Linear navigation problem
- Extended Kalman Filter (EKF)
- Introduction to EKF-SLAM

References

1 Paul Newman, EKF Based Navigation and SLAM, SLAM Summer School 2006, http://www.robots.ox.ac.uk/ SSS06/Website/index.htm, University of Oxford

2 Sebastian Thrun, Wolfram Burgard, and Dieter Fox. Probabilistic robotics. MIT press, 2005.

3 Grewal, Mohinder S., and Angus P. Andrews. Kalman filtering: theory and practice using MATLAB. John Wiley & Sons, 2011.

What is Estimation?

„Estimation is the process by which we infer the value of a quantity of interest, x, by processing data that is in some way dependent on x."

- Measured data corrupted by noise—uncertainty in input transformed into uncertainty in inference (e.g. Bayes rule)
- Quantity of interest not measured directly (e.g. odometry in skid-steer robots)
- Incorporating prior (expected) information (e.g. best guess or past experience)
- Open-loop prediction (e.g. knowing current heading and speed, infer future position)
- Uncertainty due to simplifications of analytical models (e.g. performance reasons—linearization)

Bayes Theorem & Probability Rules

- The Product rule: $P(A, B) = P(A|B)\,P(B) = P(B|A)\,P(A)$
- The Sum rule: $P(B) = \sum_A P(A, B) = \sum_A P(B|A)\,P(A)$
- Random events A, B are independent $\Leftrightarrow P(A, B) = P(A)\,P(B)$,
- and the independence means: $P(A|B) = P(A)$, $P(B|A) = P(B)$
- A, B are conditionally independent $\Leftrightarrow P(A, B|C) = P(A|C)P(B|C)$
- The Bayes theorem:

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)} = \frac{P(B|A)P(A)}{\sum_A P(B|A)\,P(A)}$$

Bayes Theorem (1)

In Urban Search & Rescue (USAR), the ability of robots to reliably detect presence of a victim is crucial. How do we implement and evaluate this ability?

Example - Victim detection (1)

Assume we have a sensor S (e.g. a camera) and a computer vision algorithm that detects victims. We evaluated the sensor on ground truth data statistically:

- There is 20% chance of false negative detection (missed target).
- There is 10% chance of false positive detection.
- A priori probability of the victim presence V is 60%.
- *What is the probability that there is a victim if the sensor says no victim is detected?*

Bayes Theorem (2)

We express the sensor S measurements as a conditional probability of V:

$P(S\|V)$	$S = True$	$S = False$
$V = True$	0.8	0.2
$V = False$	0.1	0.9

Express the a priori knowledge as the probability:
$P(V = True) = 0.6$ and $P(V = False) = 1 - 0.6 = 0.4$

Express what-we-want: $P(V|S) =?$ given $S = False$ (not detecting a victim) and $V = True$ (but there is one).

Bayes Theorem (3)

- Use the tools to express what-we-want in the terms of what-we-know:

$$P(V|S) = \frac{P(V,S)}{P(S)} = \frac{P(S|V)P(V)}{\sum_V P(S,V)} = \frac{P(S|V)P(V)}{\sum_V P(S|V)\,P(V)}$$

- Substitute $S = False$ and $V = True$ and sum over V to obtain:

$$P(V|S) = \frac{P(S = False|V = True)P(V = True)}{\sum_V P(S = False|V = True)\,P(V = True)} =$$

$$= \frac{0.2 \cdot 0.6}{0.2 \cdot 0.6 + 0.9 \cdot 0.4} = 0.25$$

- **Conclusion**: if our sensors says there is no victim, we have **25%** chance of missing out someone! We need an additional sensor . . .

Bayes Theorem (4)

In Urban Search & Rescue (USAR), the reliability is achieved through the sensor fusion: use the statistics to evaluate sensors and the probability theory to perform fusion.

Example - Victim detection (2)

Assume we have a sensor S as in the previous case and we add one more sensor T with the following properties:

- There is 5% chance of false negative detection (missed target).
- There is 5% chance of false positive detection.
- A priori probability of the victim presence is the same, V is 60%.
- *What is the probability that there is a victim if both sensors confirm its presence?*

Bayes Theorem (5)

We express the sensor T measurements as a conditional probability of V:

$P(T\|V)$	$T = True$	$T = False$
$V = True$	0.95	0.05
$V = False$	0.05	0.95

The a priori probability is the same:
$P(V = True) = 0.6$ and $P(V = False) = 1 - 0.6 = 0.4$

Express what-we-want: $P(V|S, T) =?$ given $S = True, T = True$ (both sensors see a victim) and $V = True$ (and there is one). Furthermore, we know that both sensors provide independent measurements with respect to each other.

Bayes Theorem (6)

- Naive approach using joint probability: $P(S,T,V) = P(S,T|V)P(V)$
- Conditional independence: $P(S,T|V)P(V) = P(S|V)P(T|V)P(V)$
- Applying the tools:

$$P(V|S,T) = \frac{P(V,S,T)}{P(S,T)} = \frac{P(S|V)P(T|V)P(V)}{\sum_V P(V,S,T)} =$$

$$= \frac{P(S|V)P(T|V)P(V)}{\sum_V P(S|V)P(T|V)\,P(V)}$$

- Substitute: $S = True, T = True$, $V = True$ and sum over V to obtain:

$$= \frac{0.8 \cdot 0.95 \cdot 0.6}{0.8 \cdot 0.95 \cdot 0.6 + 0.1 \cdot 0.05 \cdot 0.4} = 0.9956$$

- **Conclusion**: if both sensors confirm there is a victim, we have **99.56%** chance that there is a victim.

Mean & Covariance

Expectation $=$ the average of a variable under the probability distribution.

Continuous definition: $E(x) = \int_{-\infty}^{\infty} x\, f(x)\, \mathrm{d}x$ vs. discrete: $E(x) = \sum_x x\, P(x)$

Mutual covariance σ_{xy} of two random variables X, Y is

$$\sigma_{xy} = E\left((X - \mu_x)(Y - \mu_y)\right)$$

Covariance matrix[1] Σ of n variables $X_1, \ldots, X_n$ is

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \ldots & \sigma_{1n}^2 \\ & \ddots & \\ \sigma_{n1}^2 & \ldots & \sigma_n^2 \end{bmatrix}$$

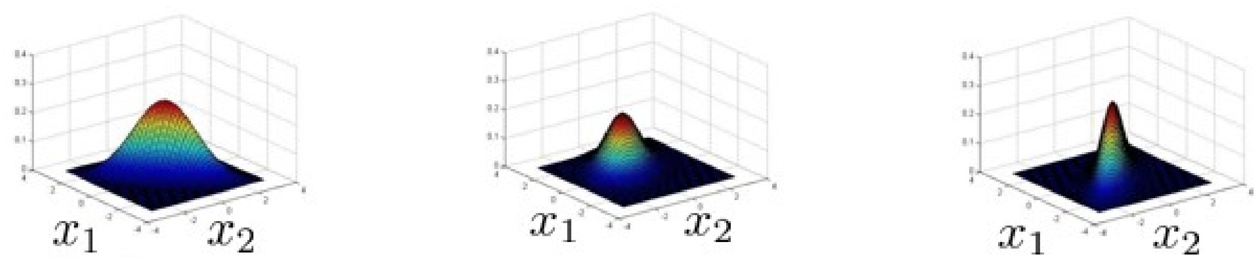
[1]Note: The covariance matrix is symmetric (i.e. $\Sigma = \Sigma^\top$) and positive-semidefinite (as the covariance matrix is real valued, the positive-semidefinite means that $x^\top M x \geq 0$ for all $x \in \mathbb{R}$).

Multivariate Normal distribution (1)

Multivariate Gaussian (Normal) distribution

Parameters μ, Σ

$$p(x;\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$



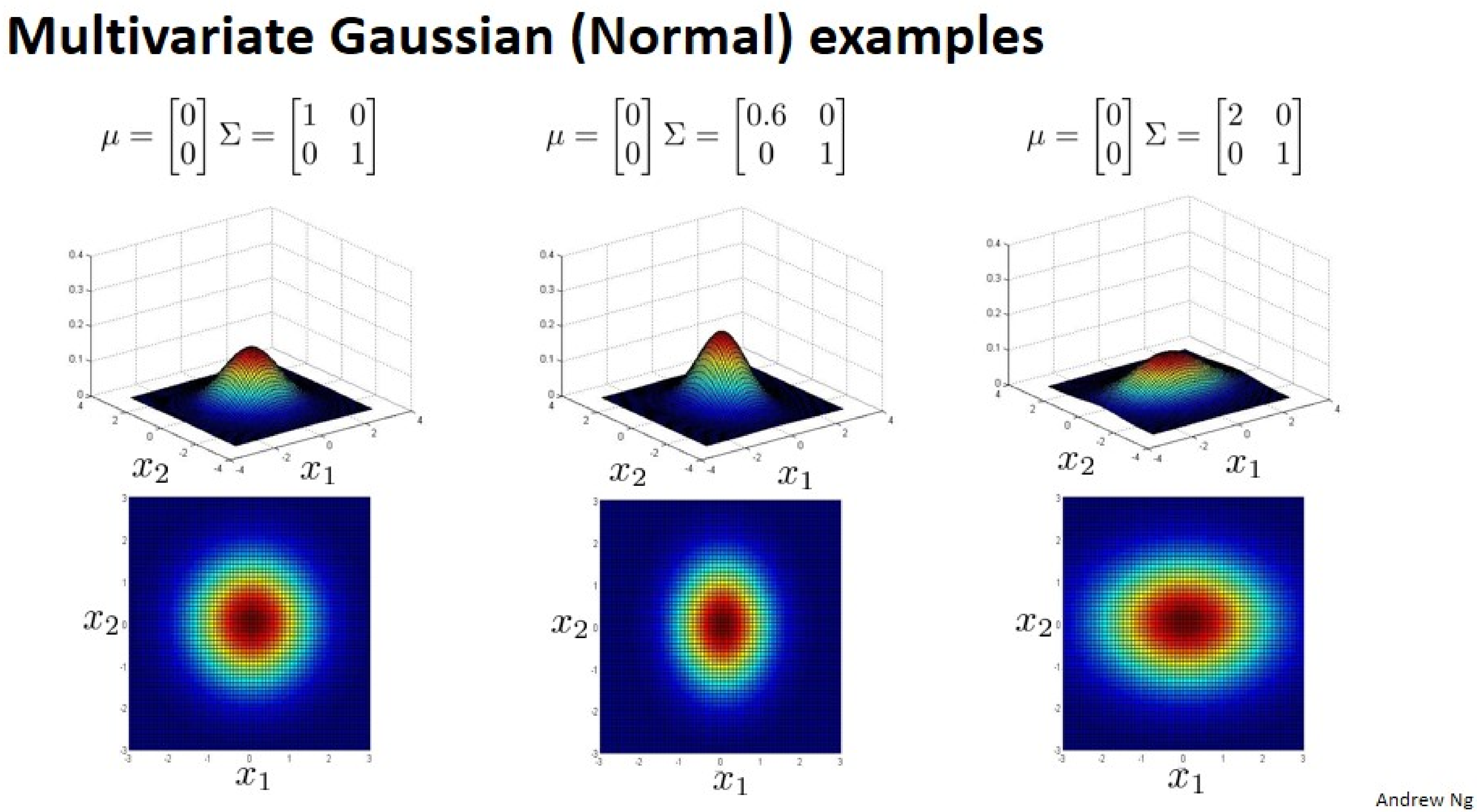
Parameter fitting:

Given training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)} \qquad \Sigma = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)}-\mu)(x^{(i)}-\mu)^T$$

Andrew Ng

Multivariate Normal distribution (2)

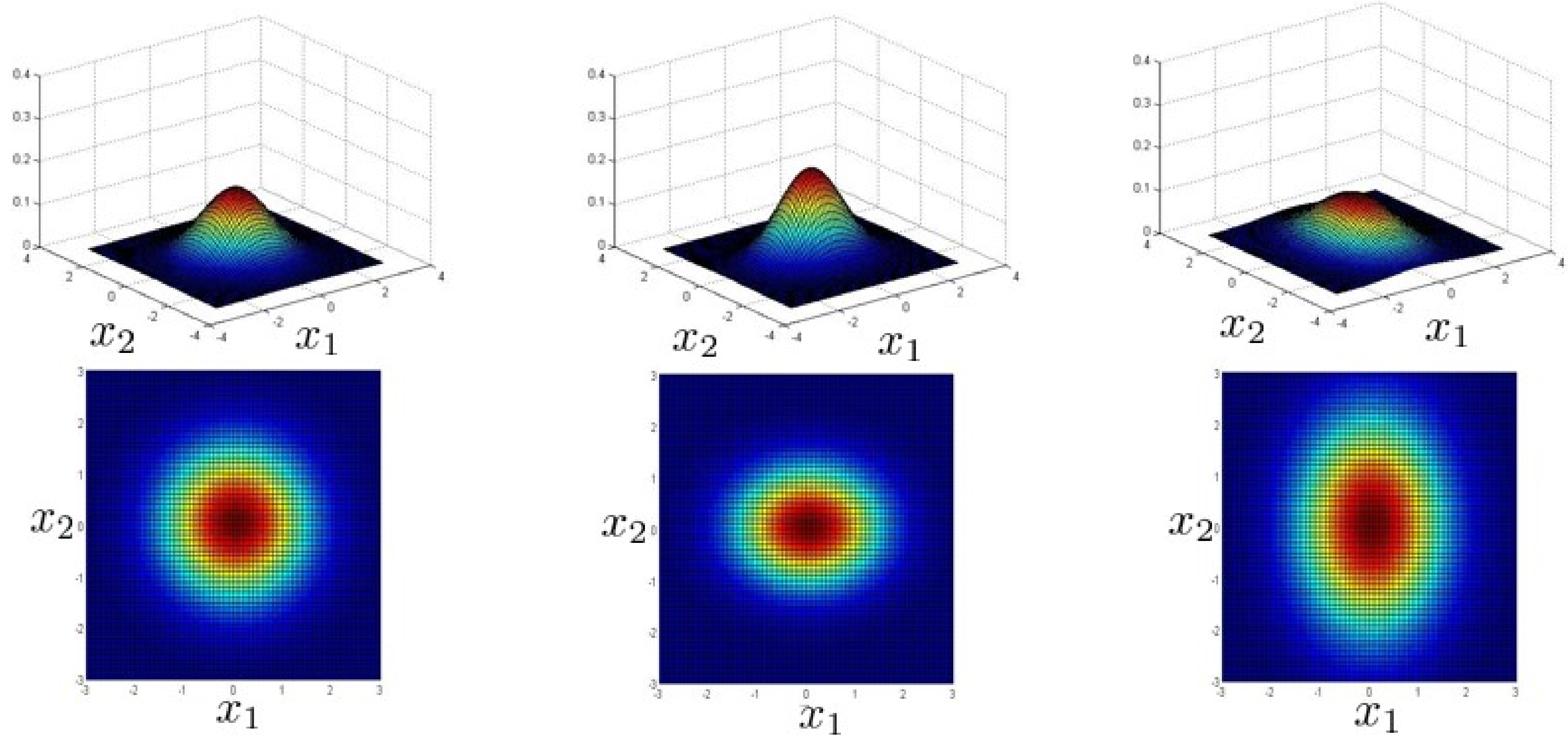


Source: Andrew Ng, Stanford University, Machine Learning course 2012, Lecture 15

Multivariate Normal distribution (3)

