From Bayes to Extended Kalman Filter

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: V. Hlavac — Introduction to probability theory and P. Newman — SLAM Summer School 2006, Oxford

Outline of the lecture:

- Overview: From MAP to RBE
- Overview: From LSQ to NLSQ
- 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

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

where $\boxed{\text{P(B|A)}}$ is the posterior probability and $\boxed{\text{P(A|B)}}$ is the likelihood.

- This is a fundamental rule for machine learning (pattern recognition) as it allows to compute the probability of an output B given measurements A.
- The prior probability is $P(B)$ without any evidence from measurements.
- The likelihood $\boxed{\text{P(A|B)}}$ evaluates the measurements given an output B. Seeking the output that maximizes the likelihood (*the most likely output*) is known as the maximum likelihood estimation (ML).
- The posterior probability $\boxed{\text{P(B|A)}}$ is the probability of B after taking the measurement A into account. Its maximization leads to the maximum a-posteriori estimation (MAP).

Overview of the Probability Rules

- The Product rule: $P(A,B) = P(A|B)\,P(B) = P(B|A)\,P(A)$
- The Sum rule: $P(B) = \sum\limits_{A} P(A,B) = \sum\limits_{A} P(B|A)\,P(A)$
- 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:

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

- General inference:

$$P(V|S) = \frac{P(V,S)}{P(S)} = \frac{\sum\limits_{A,B,C} P(S,A,B,C,V)}{\sum\limits_{S,A,B,C} P(S,A,B,C,V)}$$

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

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

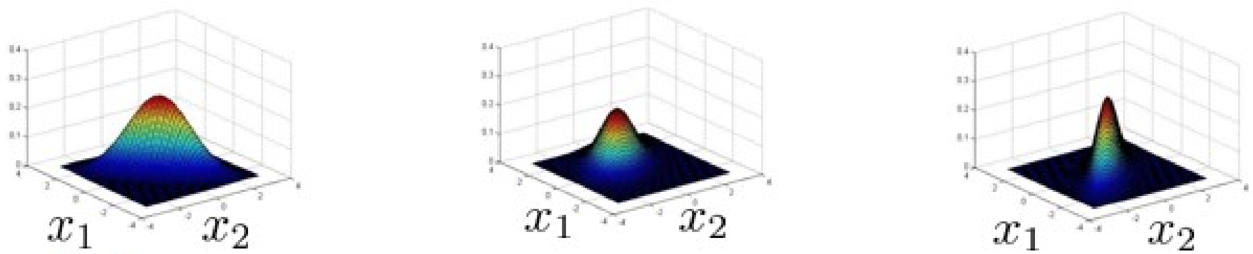
[1] Note: The covariance matrix is symmetric (i.e. $\Sigma = \Sigma^\top$) and positive-semidefinite (as the covariance matrix is real valued, the positive-semidefinite means that $x^\top M x \geq 0$ for all $x \in \mathbb{R}$).

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

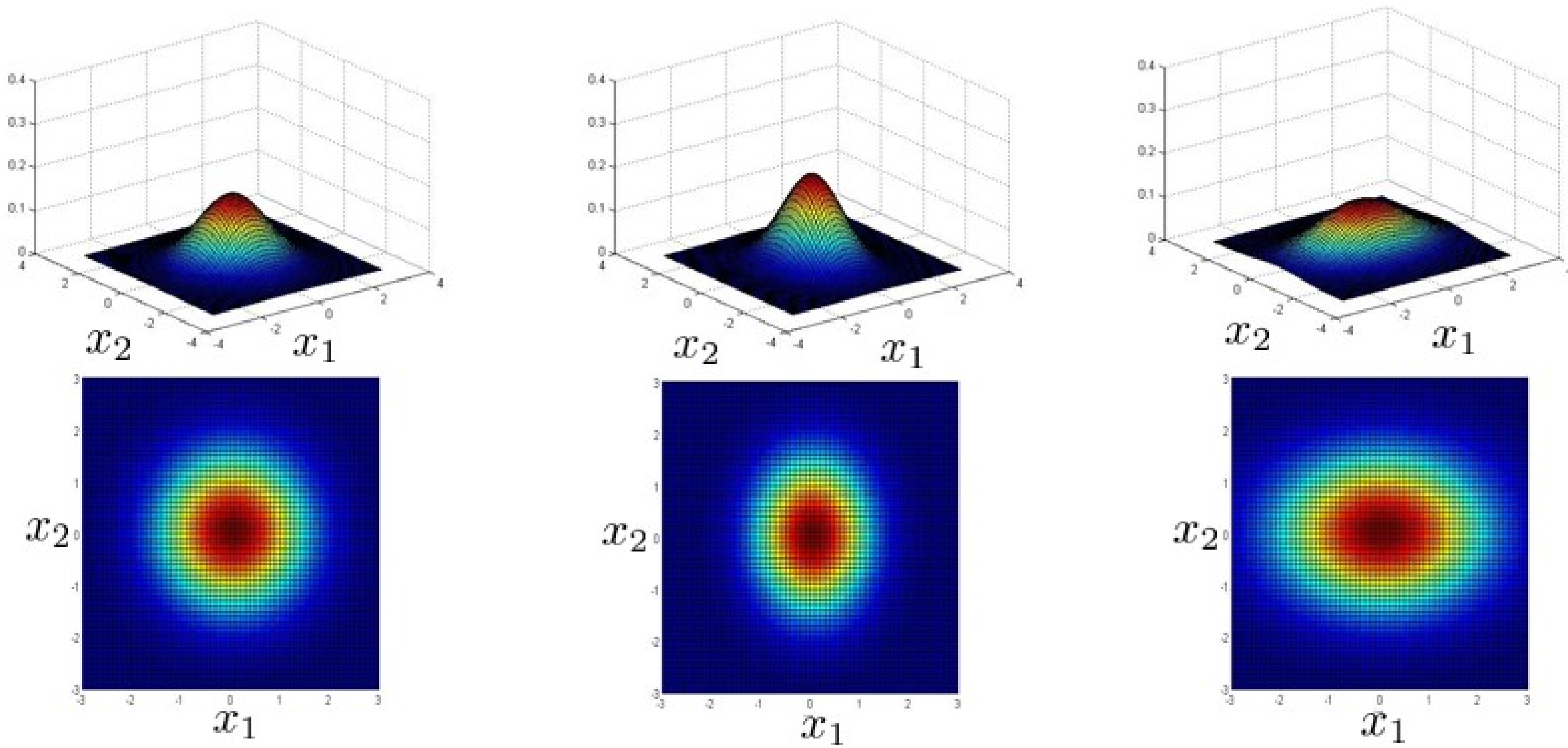
$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)} \qquad \Sigma = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)}-\mu)(x^{(i)}-\mu)^T$$

Andrew Ng

Multivariate Normal distribution (2)

Multivariate Gaussian (Normal) examples

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



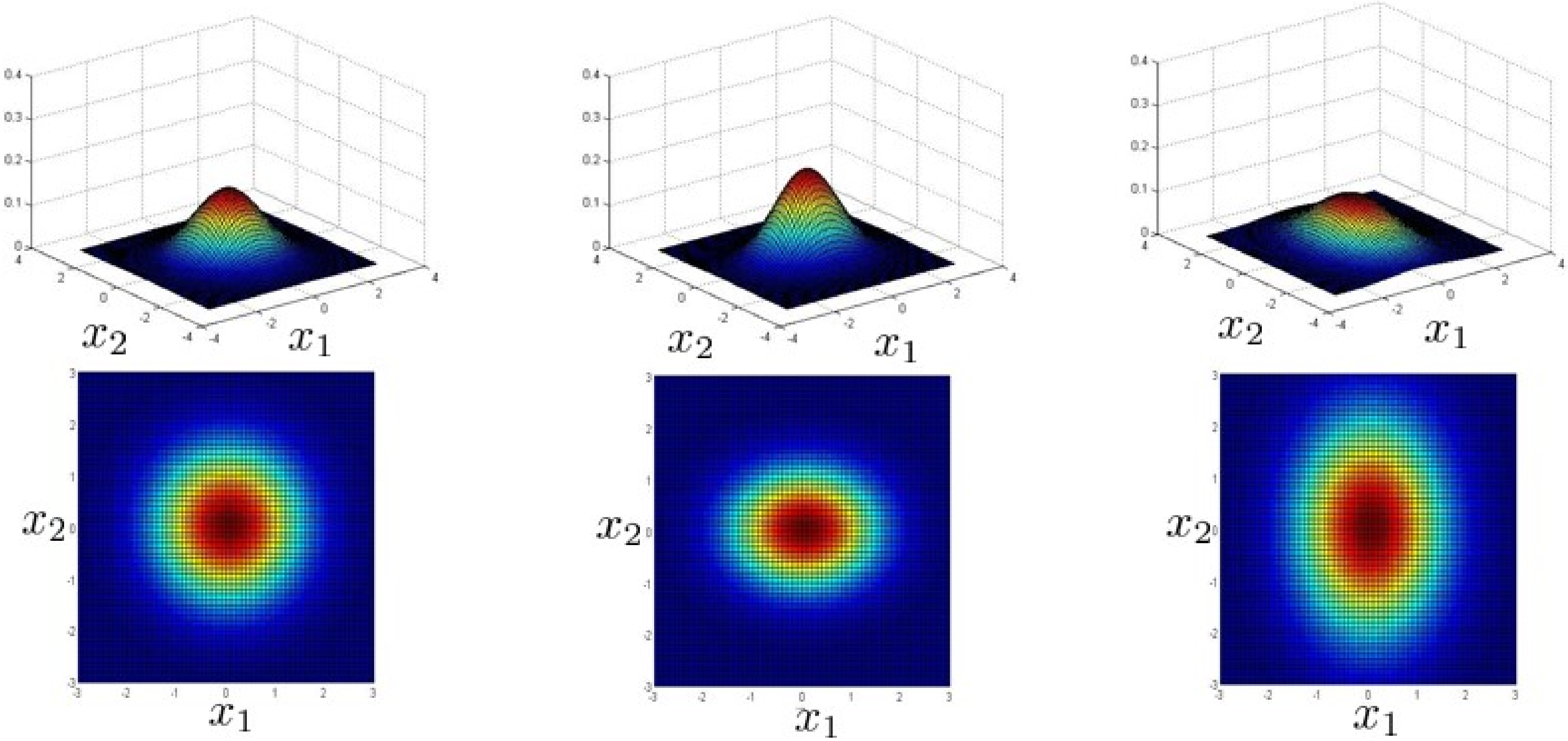
Andrew Ng

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

Multivariate Normal distribution (3)