Multivariate Gaussian (Normal) examples

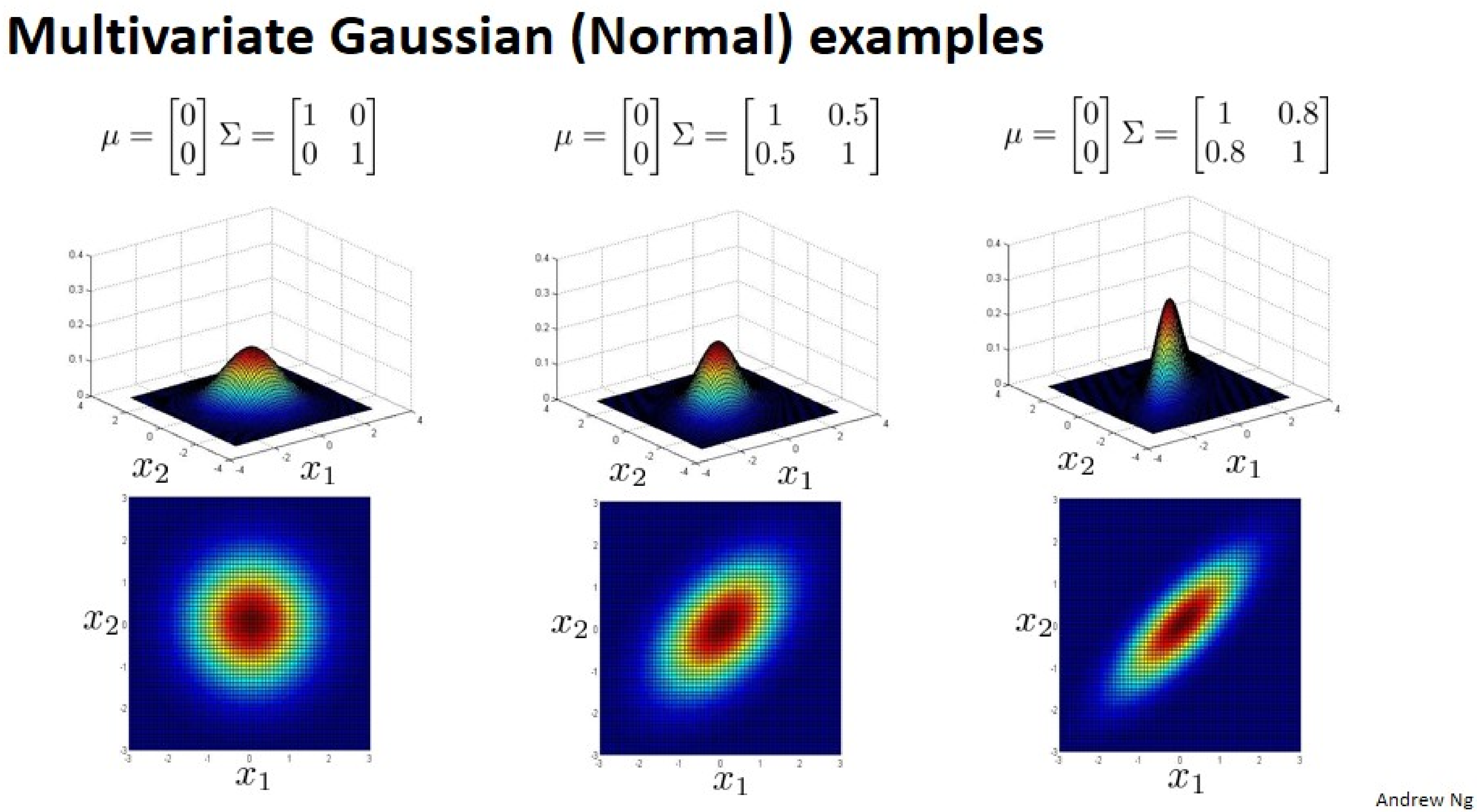
$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ $\qquad$ $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 0.6 \end{bmatrix}$ $\qquad$ $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$



Andrew Ng

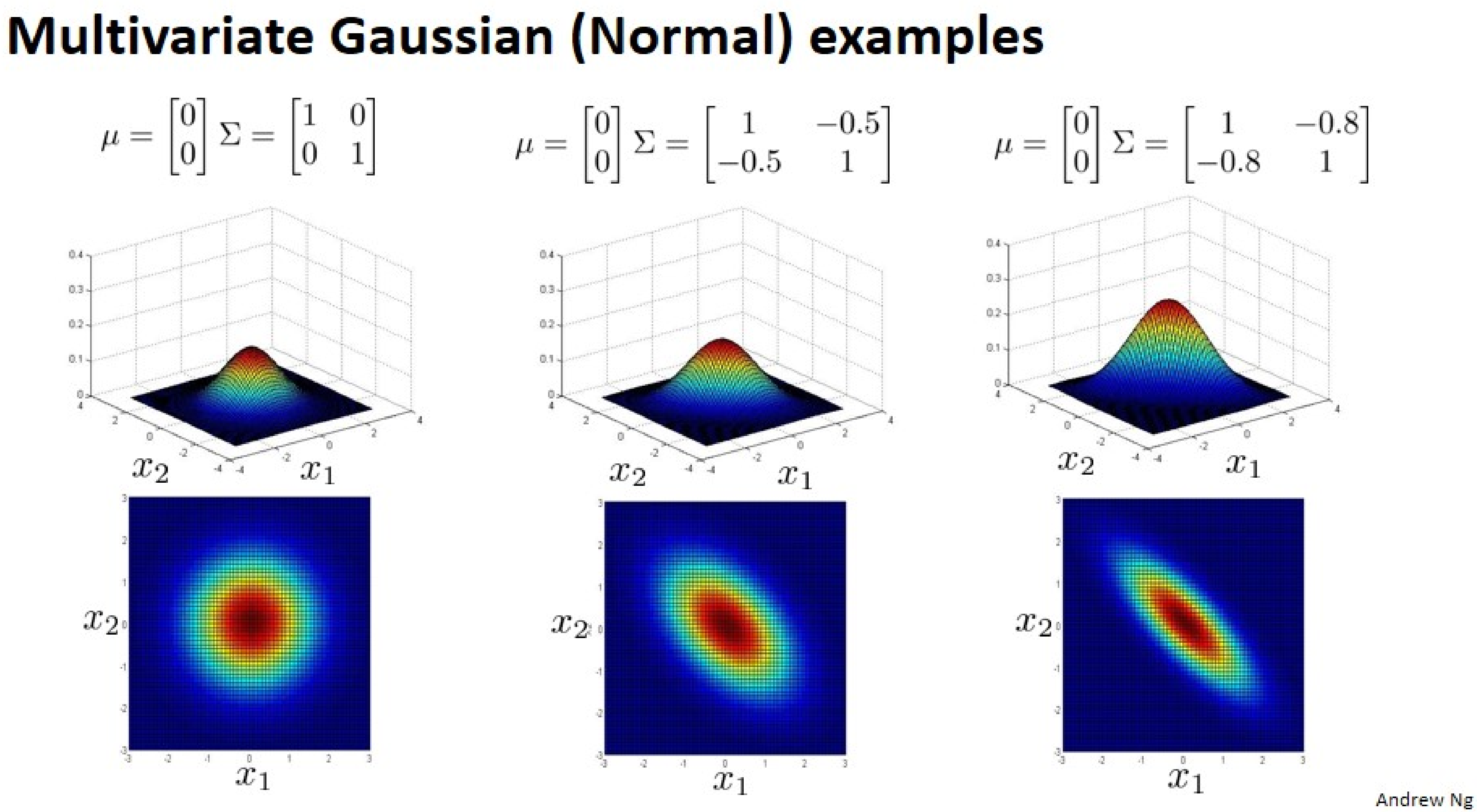
Source: Andrew Ng, Stanford University, Machine Learning course 2012, Lecture 15

Multivariate Normal distribution (4)



Source: Andrew Ng, Stanford University, Machine Learning course 2012, Lecture 15

Multivariate Normal distribution (5)

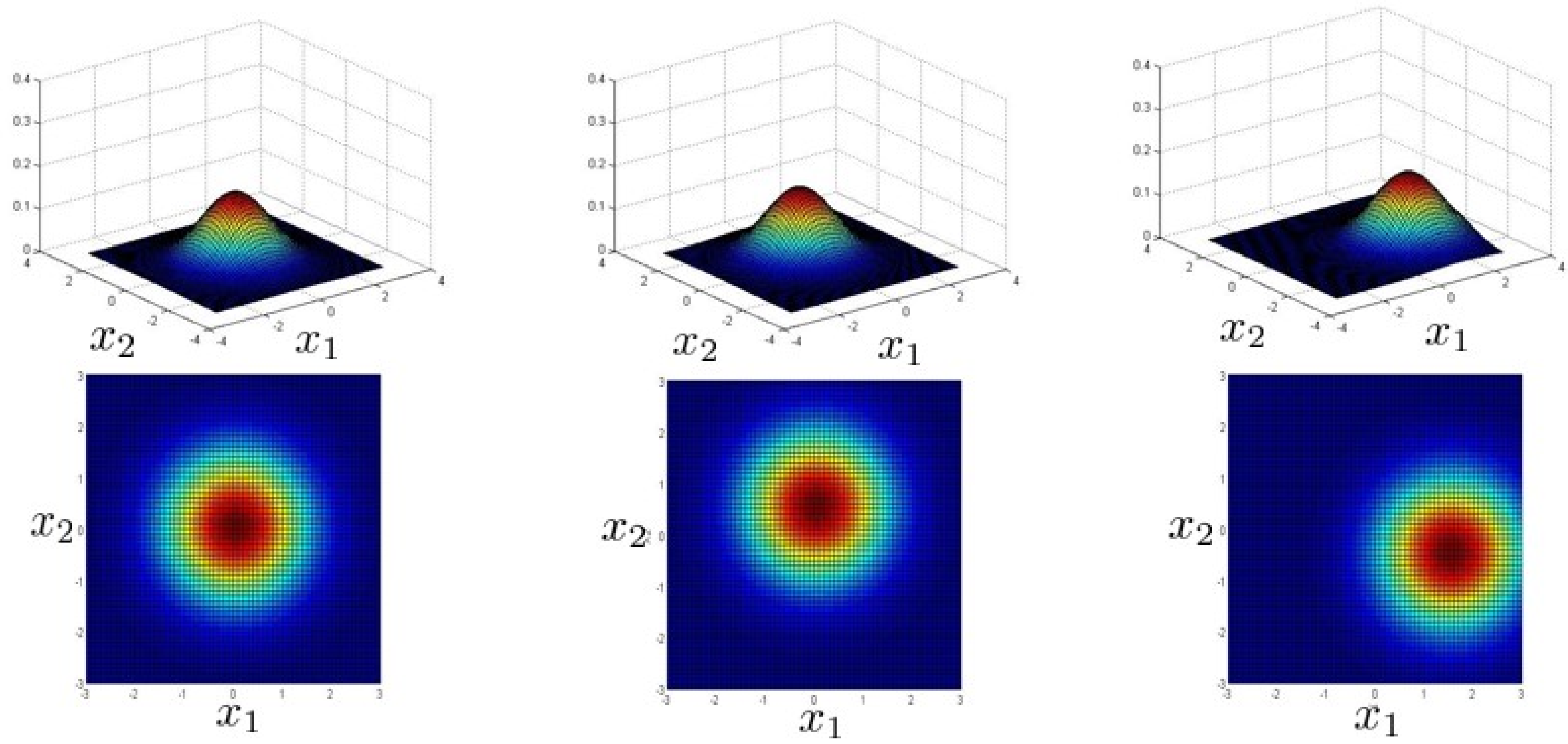


Source: Andrew Ng, Stanford University, Machine Learning course 2012, Lecture 15

Multivariate Normal distribution (6)

Multivariate Gaussian (Normal) examples

$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \; \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ $\qquad$ $\mu = \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} \; \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ $\qquad$ $\mu = \begin{bmatrix} 1.5 \\ -0.5 \end{bmatrix} \; \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$



Andrew Ng

Source: Andrew Ng, Stanford University, Machine Learning course 2012, Lecture 15

MLE - Maximum Likelihood Estimation (1)

- The likelihood $\mathcal{L}(\mathbf{x})$ is the conditional probability $p(\mathbf{z}|\mathbf{x})$ of the measurements[2] $\mathbf{z}$ given a particular true value of $\mathbf{x}$.
- If the distribution is Gaussian and observations $\mathbf{z}$ are measured, the likelihood $\mathcal{L}(\mathbf{x})$ is a function only of $\mathbf{x}$.
- How do we obtain MLE? Knowing the distribution of $\mathcal{L}(\mathbf{x})$ and measurements $\mathbf{z}$, then $\mathbf{x}$ is varied until the maximum of the distribution is found:

$$\hat{\mathbf{x}}_{MLE} = \operatorname*{argmax}_{x} p(\mathbf{z}|\mathbf{x})$$

[2] Note: The likelihood is a function of $\mathbf{x}$ but it is not a probability distribution over $\mathbf{x}$, it would be incorrect to refer to it as the *likelihood of the data.*

MLE - Maximum Likelihood Estimation (2)

Example - Sonar MLE (1)

Suppose we have two independent sonar measurements z_1, z_2 of a position x. The sensors are modeled both in the same way as $p(z_i|x) = \mathcal{N}(x, \sigma^2)$.

- Since the two sensors are independent the likelihood is:

$$\mathcal{L}(x) = p(z_1, z_2|x) = p(z_1|x)p(z_2|x)$$

- and since the sensors are Gaussian[3]:

$$\mathcal{L}(x) \sim e^{-\frac{(z_1-x)^2}{2\sigma^2}} \times e^{-\frac{(z_2-x)^2}{2\sigma^2}} = e^{-\frac{(z_1-x)^2+(z_2-x)^2}{2\sigma^2}}$$

[3] Note: we ignore the irrelevant normalization constant.