Multivariate Gaussian (Normal) examples

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



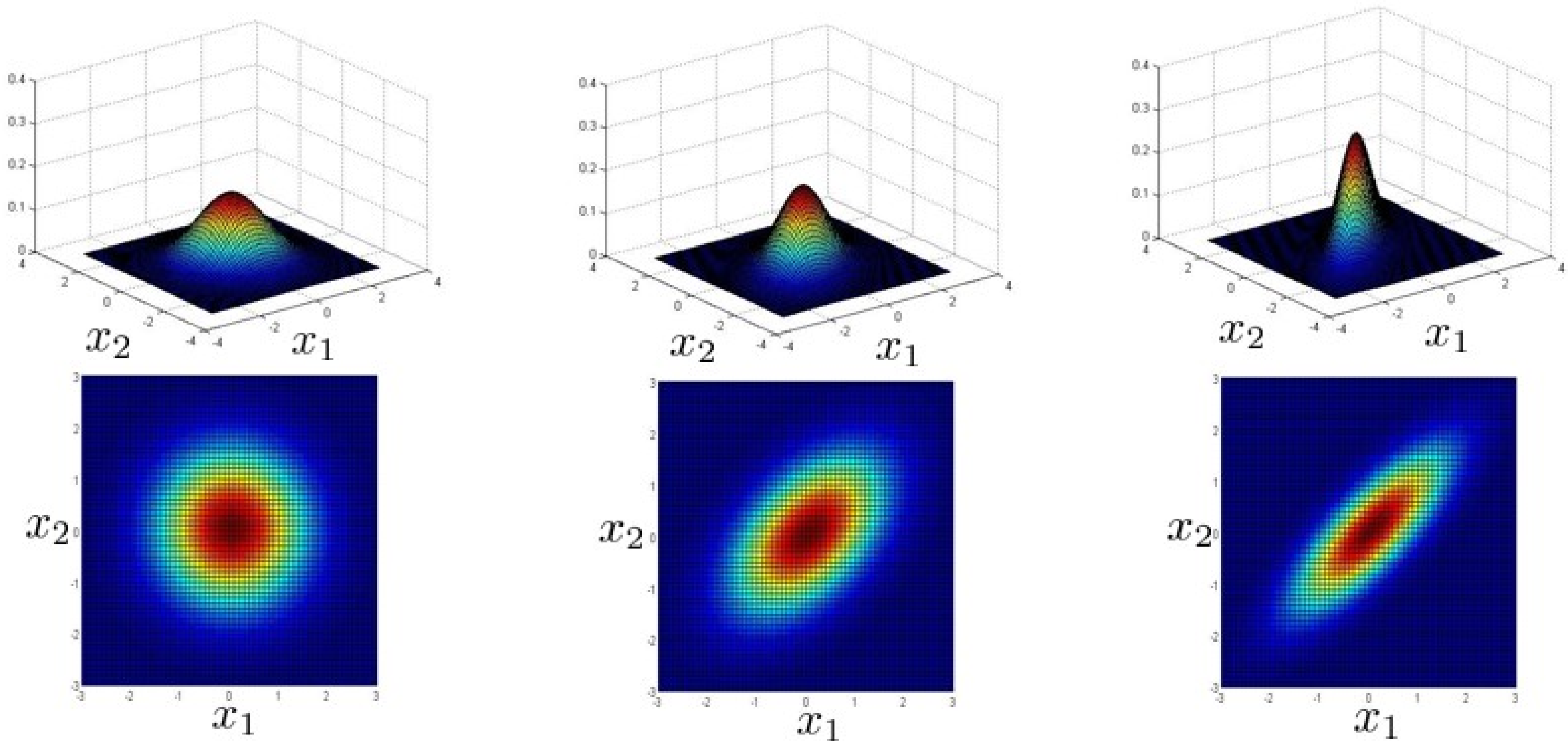
Andrew Ng

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

Multivariate Normal distribution (4)

Multivariate Gaussian (Normal) examples

$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix} \qquad \mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$



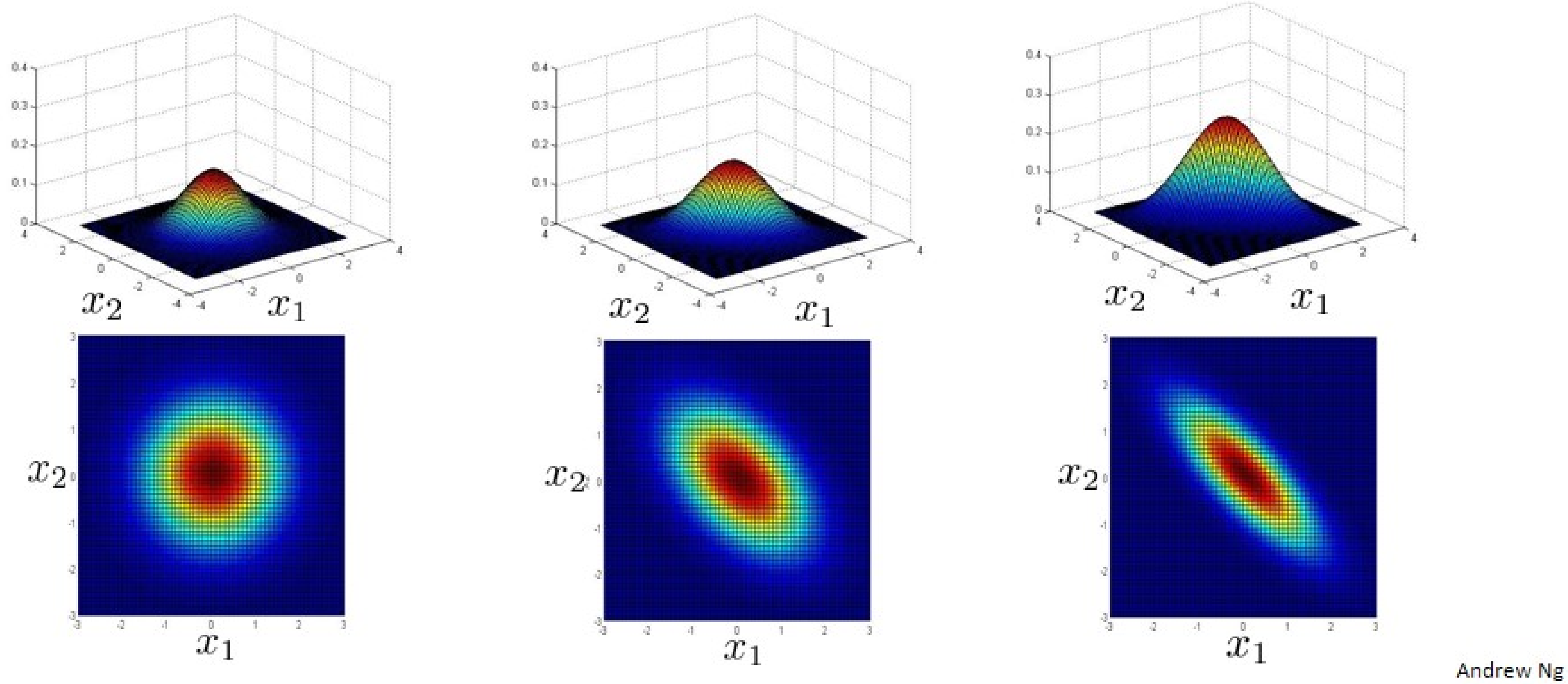
Andrew Ng

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

Multivariate Normal distribution (5)

Multivariate Gaussian (Normal) examples

$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix} \qquad \mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix}$$

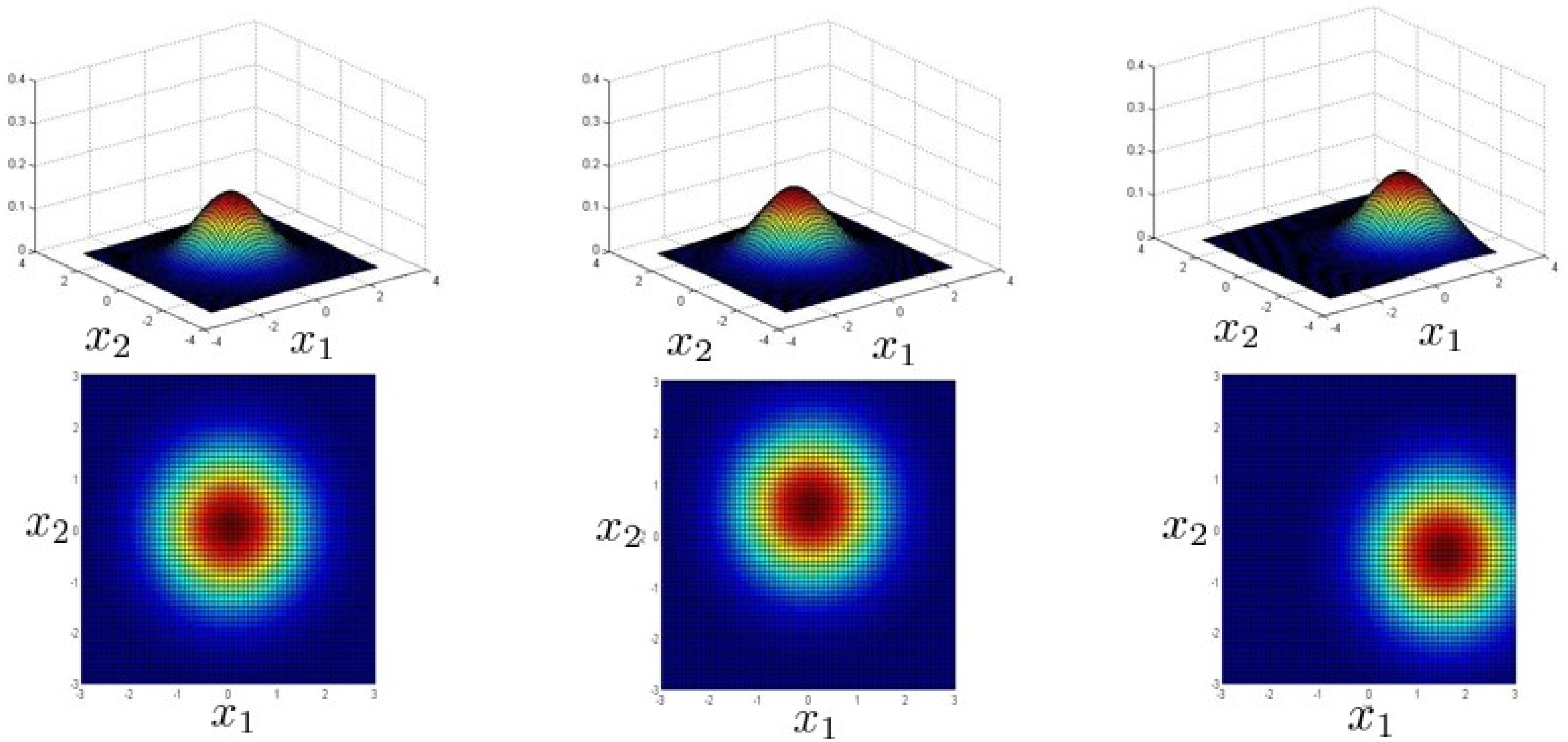


Source: Andrew Ng, Stanford University, Machine Learning course 2012, Lecture 15

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

MAP - Maximum A-Posteriori Estimation

- 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 $p(\mathbf{x}|\mathbf{z})$:

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

MMSE - Minimum Mean Squared Error

Without proof[2]: 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.[3]

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

Why is this important? The MMSE estimate given a set of measurements is the mean of that variable conditioned on the measurements! [4]

[2] See reference [1] pages 11-12

[3] Note: We minimize a scalar quantity.