MLE - Maximum Likelihood Estimation (3)

Example - Sonar MLE (2)

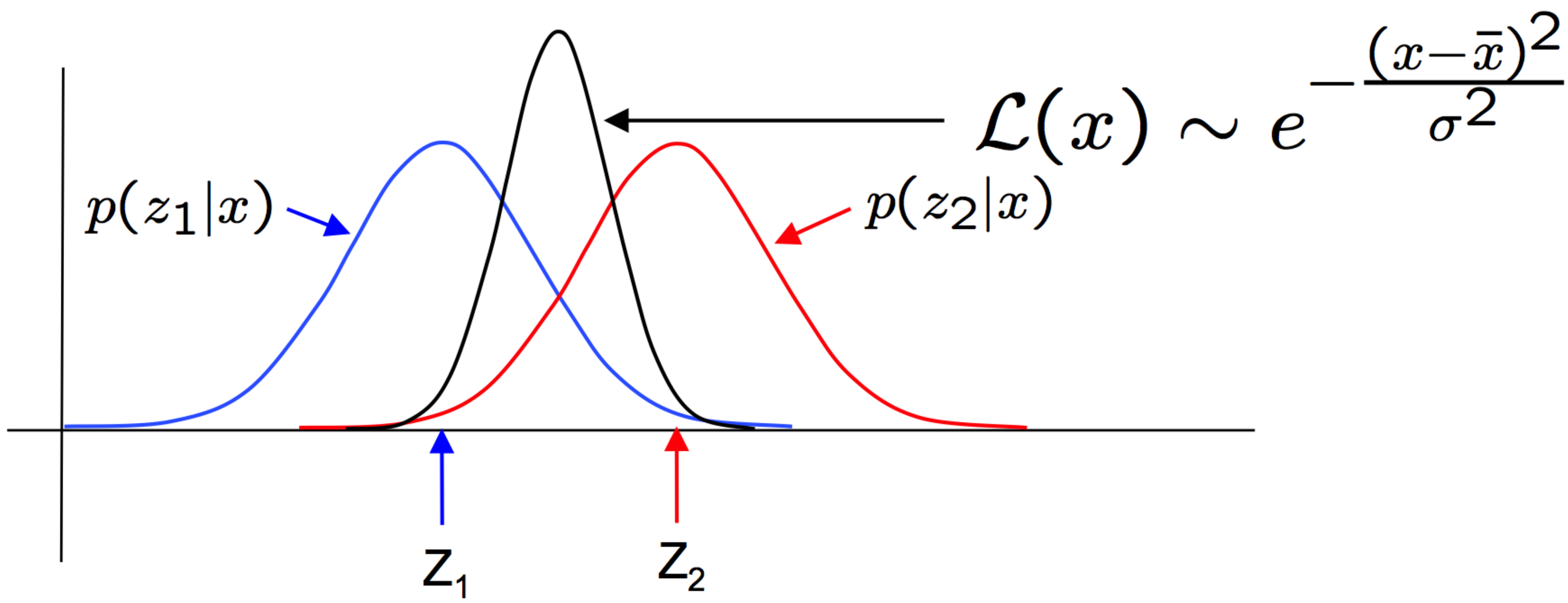
- We can express the negative log likelihood as follows:

$$-\ln \mathcal{L}(x) = \frac{(z_1 - x)^2 + (z_2 - x)^2}{2\sigma^2} = \frac{2x^2 - 2x(z_1 + z_2) + z_1^2 + z_2^2}{2\sigma^2}$$

- We redefine the MLE task to: $\hat{\mathbf{x}}_{\mathrm{MLE}} = \operatorname*{argmin}_x \; -\ln \mathcal{L}(x)$
- We minimize by differentiating w.r.t. x and setting equal to 0,
- which leads to: $\hat{\mathbf{x}}_{\mathrm{MLE}} = \frac{z_1+z_2}{2} = \overline{x}$

MLE - Maximum Likelihood Estimation (4)

Example - Sonar MLE (3)



MLE - Maximum Likelihood Estimation (5)

Example - Sonar MLE (4)

Suppose we have two independent sonar measurements z_1, z_2 of a position x, but each sensor has a different model: $p(z_1|x) = \mathcal{N}(x, \sigma_1^2)$ and $p(z_2|x) = \mathcal{N}(x, \sigma_2^2)$.

- Again, the two sensors are independent and the likelihood is:

$$\mathcal{L}(x) = p(z_1, z_2|x) = p(z_1|x)p(z_2|x) \rightarrow \mathcal{L}(x) \sim e^{-\frac{(z_1-x)^2}{2\sigma_1^2}} \times e^{-\frac{(z_2-x)^2}{2\sigma_2^2}}$$

- We express the negative log likelihood:

$$-\ln \mathcal{L}(x) = 0.5(\sigma_1^{-2}(z_1 - x)^2 + \sigma_2^{-2}(z_2 - x)^2) + \text{const}$$

- and we minimize it by differentiating w.r.t. to x and setting to 0:

$$\hat{\mathbf{x}}_{\text{MLE}} = \frac{\sigma_1^{-2} z_1 + \sigma_2^{-2} z_2}{\sigma_1^{-2} + \sigma_2^{-2}}, \; \hat{\sigma}_{\text{MLE}}^{-2} = \sigma_1^{-2} + \sigma_2^{-2}$$

MLE - Maximum Likelihood Estimation (6)

Example - Sonar MLE (5)

Now, assume we tested the sensors and we identified their variances of the measurements, such that: $p(z_1|x) \sim \mathcal{N}(x, 10^2)$ and $p(z_2|x) \sim \mathcal{N}(x, 20^2)$. What will be the MLE for these sensor readings $z_1 = 130$ and $z_2 = 170$?

$$\hat{\mathbf{x}}_{\mathrm{MLE}} = \frac{130/10^2 + 170/20^2}{1/10^2 + 1/20^2} = 138$$

$$\hat{\sigma}_{\mathrm{MLE}} = \frac{1}{\sqrt{1/10^2 + 1/20^2}} = 8.94$$

Conclusion: the ML estimate is closer to the more confident measurement.

MAP - Maximum A-Posteriori Estimation (1)

- In many cases, we already have some prior (expected) knowledge about the random variable $\mathbf{x}$, i.e. the parameters of its probability distribution $p(\mathbf{x})$.
- With the Bayes rule, we go from prior to a-posterior knowledge about $\mathbf{x}$, when given the observations $\mathbf{z}$:

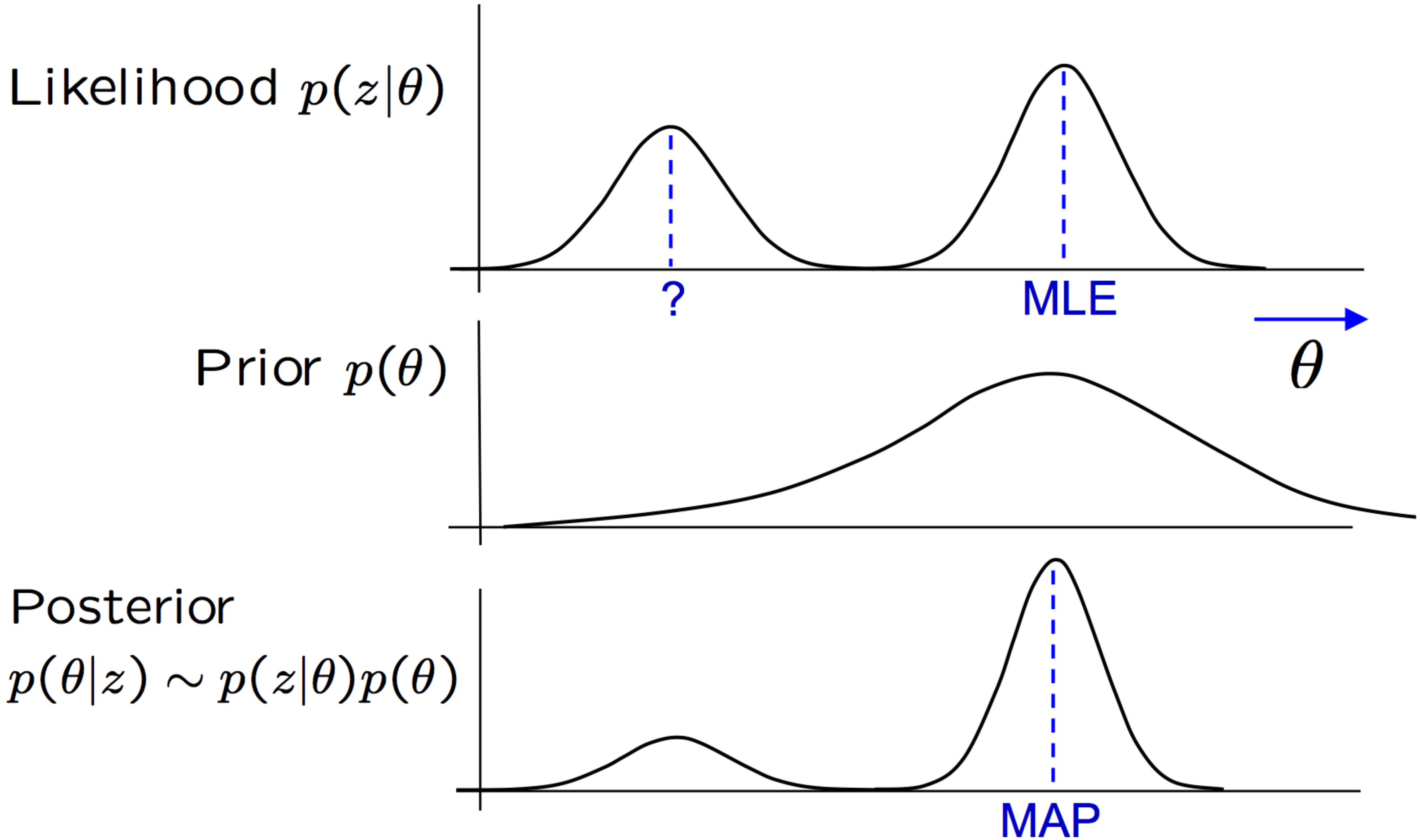
$$p(\mathbf{x}|\mathbf{z}) = \frac{p(\mathbf{z}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{z})} = \frac{\text{likelihood} \times \text{prior}}{\text{normalizing constant}} \sim C \times p(\mathbf{z}|\mathbf{x})p(\mathbf{x})$$

- Given an observation $\mathbf{z}$, a likelihood function $p(\mathbf{z}|\mathbf{x})$ and prior distribution $p(\mathbf{x})$ on $\mathbf{x}$, the maximum a posteriori estimator MAP finds the value of $\mathbf{x}$ which maximizes the posterior distribution $p(\mathbf{x}|\mathbf{z})$:

$$\hat{\mathbf{x}}_{\text{MAP}} = \operatorname*{argmax}_{x} p(\mathbf{z}|\mathbf{x})p(\mathbf{x})$$

MAP - Maximum A-Posteriori Estimation (2)

Example - Application of MAP to a random variable of θ



MAP - Maximum A-Posteriori Estimation (3)

Example - Sonar MAP (1)

Suppose we again have two independent sonar measurements z_1, z_2 of a position x, and each sensor modeled as: $p(z_1|x) = \mathcal{N}(x, \sigma_1^2)$ and $p(z_2|x) = \mathcal{N}(x, \sigma_2^2)$.

- The joint likelihood is defined as:

$$\mathcal{L}(x) = p(z_1, z_2|x) = p(z_1|x)p(z_2|x).$$

- In addition, we also have a prior (expected) information about x:

$$p(x) \sim \mathcal{N}(x_{prior}, \sigma_{prior}^2).$$

- The posterior probability density is given by a Gaussian distribution:

$$p(x|z_1, z_2) \sim p(z_1, z_2|x)p(x) \sim \mathcal{N}(x_{pos}, \sigma_{post}^2)$$

MAP - Maximum A-Posteriori Estimation (4)

Example - Sonar MAP (2)

- Using the same approach as for deriving the MLE, the mean of the posteriori distribution of MAP is obtained as:

$$x_{post} = \frac{\sigma_1^{-2} z_1 + \sigma_2^{-2} z_2 + \sigma_{prior}^{-2} x_{prior}}{\sigma_1^{-2} + \sigma_2^{-2} + \sigma_{prior}^{-2}} = \hat{\mathbf{x}}_{\mathrm{MAP}}$$

- and the variance is:

$$\sigma_{post}^{-2} = \sigma_1^{-2} + \sigma_2^{-2} + \sigma_{prior}^{-2} = \hat{\sigma}_{\mathrm{MAP}}^{-2}$$

MAP - Maximum A-Posteriori Estimation (5)

Example - Sonar MAP (3)

We assume the same sensors as in the previous example $p(z_1|x) \sim \mathcal{N}(x, 10^2)$ and $p(z_2|x) \sim \mathcal{N}(x, 20^2)$, but now consider a prior (expected) knowledge[4] $p(x) \sim \mathcal{N}(x_{prior} = 150, \sigma^2_{prior} = 30^2)$. What will be the MAP for these sensor readings $z_1 = 130$ and $z_2 = 170$?

$$\hat{\mathbf{x}}_{\mathrm{MAP}} = \frac{130/10^2 + 170/20^2 + 150/30^2}{1/10^2 + 1/20^2 + 1/30^2} = 139.04$$

$$\hat{\sigma}_{\mathrm{MAP}} = \frac{1}{\sqrt{1/10^2 + 1/20^2 + 1/30^2}} = 8.57$$

[4] Note: The prior knowledge is obtained for example statistically or from a datasheet.

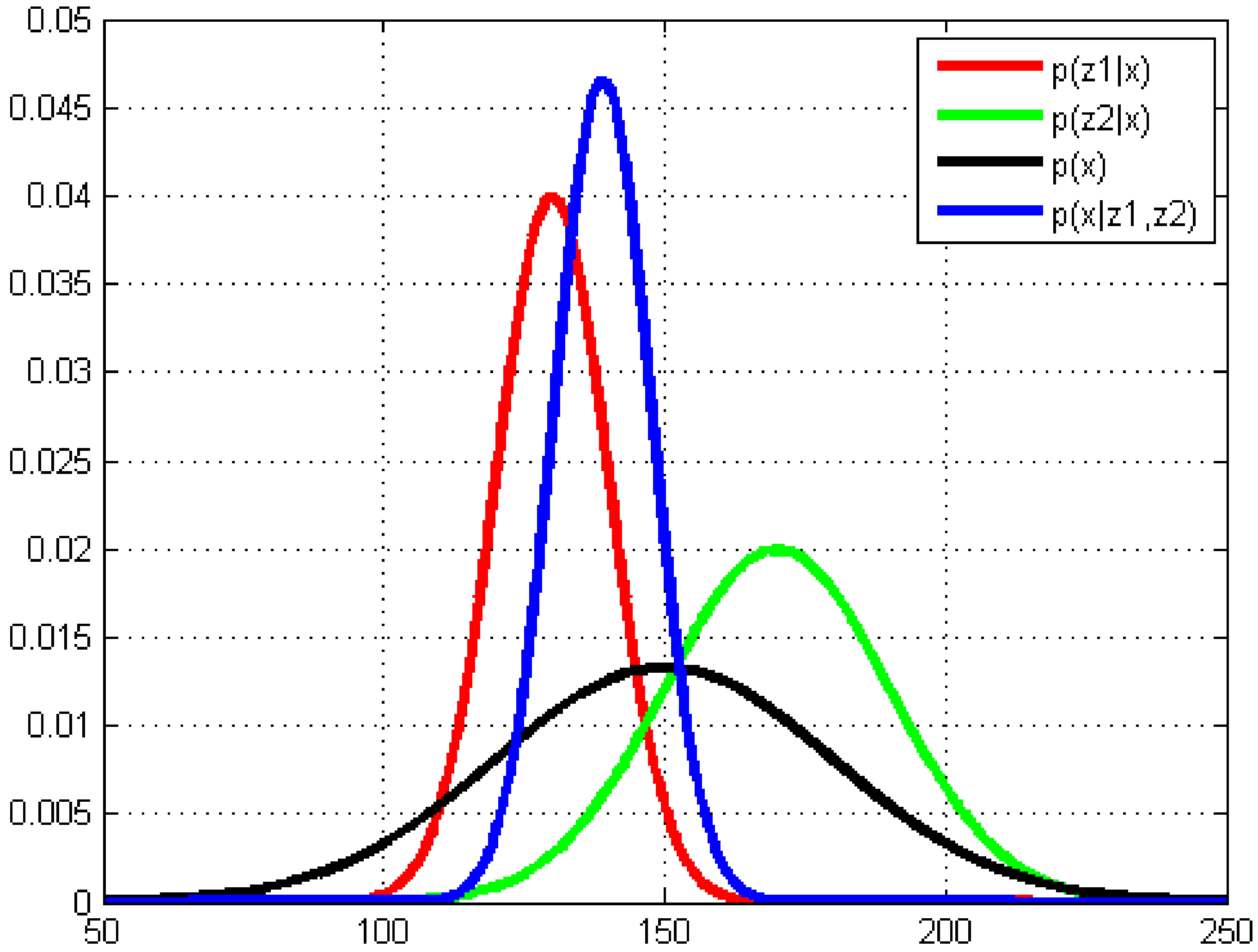
MAP - Maximum A-Posteriori Estimation (6)

Example - Sonar MAP (4)

```
step =0.1;
x = [50:step:250];
p_z1 = normpdf(x, 130,10);
p_z2 = normpdf(x, 170,20);
p_prior = normpdf(x, 150,30);
p_posterior = p_z1 .* p_z2 .* p_prior / (step*(sum(p_z1 .* p_z2 .* p_prior)));
plot(x,p_z1,'r', x,p_z2,'g', x,p_prior,'k:', x, p_posterior, 'b');
grid on
legend('p(z1|x)', 'p(z2|x)', 'p(x)', 'p(x|z1,z2)')
[val ind] = max(p_posterior);
fprintf('MAP estimate of x = %2.2f',x(ind))
```

MAP - Maximum A-Posteriori Estimation (7)

Example - Sonar MAP (5)



What is the relationship between MLE and MAP?