[4] 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 the natural extension of MAP to time-stamped sequence of observations being processed at each time step. In RBE we use the priory estimate and current measurement to compute the posteriori estimate $\hat{\mathbf{x}}$.

- When the next measurement comes we use our previous posteriori estimate as a new prior and proceed with the same estimation rule.
- Hence for each time-step k we obtain an estimate for it's state given all observations up to that time (the set $\mathbf{Z^k}$).
- Using the Bayes rule and conditional independence of measurements ($\mathbf{z}_k$ being single measurement at time k):

$$p(\mathbf{x}, \mathbf{Z^k}) = p(\mathbf{x}|\mathbf{Z^k})p(\mathbf{Z^k}) = p(\mathbf{Z^k}|\mathbf{x})p(\mathbf{x}) = p(\mathbf{Z}^{k-1}|\mathbf{x})p(\mathbf{z}_k|\mathbf{x})p(\mathbf{x})$$

- We express $p(\mathbf{Z}^{k-1}|\mathbf{x})$ and substitute for it to get:

$$p(\mathbf{x}|\mathbf{Z^k}) = \frac{p(\mathbf{z}_k|\mathbf{x})p(\mathbf{x}|\mathbf{Z}^{k-1})p(\mathbf{Z}^{k-1})}{p(\mathbf{Z^k})}$$

RBE - Recursive Bayesian Estimation

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

Without proof[5]: 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*).

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

[5] 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

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

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[6]:

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

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

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

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

LSQ - Weighted Least Squares Estimation

If we have information about reliability of the measurements in $\mathbf{z}$, we can capture this as a covariance matrix $\mathbf{R}$ (diagonal terms only since the measurements are not correlated:

$$\mathbf{R} = \begin{bmatrix} \sigma_{z1}^2 & 0 & 0 \\ 0 & \sigma_{z2}^2 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

In the error vector $\mathbf{e}$ defined as $\mathbf{e} = \mathbf{Hx} - \mathbf{z}$ we can weight each its element by uncertainty in each element of the measurement vector $\mathbf{z}$, i.e. by $\mathbf{R}^{-1}$. The optimization criteria then becomes:

$$\hat{\mathbf{x}} = \underset{x}{\operatorname{argmin}} \, ||\mathbf{R}^{-1}(\mathbf{Hx} - \mathbf{z})||^2$$

Following the same derivation procedure, we obtain the weighted least squares:

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

LSQ - Least Squares Estimation

The world is non-linear → nonlinear model function $\mathbf{h}(\mathbf{x})$ → non-linear LSQ[7]:

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

- 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 $\mathbf{h}(\mathbf{x})$:

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

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

LSQ - Least Squares Estimation

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

1. Start with an initial guess $\hat{\mathbf{x}}$. [8]
2. Evaluate the LSQ expression for δ (update the $\nabla \mathbf{H}_{\hat{\mathbf{x}}}$ and substitute). [9]

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

3. Apply the δ correction to our initial estimate: $\hat{\mathbf{x}} := \hat{\mathbf{x}} + \delta$.[10]
4. Check for the stopping precision: if $||\mathbf{h}(\hat{\mathbf{x}}) - \mathbf{z}||^2 > \epsilon$ proceed with step (2) or stop otherwise.[11]

---

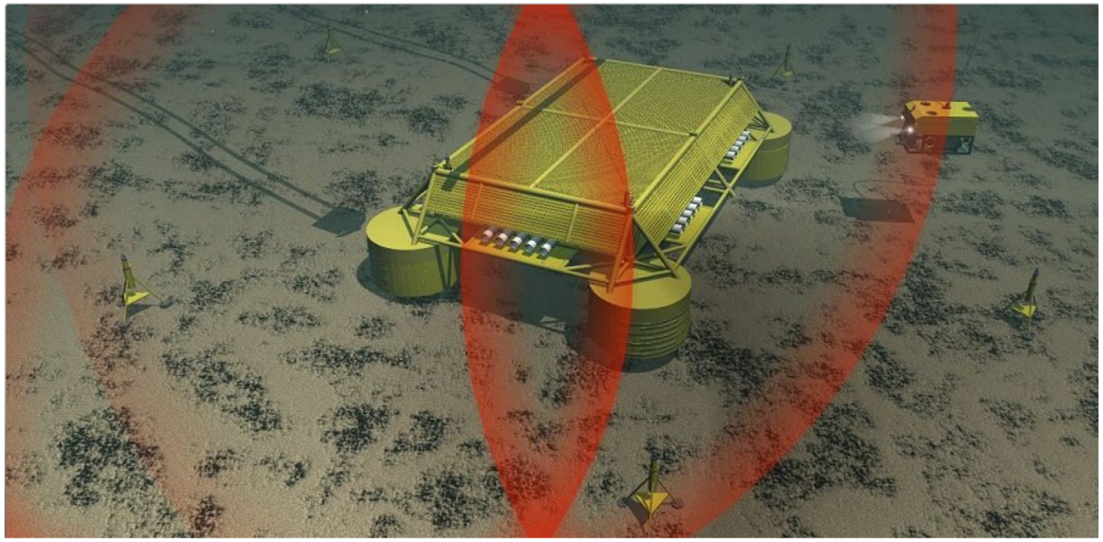
[8] Note: We can usually set to zero.

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

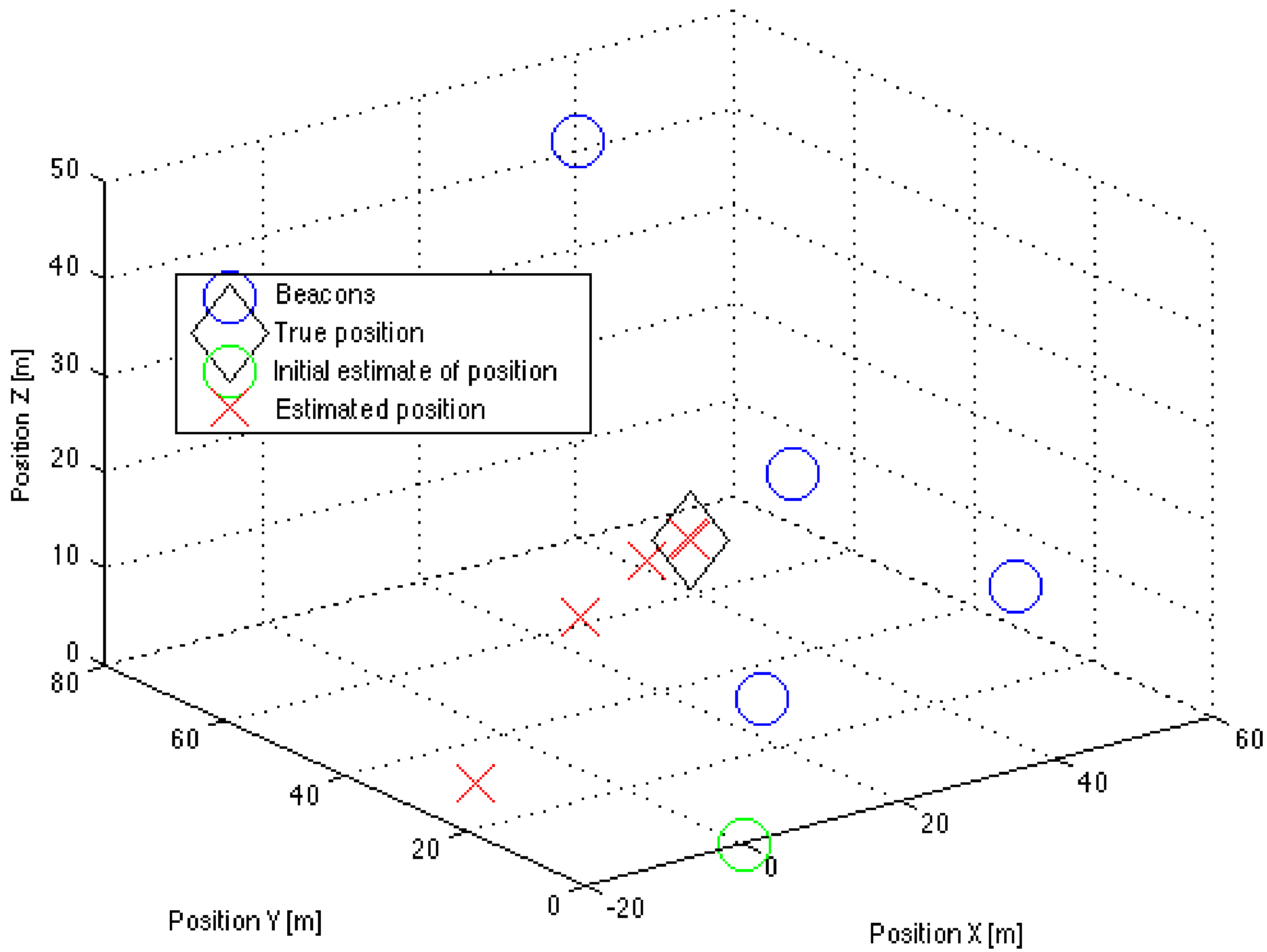
[10] Note: Due to these updates our initial guess should converge to such $\hat{\mathbf{x}}$ that minimizes the $||\mathbf{h}(\hat{\mathbf{x}}) - \mathbf{z}||^2$