The relationship between MLE and MAP is the update rule:

$$\hat{\mathbf{x}}_{\mathrm{MAP}} = \frac{\sigma_{prior}^{-2} x_{prior} + \sigma_{lik}^{-2} \hat{\mathbf{x}}_{\mathrm{MLE}}}{\sigma_{prior}^{-2} + \sigma_{lik}^{-2}} = x_{prior} + \frac{\sigma_{prior}^2}{\sigma_{prior}^2 + \sigma_{lik}^2} \left(\hat{\mathbf{x}}_{\mathrm{MLE}} - x_{prior} \right)$$

- We can see that the prior acts as an additional sensor.
- If $\hat{\mathbf{x}}_{\mathrm{MLE}} = x_{prior}$ then $\hat{\mathbf{x}}_{\mathrm{MAP}}$ is unchanged by prior but variance decreases.
- If $\sigma_{lik} >> \sigma_{prior}$ then $\hat{\mathbf{x}}_{\mathrm{MAP}} \approx x_{prior}$ (noisy sensor!).
- If $\sigma_{prior} >> \sigma_{lik}$ then $\hat{\mathbf{x}}_{\mathrm{MAP}} \approx \hat{\mathbf{x}}_{\mathrm{MLE}}$ (weak prior knowledge!).

MMSE - Minimum Mean Squared Error

Without proof[5]: We want to find such a $\hat{\mathbf{x}}$, an estimate of $\mathbf{x}$, that given a set of measurements $\mathbf{Z}^k = \{\mathbf{z_1}, \mathbf{z_2}, ..., \mathbf{z_k}\}$ it minimizes the mean squared error between the true value and this estimate.[6]

$$\hat{\mathbf{x}}_{\mathrm{MMSE}} = \underset{\hat{\mathbf{x}}}{\operatorname{argmin}} \, \mathcal{E}\{(\hat{\mathbf{x}} - \mathbf{x})^\top (\hat{\mathbf{x}} - \mathbf{x}) | \mathbf{Z^k}\} = \mathcal{E}\{\mathbf{x} | \mathbf{Z^k}\}$$

Why is this important? The MMSE estimate given a set of measurements is the mean of that variable conditioned on the measurements! [7]

[5] See reference [1] pages 11-12
[6] Note: We minimize a scalar quantity.
[7] Note: In LSQ the $\mathbf{x}$ is a unknown constant but in MMSE $\mathbf{x}$ is a random variable.

RBE - Recursive Bayesian Estimation

RBE is extension of MAP to time-stamped sequence of observations.

Without proof[8]: We obtain RBE as the likelihood of current k^{th} measurement $\times$ prior which is our last best estimate of x at time $k-1$ conditioned on measurement at time $k-1$ (*denominator is just a normalizing constant*).

$$p(\mathbf{x}|\mathbf{Z^k}) = \frac{p(\mathbf{z}_k|\mathbf{x})p(\mathbf{x}|\mathbf{Z}^{k-1})}{p(\mathbf{z}_k|\mathbf{Z}^{k-1})} = \frac{\text{current likelihood} \times \text{last best estimate}}{\text{normalizing constant}}$$

[8] See reference [1] pages 12-14, note: if Gaussian *pdf* of both prior and likelihood then the RBE $\rightarrow$ the LKF

LSQ - Least Squares Estimation (1)

Given measurements $\mathbf{z}$, we wish to solve for $\mathbf{x}$, assuming linear relationship:

$$\mathbf{H}\mathbf{x} = \mathbf{z}$$

If $\mathbf{H}$ is a square matrix with $\det \mathbf{H} \neq 0$ then the solution is trivial:

$$\mathbf{x} = \mathbf{H}^{-1}\mathbf{z},$$

otherwise (most commonly), we seek such solution $\hat{\mathbf{x}}$ that is closest (in Euclidean distance sense) to the ideal:

$$\hat{\mathbf{x}} = \operatorname*{argmin}_x ||\mathbf{H}\mathbf{x} - \mathbf{z}||^2 = \operatorname*{argmin}_x \left\{ (\mathbf{H}\mathbf{x} - \mathbf{z})^\top (\mathbf{H}\mathbf{x} - \mathbf{z}) \right\}$$

LSQ - Least Squares Estimation (2)

Given the following matrix identities:

- $(\mathbf{AB})^\top = \mathbf{B}^\top \mathbf{A}^\top$
- $||\mathbf{x}||^2 = \mathbf{x}^\top \mathbf{x}$
- $\nabla_x \, \mathbf{b}^\top \mathbf{x} = \mathbf{b}$
- $\nabla_x \, \mathbf{x}^\top \mathbf{A} \mathbf{x} = 2\mathbf{A}\mathbf{x}$

We can derive the closed form solution[9]:

$$||\mathbf{Hx} - \mathbf{z}||^2 = \mathbf{x}^\top \mathbf{H}^\top \mathbf{H} \mathbf{x} - \mathbf{x}^\top \mathbf{H}^\top \mathbf{z} - \mathbf{z}^\top \mathbf{H} \mathbf{x} + \mathbf{z}^\top \mathbf{z}$$

$$\frac{\partial ||\mathbf{Hx} - \mathbf{z}||^2}{\partial \mathbf{x}} = 2\mathbf{H}^\top \mathbf{H} \mathbf{x} - 2\mathbf{H}^\top \mathbf{z} = 0$$

$$\Rightarrow \quad \mathbf{x} = (\mathbf{H}^\top \mathbf{H})^{-1} \mathbf{H}^\top \mathbf{z}$$

[9] in MATLAB use the pseudo-inverse *pinv()*

LSQ - Least Squares Estimation (3)

The world is non-linear → nonlinear model function $\mathbf{h}(\mathbf{x})$ → non-linear LSQ[10]:

$$\hat{\mathbf{x}} = \operatorname*{argmin}_{x} ||(\mathbf{h}(\mathbf{x}) - \mathbf{z})||^2$$

- We seek such δ that for $\mathbf{x}_1 = \mathbf{x}_0 + \delta$ the $||\mathbf{h}(\mathbf{x}_1) - \mathbf{z}||^2$ is minimized.
- We use Taylor series expansion: $\mathbf{h}(\mathbf{x}_0 + \delta) = \mathbf{h}(\mathbf{x}_0) + \nabla \mathbf{H}_{\mathbf{x}0}\delta$

$$||\mathbf{h}(\mathbf{x}_1) - \mathbf{z}||^2 = ||\mathbf{h}(\mathbf{x}_0) + \nabla \mathbf{H}_{\mathbf{x}0}\delta - \mathbf{z}||^2 = ||\underbrace{\nabla \mathbf{H}_{\mathbf{x}0}}_{\mathbf{A}}\delta - \underbrace{(\mathbf{z} - \mathbf{h}(\mathbf{x}_0))}_{\mathbf{b}})||^2$$

where $\nabla \mathbf{H}_{\mathbf{x}0}$ is Jacobian of $\mathbf{h}(\mathbf{x})$:

$$\nabla \mathbf{H}_{\mathbf{x}0} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{h}_1}{\partial \mathbf{x}_1} & \cdots & \frac{\partial \mathbf{h}_1}{\partial \mathbf{x}_m} \\ \vdots & & \vdots \\ \frac{\partial \mathbf{h}_n}{\partial \mathbf{x}_1} & \cdots & \frac{\partial \mathbf{h}_n}{\partial \mathbf{x}_m} \end{bmatrix}$$

[10] Note: We still measure the Euclidean distance between two points that we want to optimize over.

LSQ - Least Squares Estimation (4)

The extension of LSQ to the non-linear LSQ can be formulated as an algorithm:

1. Start with an initial guess $\hat{\mathbf{x}}$. [11]
2. Evaluate the LSQ expression for δ (update the $\nabla \mathbf{H}_{\hat{\mathbf{x}}}$ and substitute). [12]

$$\delta := (\nabla \mathbf{H}_{\hat{\mathbf{x}}}{}^\top \nabla \mathbf{H}_{\hat{\mathbf{x}}})^{-1} \nabla \mathbf{H}_{\hat{\mathbf{x}}}{}^\top [\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}})]$$

3. Apply the δ correction to our initial estimate: $\hat{\mathbf{x}} := \hat{\mathbf{x}} + \delta$.[13]
4. Check for the stopping precision: if $||\mathbf{h}(\hat{\mathbf{x}}) - \mathbf{z}||^2 > \epsilon$ proceed with step (2) or stop otherwise.[14]

[11] Note: We can usually set to zero.

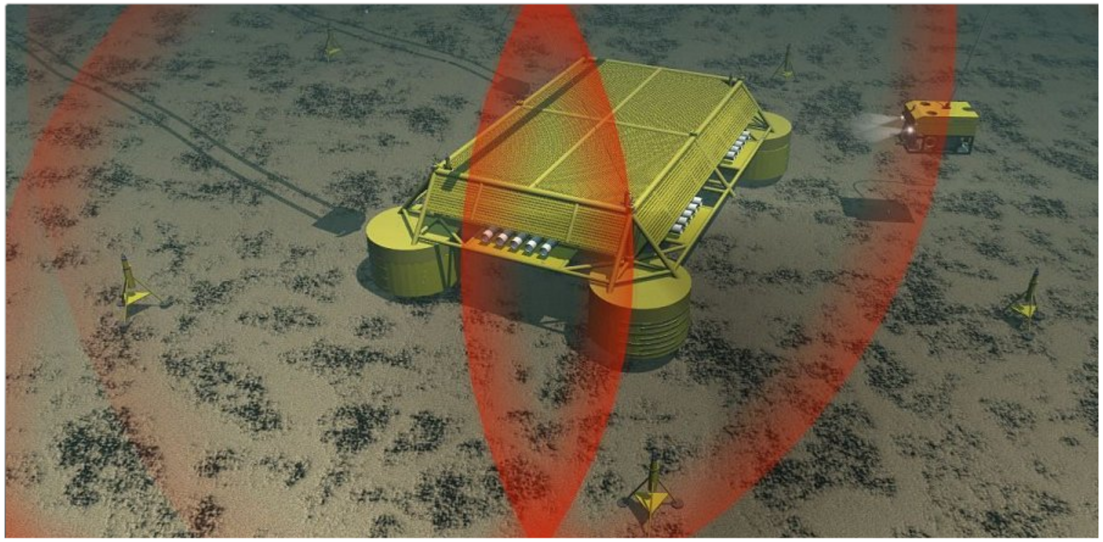
[12] Note: This expression is obtained using the LSQ closed form and substitution from previous slide.

[13] Note: Due to these updates our initial guess should converge to such $\hat{\mathbf{x}}$ that minimizes the $||\mathbf{h}(\hat{\mathbf{x}}) - \mathbf{z}||^2$

[14] Note: ϵ is some small threshold, usually set according to the noise level in the sensors.

LSQ - Least Squares Estimation (5)

Example - Long Base-line Navigation (1) SONARDYNE



LSQ - Least Squares Estimation (6)

Example - Long Base-line Navigation (2)

Assume an underwater robot operating within the range of 4 beacons and receiving time-of-flight measurements simultaneously and without delay.

We wish to find the LSQ estimate of robot position $\mathbf{x}_v = [x, y, z]^\top$ while each beacon i is at known position $\mathbf{x}_{bi} = [x_{bi}, y_{bi}, z_{bi}]^\top$. The observation model is[15]:

$$\mathbf{z} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \end{bmatrix} = h(\mathbf{x}_v) = \frac{2}{c} \begin{bmatrix} ||\mathbf{x}_{b1} - \mathbf{x}_v|| \\ ||\mathbf{x}_{b2} - \mathbf{x}_v|| \\ ||\mathbf{x}_{b3} - \mathbf{x}_v|| \\ ||\mathbf{x}_{b4} - \mathbf{x}_v|| \end{bmatrix}$$

where t_i is the measured time-of-flight from beacon i.