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

Example - Long Base-line Navigation SONARDYNE

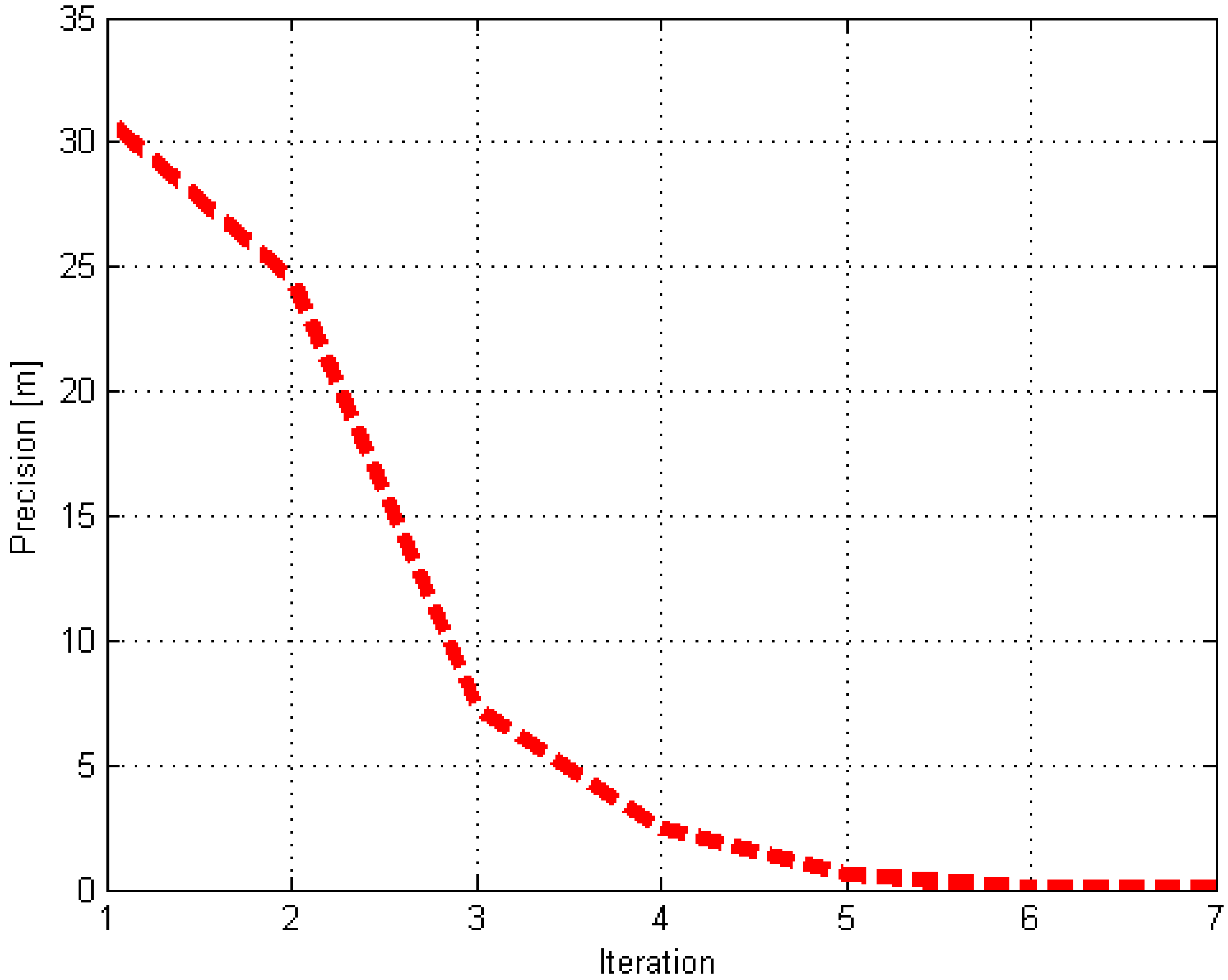


Example - Long Base-line Navigation



LSQ - Least Squares Estimation

Example - Long Base-line Navigation



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[12]:

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

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.

[12] 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[13], 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[14]
- 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*

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

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

LKF - The proof?

For the proof see reference [1] pages 22-26

.

We can figure out the new mean $\mathbf{x}_\oplus$ and covariance $\mathbf{P}_\oplus$ by expanding expression 3.9 and comparing terms with expression 3.10. Remember we want to find the new mean because Equation 1.14 tells us this will be the MMSE estimate. So expanding 3.9 we have:

$$\mathbf{x}^T\mathbf{P}_\ominus^{-1}\mathbf{x}-\mathbf{x}_\ominus^T\mathbf{P}_\ominus^{-1}\mathbf{x}_\ominus-\mathbf{x}^T\mathbf{P}_\ominus^{-1}\mathbf{x}_\ominus+\mathbf{x}_\ominus^T\mathbf{P}_\ominus^{-1}\mathbf{x}_\ominus+\mathbf{x}^T\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{x}-\mathbf{x}^T\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z}-\mathbf{z}\mathbf{R}^{-1}\mathbf{H}\mathbf{x}+\mathbf{z}^T\mathbf{R}^{-1}\mathbf{z} \tag{3.11}$$

Now collecting terms this becomes:

$$\mathbf{x}^T(\mathbf{P}_\ominus^{-1}+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})\mathbf{x}-\mathbf{x}^T(\mathbf{P}_\ominus^{-1}\mathbf{x}_\ominus+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z})-(\mathbf{x}^T\mathbf{P}_\ominus^{-1}+\mathbf{z}\mathbf{R}^{-1}\mathbf{H})\mathbf{x}+(\mathbf{x}_\ominus^T\mathbf{P}_\ominus^{-1}\mathbf{x}_\ominus+\mathbf{z}^T\mathbf{R}^{-1}\mathbf{z}) \tag{3.12}$$

Expanding 3.10:

$$\mathbf{x}^T\mathbf{P}_\oplus^{-1}\mathbf{x}-\mathbf{x}^T\mathbf{P}_\oplus^{-1}\mathbf{x}_\oplus-\mathbf{x}_\oplus^T\mathbf{P}_\oplus^{-1}\mathbf{x}+\mathbf{x}_\oplus^T\mathbf{P}_\oplus^{-1}\mathbf{x}_\oplus. \tag{3.13}$$

Comparing first terms in 3.12 and 3.13 we immediately see that

$$\mathbf{P}_\oplus=(\mathbf{P}_\ominus^{-1}+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}. \tag{3.14}$$

Comparing the second terms we see that:

$$\mathbf{P}_\oplus^{-1}\mathbf{x}_\oplus=\mathbf{P}_\ominus^{-1}\mathbf{x}_\ominus+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z}. \tag{3.15}$$

Therefore we can write the MMSE estimate, $\mathbf{x}_\oplus$ as

$$\mathbf{x}_\oplus=(\mathbf{P}_\ominus^{-1}+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}(\mathbf{P}_\ominus^{-1}\mathbf{x}_\ominus+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z}). \tag{3.16}$$

We can combine this result with our understanding of the recursive Bayesian filter we covered in section 1.6. Every time a new measurement becomes available we update our estimate and its covariance using the above two equations.