[15] Note: We assume the transceiver operates at speed of sound c

LSQ - Least Squares Estimation (7)

Example - Long Base-line Navigation (3)

We derive the $\nabla \mathbf{H}_{\mathbf{x}v}$ and plug it into the 4-step algorithm already introduced:

$$\nabla \mathbf{H}_{\mathbf{x}v} = -\frac{2}{c} \begin{bmatrix} \Delta_{x1} & \Delta_{y1} & \Delta_{z1} \\ \Delta_{x2} & \Delta_{y2} & \Delta_{z2} \\ \Delta_{x3} & \Delta_{y3} & \Delta_{z3} \\ \Delta_{x4} & \Delta_{y4} & \Delta_{z4} \end{bmatrix}$$

where:

$$\Delta_{xi} = (x_{bi} - x)/r_i, \Delta_{yi} = (y_{bi} - y)/r_i, \Delta_{zi} = (z_{bi} - z)/r_i$$

$$r_i = \sqrt{(x_{bi} - x)^2 + (y_{bi} - y)^2 + (z_{bi} - z)^2}$$

LSQ - Least Squares Estimation (8)

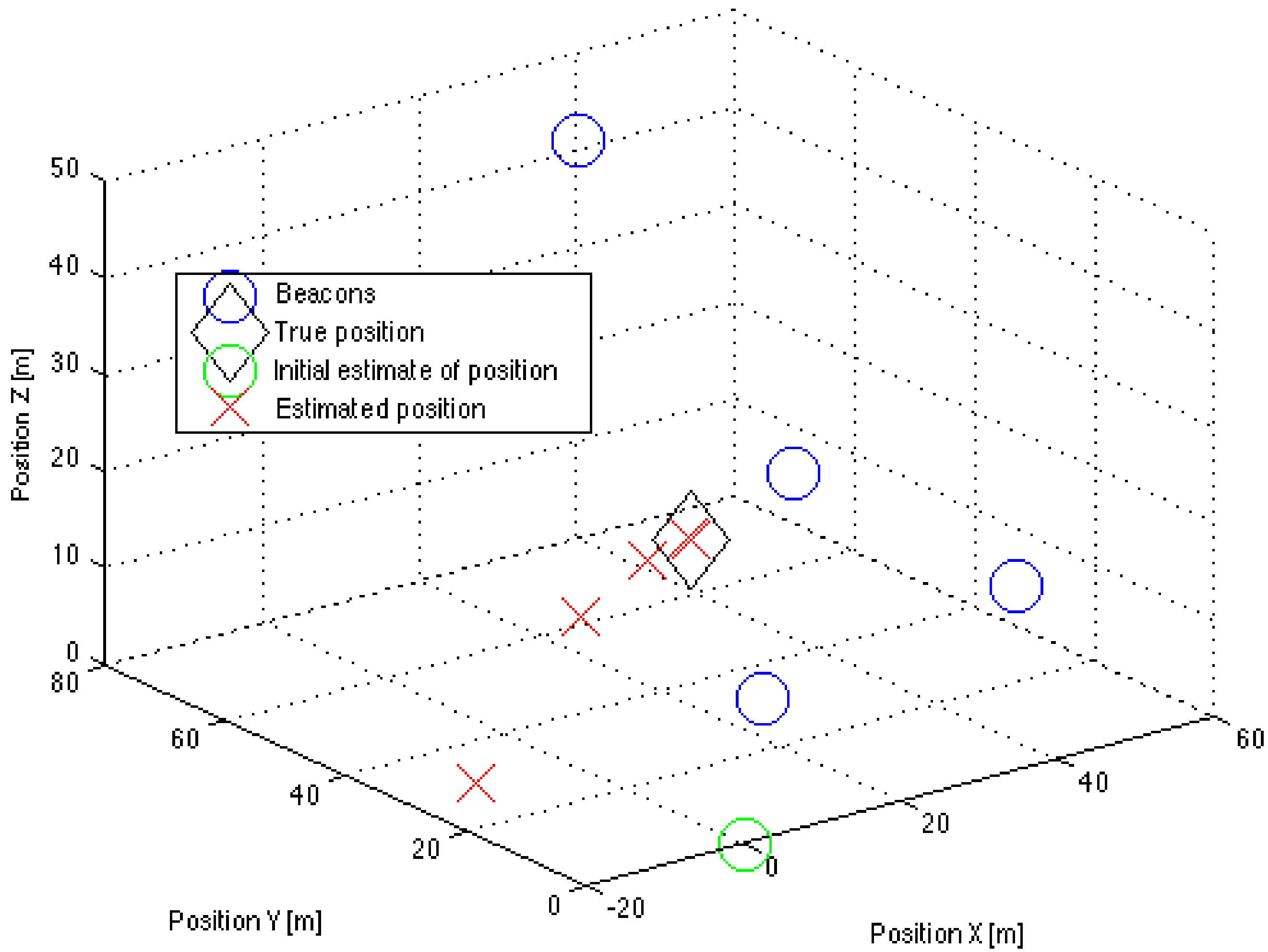
Example - Long Base-line Navigation (4)

```
%% Non-linear least squares solution to the Long Base-line Navigation
precision_history = [];                       % initialization precision history [m]
desired_precision = 0.001;                    % desired precision of the estimated position [m]
c = 343;                                      % speed fo sound [mps]
dH = zeros(4,3);                              % initial Jacobian values
Xb = [10 50 60 25; 10 20 70 60; 10 10 5 50];  % known beacon positions [m]
Xv_est = [0; 0; 0];                           % initial estimate of vehicle position [m]
Xv_true = [5.123; 15.456; 25.789];            % unknown true vehicle position [m]
% generating time-of-flight measurements (no sensor noise assumed):
Xdiff_true = Xb - repmat(Xv_true, 1, size(Xb, 2));
Ztof = 2*([norm(Xdiff_true(:,1)); norm(Xdiff_true(:,2)); norm(Xdiff_true(:,3)); norm(Xdiff_true(:,4))])/c;

Xdiff_est = Xb - repmat(Xv_est, 1, size(Xb, 2));
Hest = 2*([norm(Xdiff_est(:,1)); norm(Xdiff_est(:,2)); norm(Xdiff_est(:,3)); norm(Xdiff_est(:,4))])/c;
precision = 0.5*c*norm(Ztof - Hest);
while  precision > desired_precision
% updating the Jacobian
    for i=1:size(Xb,2)
        dH(i,:) = -2/c*transpose(Xdiff_est(:,i)./norm(Xdiff_est(:,i)));
    end
% updating the position estimate
Xv_est = Xv_est + pinv(dH'*dH)*dH'*(Ztof - Hest);
% propagating new estimate thrgough the observation model
Xdiff_est = Xb - repmat(Xv_est, 1, size(Xb, 2));
Hest = 2*([norm(Xdiff_est(:,1)); norm(Xdiff_est(:,2)); norm(Xdiff_est(:,3)); norm(Xdiff_est(:,4))])/c;
% updating the precision of the current estimate
precision = 0.5*c*norm(Ztof - Hest); %[m]
end
```

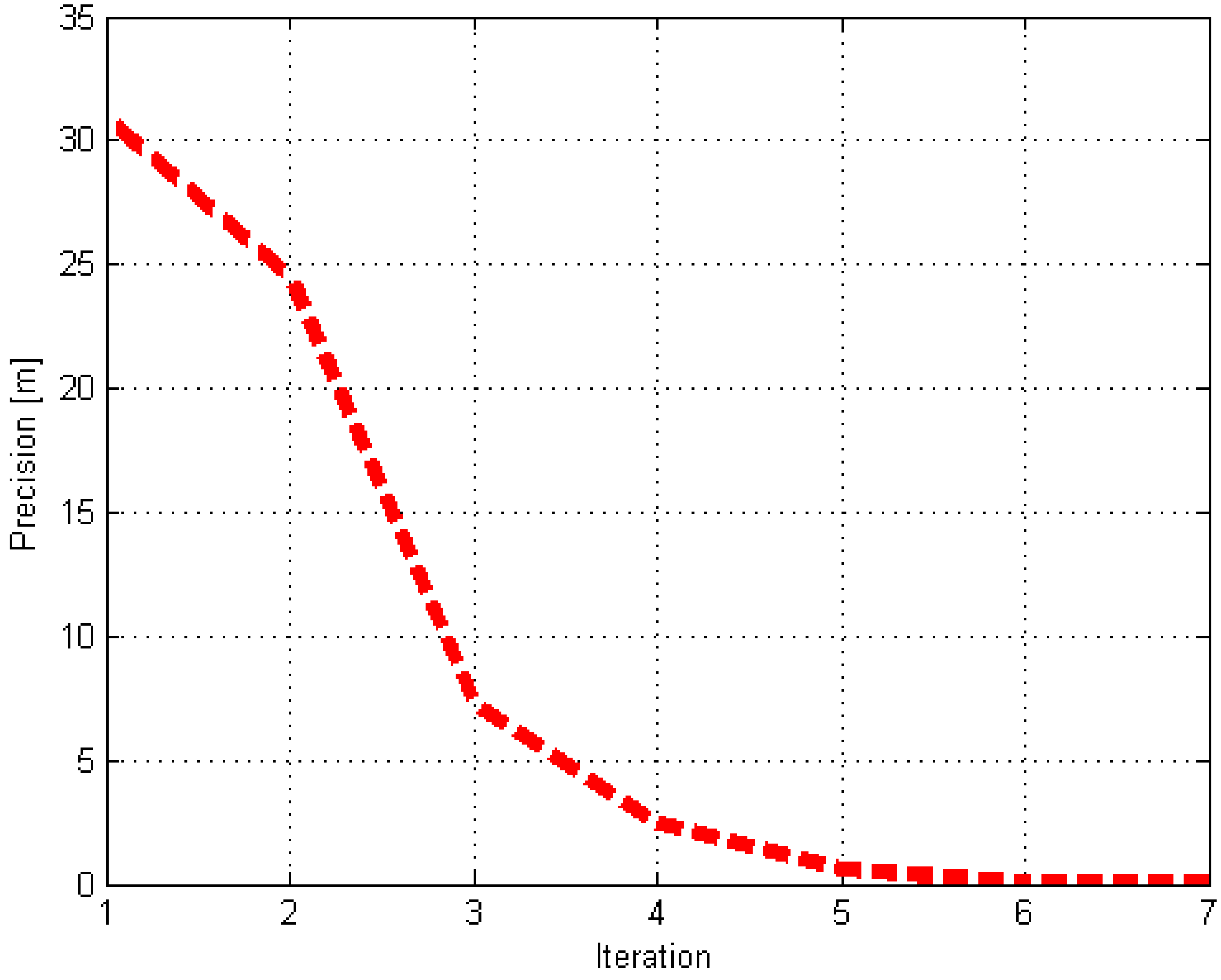
LSQ - Least Squares Estimation (9)

Example - Long Base-line Navigation (5)



LSQ - Least Squares Estimation (10)

Example - Long Base-line Navigation (6)



Overview of Estimators

What have we learnt so far?

- MLE - we have the likelihood (conditional probability of measurements)
- MAP - we have the likelihood and some prior (expected) knowledge
- MMSE - we have a set of measurements of a random variable
- RBE - we have the MAP and incoming sequence of measurements
- LSQ - we have a set of measurements and some knowledge about the underlying model (linear or non-linear)

What comes next?

The Kalman filter - we have sequence of measurements and a state-space model providing the relationship between the states and the measurements (linear model $\rightarrow$ LKF, non-linear model $\rightarrow$ EKF)

LKF - Assumptions

The likelihood $p(\mathbf{z}|\mathbf{x})$ and the prior $p(\mathbf{x})$ on $\mathbf{x}$ are Gaussian, and the linear measurement model $\mathbf{z} = \mathbf{H}\mathbf{x} + \mathbf{w}$ is corrupted by Gaussian noise $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$:

$$p(\mathbf{w}) = \frac{1}{(2\pi)^{n/2}|\mathbf{R}|^{1/2}} \exp\{-\frac{1}{2}\mathbf{w}^\top \mathbf{R}^{-1}\mathbf{w}\}$$

The likelihood $p(\mathbf{z}|\mathbf{x})$ is now a multi-D Gaussian[16]:

$$p(\mathbf{z}|\mathbf{x}) = \frac{1}{(2\pi)^{n_z/2}|\mathbf{R}|^{1/2}} \exp\{-\frac{1}{2}(\mathbf{z} - \mathbf{H}\mathbf{x})^\top \mathbf{R}^{-1}(\mathbf{z} - \mathbf{H}\mathbf{x})\}$$

The prior belief in $\mathbf{x}$ with mean $\mathbf{x}_\ominus$ and covariance $\mathbf{P}_\ominus$ is a multi-D Gaussian:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{n_x/2}|\mathbf{P}_\ominus|^{1/2}} \exp\{-\frac{1}{2}(\mathbf{x} - \mathbf{x}_\ominus)^\top \mathbf{P}_\ominus^{-1}(\mathbf{x} - \mathbf{x}_\ominus)\}$$

We want the a-posteriori estimate $p(\mathbf{x}|\mathbf{z})$ that is also a multi-D Gaussian, with mean $\mathbf{x}_\oplus$ and covariance $\mathbf{P}_\oplus$ $\rightarrow$ the equations of the LKF.

[16] Note: n_z is the dimension of the observation vector and n_x is the dimension of the state vector.

LKF - The proof?

Without proof[17], here are the main ideas exploited while deriving the LKF:

- We use the Bayes rule to express the $p(\mathbf{x}|\mathbf{z}) \rightarrow$ the MAP[18]
- We know that Gaussian $\times$ Gaussian $=$ Gaussian
- Considering the above, the new mean $\mathbf{x}_{\oplus}$ will be the MMSE estimate,
- the new covariance $\mathbf{P}_{\oplus}$ is derived using a *crazy matrix identity*

[17] See reference [1] pages 22-26

[18] Note: Recall the Bayes rule $p(\mathbf{x}|\mathbf{z}) = \frac{p(\mathbf{z}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{z})} = \frac{p(\mathbf{z}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{z})} = \frac{p(\mathbf{z}|\mathbf{x})p(\mathbf{x})}{\int_{-\infty}^{+\infty} p(\mathbf{z}|\mathbf{x})p(\mathbf{x})\,dx} = \frac{p(\mathbf{z}|\mathbf{x})p(\mathbf{x})}{\text{normalising const}}$

LKF - Update Equations

We defined a linear observation model mapping the measurements $\mathbf{z}$ with uncertainty (covariance) $\mathbf{R}$ onto the states $\mathbf{x}$ using a prior mean estimate $\mathbf{x}_\ominus$ with prior covariance $\mathbf{P}_\ominus$.

The LKF update: the new mean estimate $\mathbf{x}_\oplus$ and its covariance $\mathbf{P}_\oplus$:

$$\mathbf{x}_\oplus = \mathbf{x}_\ominus + \mathbf{W}\nu$$

$$\mathbf{P}_\oplus = \mathbf{P}_\ominus - \mathbf{W}\mathbf{S}\mathbf{W}^\top$$

– where ν is the innovation given by: $\nu = \mathbf{z} - \mathbf{H}\mathbf{x}_\ominus$,
– where S is the innovation covariance given by: $\mathbf{S} = \mathbf{H}\mathbf{P}_\ominus\mathbf{H}^\top + \mathbf{R}$,[19]
– where W is the Kalman gain ($\sim$ the weights!) given by: $\mathbf{W} = \mathbf{P}_\ominus\mathbf{H}^\top\mathbf{S}^{-1}$.

What if we want to estimate states we don't measure? $\rightarrow$ model

[19] Note: Recall that if $x \sim \mathcal{N}(\mu, \Sigma)$ and $y = Mx$ then $y \sim \mathcal{N}(\mu, M\Sigma M^\top)$

LKF - System Model Definition

Standard state-space description of a discrete-time system:

$$\mathbf{x}_{(k)} = \mathbf{F}\mathbf{x}_{(k-1)} + \mathbf{B}\mathbf{u}_{(k)} + \mathbf{G}\mathbf{v}_{(k)}$$

– where $\mathbf{v}$ is a zero mean Gaussian noise $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ capturing the uncertainty (imprecisions) of our transition model (*mapped by* $\mathbf{G}$ *onto the states*),
– where $\mathbf{u}$ is the control vector[20] (*mapped by* $\mathbf{B}$ *onto the states*),
– where $\mathbf{F}$ is the state transition matrix[21].

[20] For example the steering angle on a car as input by the driver.
[21] For example the differential equations of motion relating the position, velocity and acceleration.

LKF - Temporal-Conditional Notation

The temporal-conditional[22] notation, noted as $(i|j)$, defines $\hat{\mathbf{x}}_{(i|j)}$ as the MMSE estimate of $\mathbf{x}$ at time i given measurements up until and including the time j, leading to two cases:

- $\hat{\mathbf{x}}_{(k|k)}$ estimate at k given all available measurements $\rightarrow$ the estimate
- $\hat{\mathbf{x}}_{(k|k-1)}$ estimate at k given the first $k-1$ measurements $\rightarrow$ the prediction

[22] This notation is necessary to introduce when incorporating the state-space model into the LKF equations.

LKF - Incorporating System Model

The LKF prediction: using $(i|j)$ notation

$$\hat{\mathbf{x}}_{(k|k-1)} = \mathbf{F}\hat{\mathbf{x}}_{(k-1|k-1)} + \mathbf{B}\mathbf{u}_{(k)}$$

$$\mathbf{P}_{(k|k-1)} = \mathbf{F}\mathbf{P}_{(k-1|k-1)}\mathbf{F}^{\top} + \mathbf{G}\mathbf{Q}\mathbf{G}^{\top}$$

The LKF update: using $(i|j)$ notation

$$\hat{\mathbf{x}}_{(k|k)} = \hat{\mathbf{x}}_{(k|k-1)} + \mathbf{W}_{(k)}\nu_{(k)}$$

$$\mathbf{P}_{(k|k)} = \mathbf{P}_{(k|k-1)} - \mathbf{W}_{(\mathbf{k})}\mathbf{S}\mathbf{W}_{(\mathbf{k})}{}^{\top}$$

– where ν is the innovation: $\nu_{(k)} = \mathbf{z}_{(\mathbf{k})} - \mathbf{H}\hat{\mathbf{x}}_{(k|k-1)}$
– where S is the innovation covariance: $\mathbf{S} = \mathbf{H}\mathbf{P}_{(k|k-1)}\mathbf{H}^{\top} + \mathbf{R}$
– where W is the Kalman gain($\sim$ the weights!): $\mathbf{W}_{(k)} = \mathbf{P}_{(k|k-1)}\mathbf{H}^{\top}\mathbf{S}^{-1}$

LKF - Discussion (1)

- **Recursion**: the LKF is recursive, the output of one iteration is the input to next iteration.
- **Initialization:** the $\mathbf{P}_{(0|0)}$ and $\hat{\mathbf{x}}_{(0|0)}$ have to be provided. [23]
- **Predictor-corrector structure**:

  the prediction is corrected by fusion of measurements via innovation, which is the difference between the actual observation $\mathbf{z}_{(k)}$ and the predicted observation $\mathbf{H}\hat{\mathbf{x}}_{(k|k-1)}$.

[23] Note: It can be some initial good guess or even zero for mean, one for covariance.

LKF - Discussion (2)

- **Asynchrosity**: The update step only proceeds when the measurements come, not necessarily at every iteration. [24]
- **Prediction covariance increases:** since the model is inaccurate the uncertainty in predicted states increases with each prediction by adding the $\mathbf{GQG}^\top$ term $\rightarrow$ the $\mathbf{P}_{k|k-1}$ prediction covariance increases.
- **Update covariance decreases**: due to observations the uncertainty in predicted states decreases / not increases by subtracting the positive semi-definite $\mathbf{WSW}^\top$[25] $\rightarrow$ the $\mathbf{P}_{k|k}$ update covariance decreases / not increases.

[24] Note: If at time-step k there is no observation then the best estimate is simply the prediction $\hat{\mathbf{x}}_{(k|k-1)}$ usually implemented as setting the Kalman gain to 0 for that iteration.

[25] Each observation, even the not accurate one, contains some additional information that is added to the state estimate at each update.

LKF - Discussion (3)

- **Observability**: the measurements $\mathbf{z}$ need not to fully determine the state vector $\mathbf{x}$, the LKF can perform[26] updates using only partial measurements thanks to:

  – prior info about unobserved states and

  – correlations.[27]

- **Correlations**:

  – the diagonal elements of $\mathbf{P}$ are the principal uncertainties (variance) of each of the state vector elements.

  – the off-diagonal terms of $\mathbf{P}$ capture the correlations between different elements of $\mathbf{x}$.

Conclusion: The KF exploits the correlations to update states that are not observed directly by the measurement model.

[26] Note: In contrary to LSQ that needs enough measurements to solve for the state values.
[27] Note: Over the time for unobservable states the covariance will grow without bound.

LKF - Linear Navigation Problem (1)

Example - Planet Lander: State-space model

A lander observes its altitude $\mathbf{x}$ above planet using time-of-flight radar. Onboard controller needs estimates of height and velocity to actuate the rockets $\rightarrow$ discrete time 1D model:

$$\mathbf{x}_{(k)} = \underbrace{\begin{bmatrix} 1 & \delta T \\ 0 & 1 \end{bmatrix}}_{\mathbf{F}} \mathbf{x}_{(k-1)} + \underbrace{\begin{bmatrix} \delta T^2 \\ \delta T \end{bmatrix}}_{\mathbf{G}} \mathbf{v}_{(k)}$$

$$\mathbf{z}_{(k)} = \underbrace{\begin{bmatrix} \frac{2}{c} & 0 \end{bmatrix}}_{\mathbf{H}} \mathbf{x}_{(k)} + \mathbf{w}_{(k)}$$

where δT is sampling time, the state vector $\mathbf{x} = [h \; \dot{h}]^\top$ is composed of height h and velocity $\dot{h}$; the process noise $\mathbf{v}$ is a scalar gaussian process with covariance $\mathbf{Q}$[28], the measurement noise $\mathbf{w}$ is given by the covariance matrix $\mathbf{R}$.[29]

[28] Modelled as noise in acceleration—hence the quadratics time dependence when adding to position-state.

[29] Note: We can find $\mathbf{R}$ either statistically or use values from a datasheet.

LKF - Linear Navigation Problem (2)

Example - Planet Lander: Simulation model

A non-linear simulation model in MATLAB was created to generate the true state values and corresponding noisy observation:

1. First, we simulate motion in a thin atmosphere (small drag) and vehicle accelerates.

2. Second, as the density increases the vehicle decelerates to reach quasi-steady terminal velocity fall.

- The true σ_Q^2 of the process noise and the σ_R^2 of the measurement noise are set to different numbers than those used in our linear model.[30]

- Simple Euler integration for the true motion is used (velocity $\rightarrow$ height).

[30] Note: we can try to change these settings and observe what happens if the model and the real world are too different.

LKF - Linear Navigation Problem (3)

Example - Planet Lander: Controller model

The vehicle controller has two features implemented:

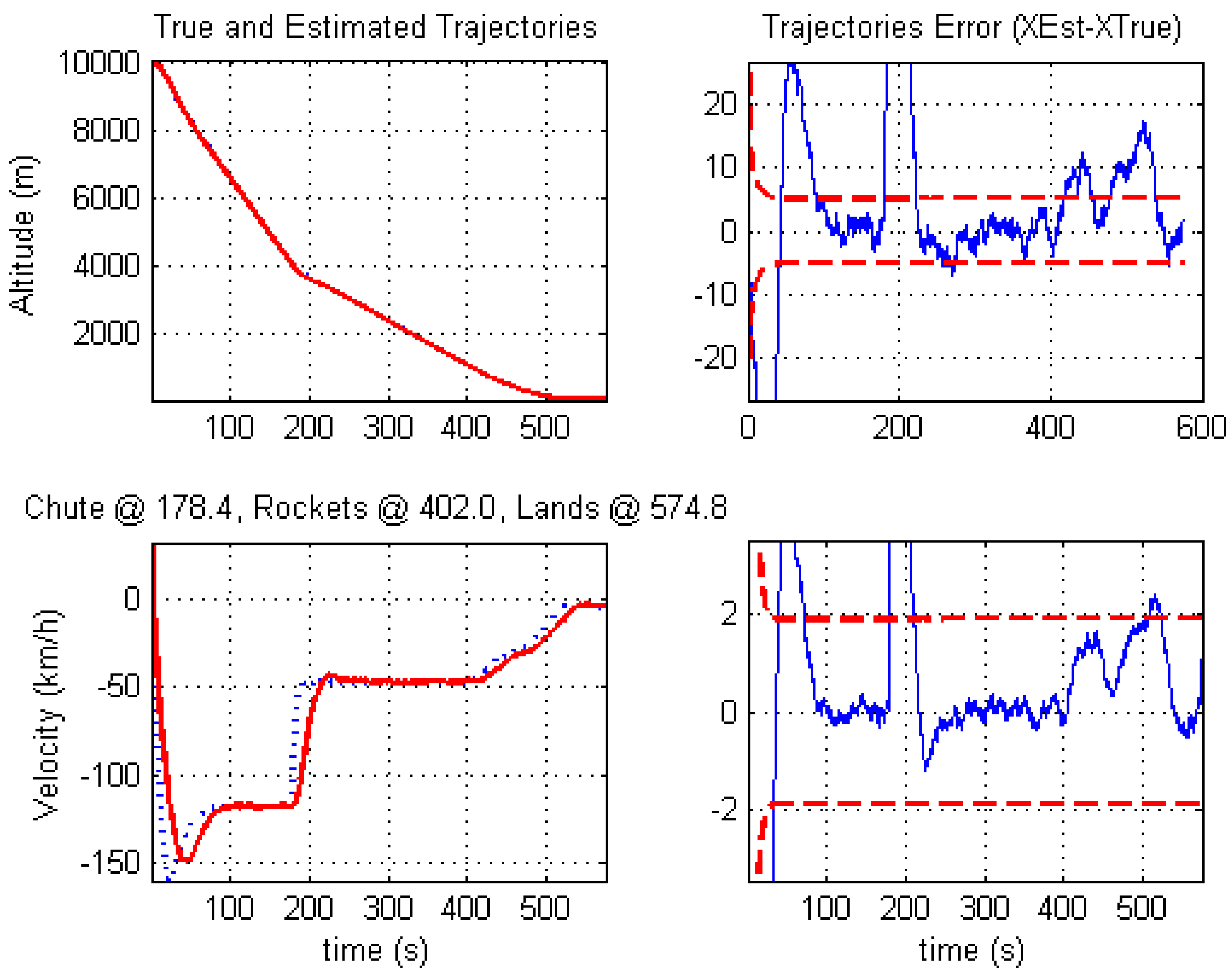
1. When the vehicle descends below a first given altitude threshold, it deploys a parachute (to increase the aerodynamic drag).
2. When the vehicle descends below a second given altitude threshold, it fires rocket burners to slow the descend and land safely.

- The controller operates only on the estimated quantities.
- Firing the rockets also destroys the parachute.

LKF - Linear Navigation Problem (4)

Example - Results for: $\sigma_{\mathrm{R}}^{\mathrm{model}} = 1.1\sigma_{\mathrm{R}}^{\mathrm{true}}$, $\sigma_{\mathrm{Q}}^{\mathrm{model}} = 1.1\sigma_{\mathrm{Q}}^{\mathrm{true}}$

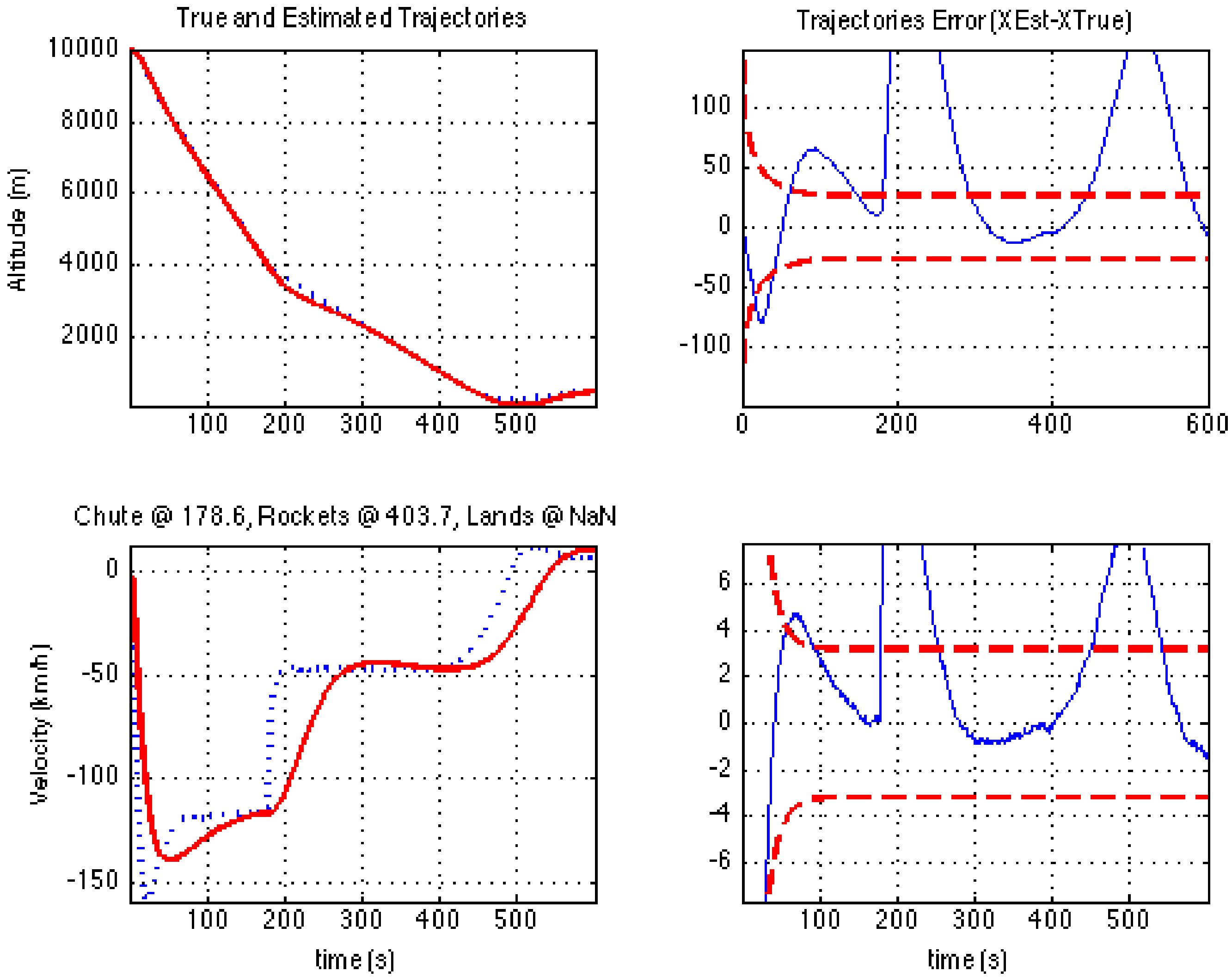
We did good modeling, errors are due to the non-linear world!



LKF - Linear Navigation Problem (5)

Example - Results for: $\boldsymbol{\sigma_{\mathrm{R}}^{\mathrm{model}} = 10\sigma_{\mathrm{R}}^{\mathrm{true}}, \sigma_{\mathrm{Q}}^{\mathrm{model}} = 1.1\sigma_{\mathrm{Q}}^{\mathrm{true}}}$

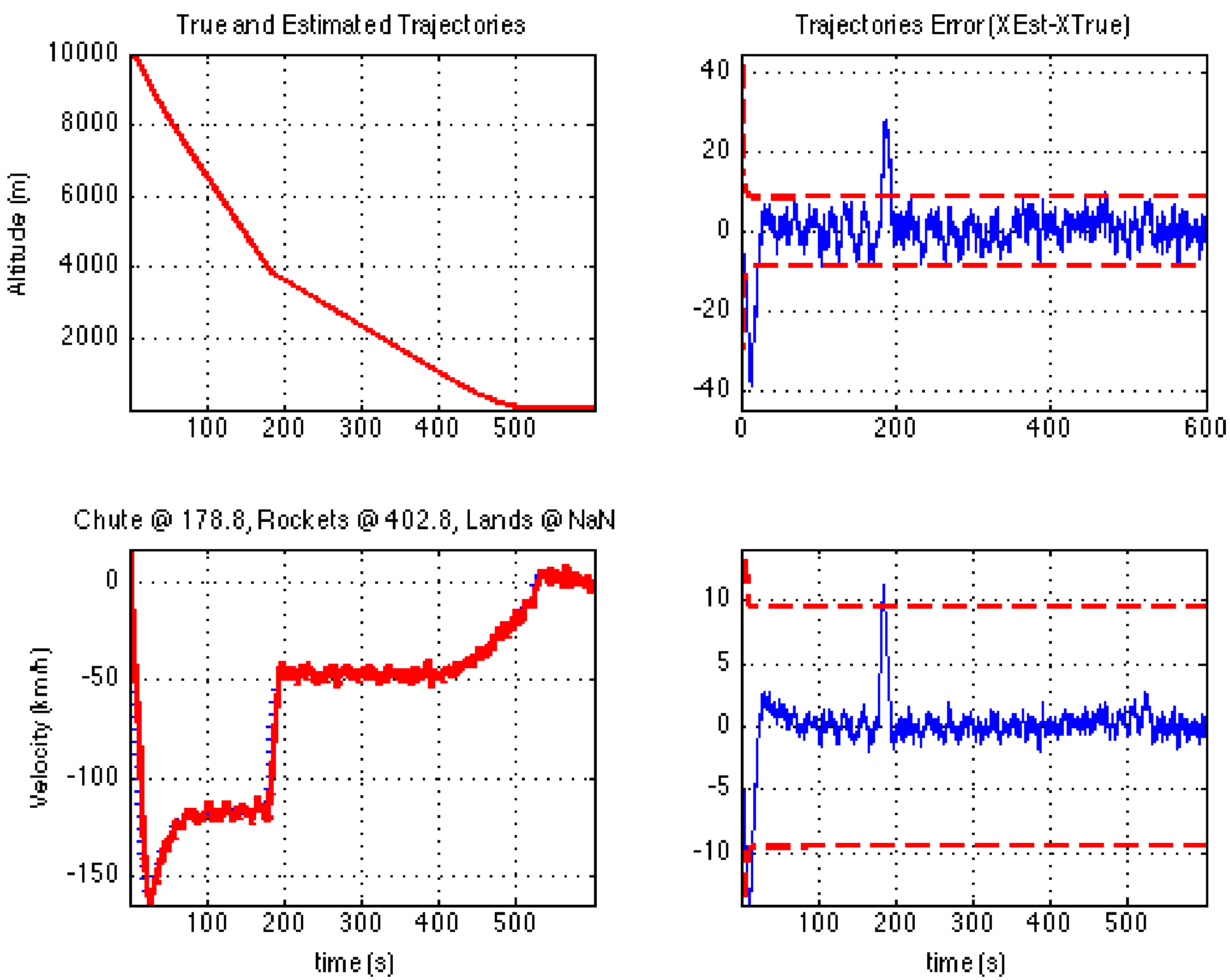
We do not trust the measurements, the good linear model alone is not enough!



LKF - Linear Navigation Problem (6)

Example - Results for: $\sigma_R^{model} = 1.1\sigma_R^{true}$, $\sigma_Q^{model} = 10\sigma_Q^{true}$

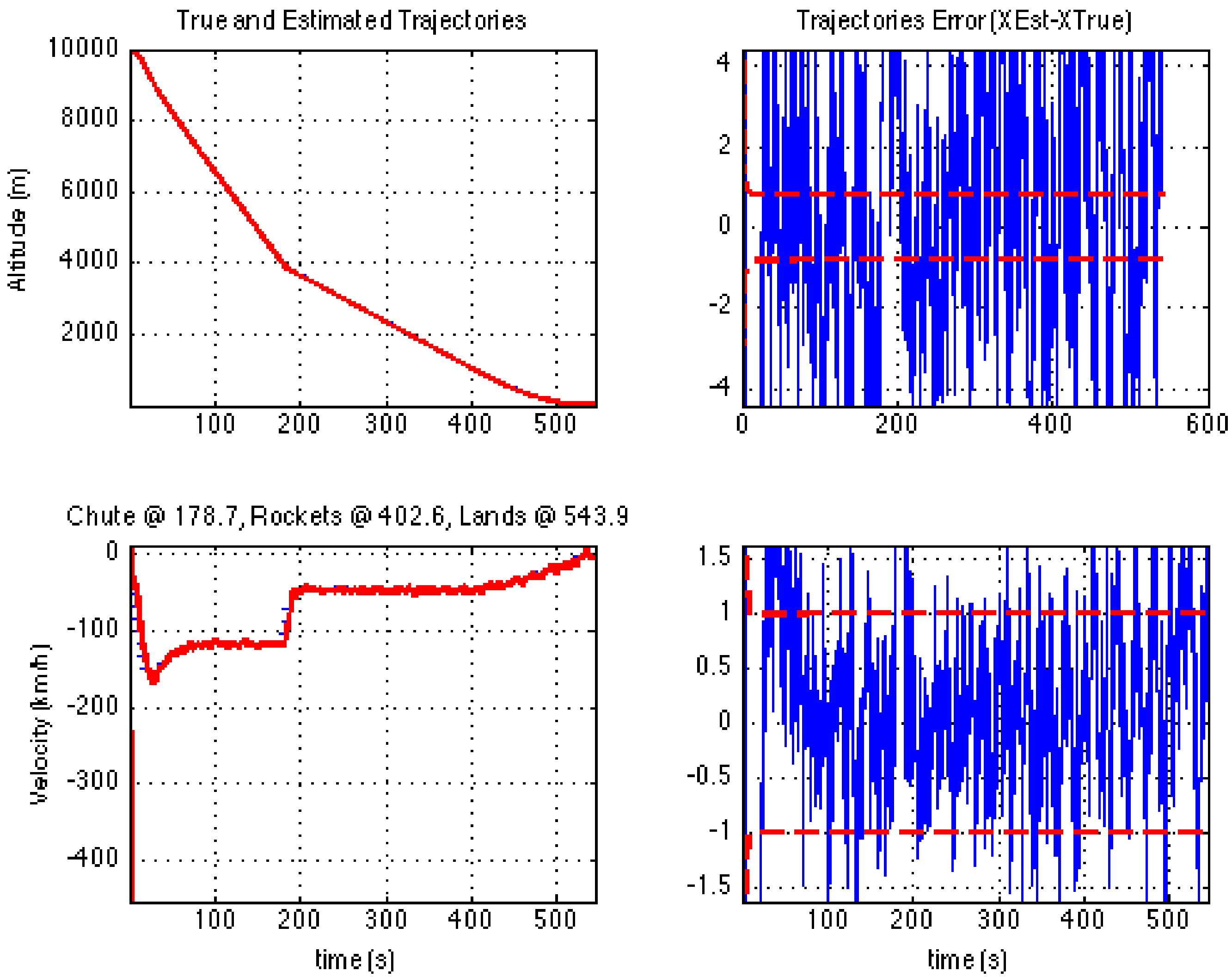
We do not trust our model, the estimates have good mean but are too noisy!



LKF - Linear Navigation Problem (7)

Example - Results for: $\boldsymbol{\sigma_R^{\text{model}} = 0.1\sigma_R^{\text{true}},\ \sigma_Q^{\text{model}} = 1.1\sigma_Q^{\text{true}}}$

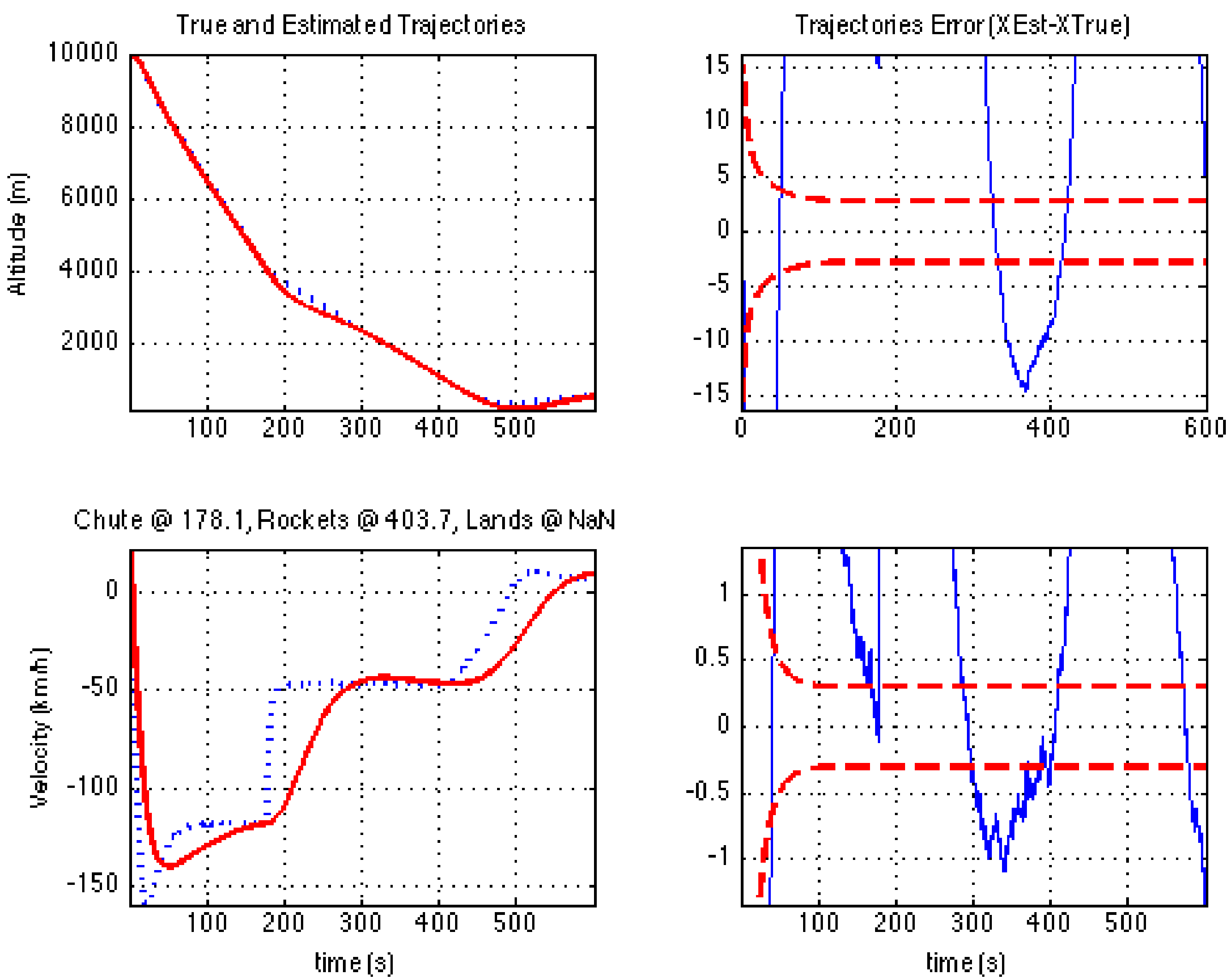
We are overconfident measurements—fortunately, the sensor is not more noisy!



LKF - Linear Navigation Problem (8)

Example - Results for: $\sigma_R^{model} = 1.1\sigma_R^{true}$, $\sigma_Q^{model} = 0.1\sigma_Q^{true}$

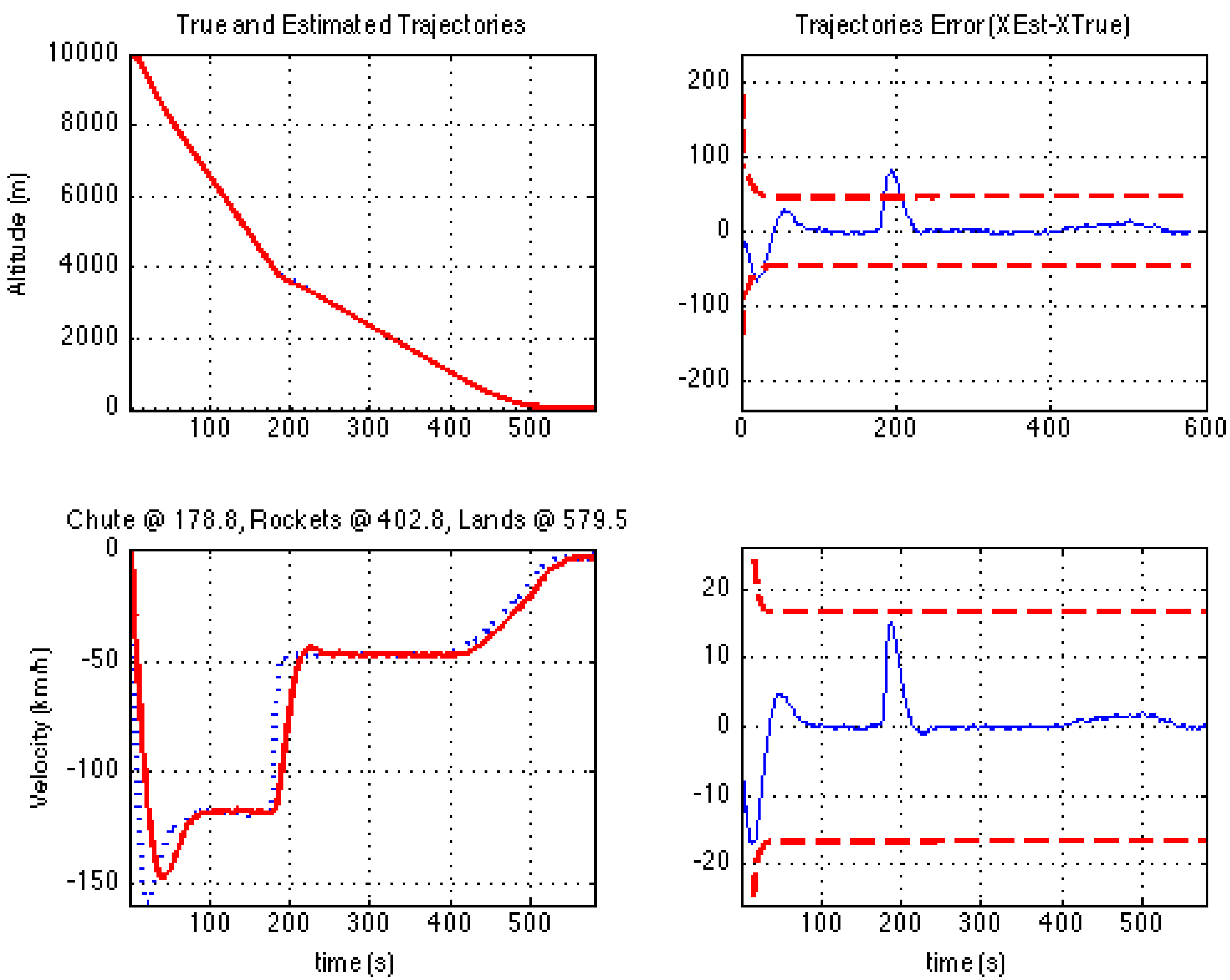
We are overconfident in our model, but the world is really not linear ...



LKF - Linear Navigation Problem (9)

Example - Results for: $\sigma_R^{model} = 10\sigma_R^{true}$, $\sigma_Q^{model} = 10\sigma_Q^{true}$

We do neither trust the model nor measurements, we cope with the nonlinearities.



From LKF to EKF

- Linear models in the non-linear environment $\rightarrow$ **BAD**.
- Non-linear models in the non-linear environment $\rightarrow$ **BETTER**.
- Assume the following the non-linear system model function $\mathbf{f}(\mathbf{x})$ and the non-linear measurement function $\mathbf{h}(\mathbf{x})$, we can reformulate:

$$\mathbf{x}_{(k)} = \mathbf{f}(\mathbf{x}_{(k-1)}, \mathbf{u}_{(k)}, k) + \mathbf{v}_{(k)}$$

$$\mathbf{z}_{(k)} = \mathbf{h}(\mathbf{x}_{(k)}, \mathbf{u}_{(k)}, k) + \mathbf{w}_{(k)}$$

EKF - Non-linear Prediction

Without proof[31]: The main idea behind EKF is to linearize the non-linear model around the „best“ current estimate[32].

This is realized using a Taylor series expansion[33].

Assume an estimate $\hat{\mathbf{x}}_{(k-1|k-1)}$ then

$$\mathbf{x}_{(k)} \approx \mathbf{f}(\hat{\mathbf{x}}_{(k-1|k-1)}, \mathbf{u}_{(k)}, k) + \nabla\mathbf{F}_{\mathbf{x}}[\mathbf{x}_{(k-1)} - \hat{\mathbf{x}}_{(k-1|k-1)}] + \cdots + \mathbf{v}_{(k)}$$

where the term $\nabla\mathbf{F}_{\mathbf{x}}$ is a Jacobian of $\mathbf{f}(\mathbf{x})$ w.r.t. $\mathbf{x}$ evaluated at $\hat{\mathbf{x}}_{(k-1|k-1)}$:

$$\nabla\mathbf{F}_{\mathbf{x}} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_1} & \cdots & \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_m} \\ \vdots & & \vdots \\ \frac{\partial \mathbf{f}_n}{\partial \mathbf{x}_1} & \cdots & \frac{\partial \mathbf{f}_n}{\partial \mathbf{x}_m} \end{bmatrix}$$

[31] See reference [1] pages 39-41

[32] Note: the „best“ meaning the prediction at $(k|k-1)$ or the last estimate at $(k-1|k-1)$

[33] Note: recall the non-linear LSQ problem of LBL navigation

EKF - Non-linear Observation

Without proof[34]: The same holds for the observation model, i.e. the predicted observation $\mathbf{z}_{(k|k-1)}$ is the projection of $\hat{\mathbf{x}}_{(k|k-1)}$ through the non-linear measurement model[35].

Hence, assume an estimate $\hat{\mathbf{x}}_{(k|k-1)}$ then

$$\mathbf{z}_{(k)} \approx \mathbf{h}(\hat{\mathbf{x}}_{(k|k-1)}, \mathbf{u}_{(k)}, k) + \nabla\mathbf{H}_{\mathbf{x}}[\hat{\mathbf{x}}_{(k|k-1)} - \mathbf{x}_{(k)}] + \cdots + \mathbf{w}_{(k)}$$

where the term $\nabla\mathbf{H}_{\mathbf{x}}$ is a Jacobian of $\mathbf{h}(\mathbf{x})$ w.r.t. $\mathbf{x}$ evaluated at $\hat{\mathbf{x}}_{(k|k-1)}$:

$$\nabla\mathbf{H}_{\mathbf{x}} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{h}_1}{\partial \mathbf{x}_1} & \cdots & \frac{\partial \mathbf{h}_1}{\partial \mathbf{x}_m} \\ \vdots & & \vdots \\ \frac{\partial \mathbf{h}_n}{\partial \mathbf{x}_1} & \cdots & \frac{\partial \mathbf{h}_n}{\partial \mathbf{x}_m} \end{bmatrix}$$

[34] See reference [1] pages 41-43

[35] Note: for the LKF it was given by $\mathbf{H}\hat{\mathbf{x}}_{(k|k-1)}$

EKF - Algorithm (1)

Prediction:

$$\underbrace{\hat{\mathbf{x}}(k|k-1)}_{\text{predicted state}} = \overbrace{\mathbf{f}(\underbrace{\hat{\mathbf{x}}(k-1|k-1)}_{\text{old state est}}, \underbrace{\mathbf{u}(k)}_{\text{control}}, k)}^{\text{plant model}}$$

$$\underbrace{\mathbf{P}(k|k-1)}_{\text{predicted covariance}} = \nabla\mathbf{F}_{\mathbf{x}} \underbrace{\mathbf{P}(k-1|k-1)}_{\text{old est covariance}} \nabla\mathbf{F}_{\mathbf{x}}{}^{T} + \underbrace{\nabla\mathbf{G}_{\mathbf{v}}\mathbf{Q}\nabla\mathbf{G}_{\mathbf{v}}{}^{T}}_{\text{process noise}}$$

$$\underbrace{\mathbf{z}(k|k-1)}_{\text{predicted obs}} = \overbrace{\mathbf{h}(\hat{\mathbf{x}}(k|k-1))}^{\text{observation model}}$$

EKF - Algorithm (2)

Update:

$$\underbrace{\hat{\mathbf{x}}(k|k)}_{\text{new state estimate}} = \overbrace{\hat{\mathbf{x}}(k|k-1) + \mathbf{W}\ \underbrace{\nu(k)}_{\text{innovation}}}^{\text{prediction and correction}}$$

$$\underbrace{\mathbf{P}(k|k)}_{\text{new covariance estimate}} = \underbrace{\mathbf{P}(k|k-1) - \mathbf{W}\mathbf{S}\mathbf{W}^T}_{\text{update decreases uncertainty}}$$

where

$$\nu(k) = \overbrace{\mathbf{z}(k)}^{\text{measurement}} - \mathbf{z}(k|k-1)$$

$$\mathbf{W} = \underbrace{\mathbf{P}(k|k-1)\nabla\mathbf{H}_{\mathbf{x}}{}^T\mathbf{S}^{-1}}_{\text{kalman gain}}$$

$$\mathbf{S} = \underbrace{\nabla\mathbf{H}_{\mathbf{x}}\mathbf{P}(k|k-1)\nabla\mathbf{H}_{\mathbf{x}}{}^T + \mathbf{R}}_{\text{Innovation Covariance}}$$

$$\nabla\mathbf{F}_{\mathbf{x}} = \frac{\partial\mathbf{f}}{\partial\mathbf{x}} = \underbrace{\begin{bmatrix} \frac{\partial\mathbf{f}_1}{\partial\mathbf{x}_1} & \cdots & \frac{\partial\mathbf{f}_1}{\partial\mathbf{x}_m} \\ \vdots & & \vdots \\ \frac{\partial\mathbf{f}_n}{\partial\mathbf{x}_1} & \cdots & \frac{\partial\mathbf{f}_n}{\partial\mathbf{x}_m} \end{bmatrix}}_{\text{evaluated at } \hat{\mathbf{x}}(k-1|k-1)} \quad \nabla\mathbf{H}_{\mathbf{x}} = \frac{\partial\mathbf{h}}{\partial\mathbf{x}} = \underbrace{\begin{bmatrix} \frac{\partial\mathbf{f}_1}{\partial\mathbf{x}_1} & \cdots & \frac{\partial\mathbf{h}_1}{\partial\mathbf{x}_m} \\ \vdots & & \vdots \\ \frac{\partial\mathbf{h}_n}{\partial\mathbf{x}_1} & \cdots & \frac{\partial\mathbf{h}_n}{\partial\mathbf{x}_m} \end{bmatrix}}_{\text{evaluated at } \hat{\mathbf{x}}(k|k-1)}$$

Source: [1] P. Newman, EKF Based Navigation and SLAM, SLAM Summer School 2006

EKF - Features & Maps

Assumption: The world is represented by a set of discrete landmarks (features) whose location / orientation and geometry can by described by a set of discrete parameters $\rightarrow$ concatenated into a feature vector called Map:

$$\mathbf{M} = \begin{bmatrix} \mathbf{x}_{\mathbf{f},1} \\ \mathbf{x}_{\mathbf{f},2} \\ \mathbf{x}_{\mathbf{f},3} \\ \vdots \\ \mathbf{x}_{\mathbf{f},n} \end{bmatrix}$$

Examples of features in 2D world:

- absolute observation: given by the position coordinates of the landmarks in the global reference frame: $\mathbf{x}_{\mathbf{f},i} = [x_i \; y_i]^\top$ (*e.g., measured by GPS*)
- relative observation: given by the radius and bearing to landmark: $\mathbf{x}_{\mathbf{f},i} = [r_i \; \theta_i]^\top$ (*e.g., measured by visual odometry, laser mapping, sonar*)

EKF - Localization

Assumption: we are given a map $\mathbf{M}$ and a sequence of vehicle-relative[36] observations $\mathbf{Z^k}$ described by likelihood $p(\mathbf{Z^k}|\mathbf{M}, \mathbf{x}_v)$.

Task: to estimate the *pdf* for the vehicle pose $p(\mathbf{x}_v|\mathbf{M}, \mathbf{Z^k})$.

$$p(\mathbf{x}_v|\mathbf{M}, \mathbf{Z^k}) = \frac{p(\mathbf{x}_v, \mathbf{M}, \mathbf{Z^k})}{p(\mathbf{M}, \mathbf{Z^k})} = \frac{p(\mathbf{Z^k}|\mathbf{M}, \mathbf{x}_v) \times p(\mathbf{M}, \mathbf{x}_v)}{p(\mathbf{Z^k}|\mathbf{M}) \times p(\mathbf{M})} =$$

$$= \frac{p(\mathbf{Z^k}|\mathbf{M}, \mathbf{x}_v) \times p(\mathbf{x}_v|\mathbf{M}) \times p(\mathbf{M})}{\int_{-\infty}^{+\infty} p(\mathbf{Z^k}|\mathbf{M}, \mathbf{x}_v) p(\mathbf{x}_v|\mathbf{M})\, dx_v \times p(\mathbf{M})} = \frac{p(\mathbf{Z^k}|\mathbf{M}, \mathbf{x}_v) \times p(\mathbf{x}_v|\mathbf{M})}{\text{normalising constant}}$$

Solution: $p(\mathbf{x}_v|\mathbf{M})$ is just another sensor $\rightarrow$ the *pdf* of locating the robot when observing a given map.

[36] Note: Vehicle-relative observations are such kind of measurements that involve sensing the relationship between the vehicle and its surroundings—the map, e.g. measuring the angle and distance to a feature.

EKF - Mapping

Assumption: we are given a vehicle location $\mathbf{x}_v$, [37] and a sequence of vehicle-relative observations $\mathbf{Z}^\mathbf{k}$ described by likelihood $p(\mathbf{Z}^\mathbf{k}|\mathbf{M}, \mathbf{x}_v)$.

Task: to estimate the *pdf* of the map $p(\mathbf{M}|\mathbf{Z}^\mathbf{k}, \mathbf{x}_v)$.

$$p(\mathbf{M}|\mathbf{Z}^\mathbf{k}, \mathbf{x}_v) = \frac{p(\mathbf{x}_v, \mathbf{M}, \mathbf{Z}^\mathbf{k})}{p(\mathbf{Z}^\mathbf{k}, \mathbf{x}_v)} = \frac{p(\mathbf{Z}^\mathbf{k}|\mathbf{M}, \mathbf{x}_v) \times p(\mathbf{M}, \mathbf{x}_v)}{p(\mathbf{Z}^\mathbf{k}|\mathbf{x}_v) \times p(\mathbf{x}_v)} =$$

$$= \frac{p(\mathbf{Z}^\mathbf{k}|\mathbf{M}, \mathbf{x}_v) \times p(\mathbf{M}|\mathbf{x}_v) \times p(\mathbf{x}_v)}{\int_{-\infty}^{+\infty} p(\mathbf{Z}^\mathbf{k}|\mathbf{M}, \mathbf{x}_v) p(\mathbf{M}|\mathbf{x}_\mathbf{v})\, dM \times p(\mathbf{x}_v)} = \frac{p(\mathbf{Z}^\mathbf{k}|\mathbf{M},\mathbf{x}_v) \times p(\mathbf{M}|\mathbf{x}_v)}{\text{normalising constant}}$$

Solution: $p(\mathbf{x}_v|\mathbf{M})$ is just another sensor $\rightarrow$ the *pdf* of observing the map at given robot location.

[37] Note: Ideally derived from absolute position measurements since position derived from relative measurements (e.g. odometry, integration of inertial measurements) is always subjected to a drift—so called dead reckoning

EKF - Simultaneous Localization and Mapping

If we parametrize the random vectors $\mathbf{x}_v$ and $\mathbf{M}$ with mean and variance then the (E)KF will compute the MMSE estimate of the posterior.

What is the SLAM and how can we achieve it?

- With no prior information about the map (and about the vehicle—no GPS),
- the SLAM is a navigation problem of building consistent estimate of both
- the environment (represented by the map—*the mapping*)
- and vehicle trajectory (6 DOF position and orientation—*the localization*),
- using only proprioceptive sensors (e.g., inertial, odometry),
- and vehicle-centric sensors (e.g., radar, camera, laser, sonar etc.).

Example - EKF-SLAM

The naive EKF-SLAM—the map is taken as additional sensor and **ALL** the features are included in the state vector (information captured in $\mathbf{P}$).

What are the EKF-SLAM characteristics?

- The naive version does not work, especially in 3D and for large areas!
- Large computational load (the update of the covariance matrix $\mathbf{P}$ proportional at best to the square of the number of features)!

How can we make the EKF-SLAM work?

- Feature management—ideally decoupled solution or more solutions together (laser-based mapping, vision-based mapping)
- Loop closures—save the history of observations and if the same place visited again, re-compute both map and trajectory (estimators called „smoothers“).

Example - Real-world EKF architecture