There is something about the above two equations 3.14 and 3.16 that may make them inconvenient — we have to keep inverting our prior covariance matrix which may be computationally expensive if the state-space is large [1]. Fortunately we can do some algebra to come up with equivalent equations that do not involve an inverse.

We begin by stating a block matrix identity. Given matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ the following is true (for non-singular $\mathbf{A}$):

$$(\mathbf{A}+\mathbf{B}\mathbf{C}\mathbf{B}^T)^{-1}=\mathbf{A}^{-1}-\mathbf{A}^{-1}\mathbf{B}(\mathbf{C}^{-1}+\mathbf{B}^T\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{A}^{-1} \tag{3.17}$$

[1] in some navigation applications the dimension of $\mathbf{x}$ can approach the high hundreds

We can immediately apply this to 3.14 to get:

$$\begin{aligned}\mathbf{P}_\oplus&=\mathbf{P}_\ominus-\mathbf{P}_\ominus\mathbf{H}^T(\mathbf{R}+\mathbf{H}\mathbf{P}_\ominus\mathbf{H}^T)^{-1}\mathbf{H}\mathbf{P}_\ominus && (3.18)\\ &=\mathbf{P}_\ominus-\mathbf{W}\mathbf{S}\mathbf{W}^T && (3.19)\\ & && (3.20)\end{aligned}$$

or

$$=(\mathbf{I}-\mathbf{W}\mathbf{H})\mathbf{P}_\ominus \tag{3.21}$$

where

$$\begin{aligned}\mathbf{S}&=\mathbf{H}\mathbf{P}_\ominus\mathbf{H}^T+\mathbf{R} && (3.22)\\ \mathbf{W}&=\mathbf{P}_\ominus\mathbf{H}^T\mathbf{S}^{-1} && (3.23)\end{aligned}$$

Now look at the form of the update equation 3.16 it is a linear combination of $\mathbf{x}$ and $\mathbf{z}$. Combining 3.20 with 3.16 we have:

$$\begin{aligned}\mathbf{x}_\oplus&=(\mathbf{P}_\ominus^{-1}+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}(\mathbf{P}_\ominus^{-1}\mathbf{x}_\ominus+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z}) && (3.24)\\ &=(\mathbf{P}_\ominus^{-1}+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z})+(\mathbf{I}-\mathbf{W}\mathbf{H})\mathbf{P}_\ominus(\mathbf{P}_\ominus^{-1}\mathbf{x}_\ominus) && (3.25)\\ &=(\mathbf{P}_\ominus^{-1}+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z})+\mathbf{x}_\ominus+\mathbf{W}(-\mathbf{H}\mathbf{x}_\ominus) && (3.26)\\ &=\mathbf{C}\mathbf{z}+\mathbf{x}_\ominus+\mathbf{W}(-\mathbf{H}\mathbf{x}_\ominus) && (3.27)\end{aligned}$$

where

$$\mathbf{C}=(\mathbf{P}_\ominus^{-1}+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{R}^{-1} \tag{3.28}$$

Taking a step aside we note that both

$$\mathbf{H}^T\mathbf{R}^{-1}(\mathbf{H}\mathbf{P}_\ominus\mathbf{H}^T+\mathbf{R})=\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P}_\ominus\mathbf{H}^T+\mathbf{H}^T \tag{3.29}$$

and also

$$(\mathbf{P}_\ominus^{-1}+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})\mathbf{P}_\ominus\mathbf{H}^T=\mathbf{H}^T+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P}_\ominus\mathbf{H}^T \tag{3.30}$$

so

$$(\mathbf{P}_\ominus^{-1}+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{R}^{-1}=\mathbf{P}_\ominus\mathbf{H}^T(\mathbf{H}\mathbf{P}_\ominus\mathbf{H}^T+\mathbf{R})^{-1} \tag{3.31}$$

therefore

$$\begin{aligned}\mathbf{C}&=\mathbf{P}_\ominus\mathbf{H}^T\mathbf{S}^{-1} && (3.32)\\ &=\mathbf{W}\text{from 3.23} && (3.33)\end{aligned}$$

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 $\mathbf{P}_\oplus$:

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

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

– where ν is the innovation given by: $\nu = \mathbf{z} - \mathbf{H}\mathbf{x}_\ominus$,
– where S is the innovation covariance given by: $\mathbf{S} = \mathbf{H}\mathbf{P}_\ominus\mathbf{H}^\top + \mathbf{R}$,[15]
– 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

[15] 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[16] (*mapped by* $\mathbf{B}$ *onto the states*),
– where $\mathbf{F}$ is the state transition matrix[17].

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

LKF - Temporal-Conditional Notation

The temporal-conditional[18] 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

[18] 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

- **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. [19]
- **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 $\mathbf{H}\hat{\mathbf{x}}_{(k|k-1)}$.

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

LKF - Discussion

- **Asynchrosity**: The update step only proceeds when the measurements come, not necessarily at every iteration. [20]
- **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 $\mathbf{WSW}^{\top}$[21] $\rightarrow$ the $\mathbf{P}_{k|k}$ update covariance decreases / not increases.

[20] 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.

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

LKF - Discussion

- **Observability**: the measurements $\mathbf{z}$ need not to fully determine the state vector $\mathbf{x}$, the LKF can perform[22] updates using only partial measurements thanks to:

  – prior info about unobserved states and

  – correlations.[23]

- **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.

---

[22] Note: In contrary to LSQ that needs enough measurements to solve for the state values.

[23] Note: Over the time for unobservable states the covariance will grow without bound.

LKF - Linear Navigation Problem

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:

$$\mathbf{x}_{(k)} = \underbrace{\begin{bmatrix} 1 & \delta T \\ 0 & 1 \end{bmatrix}}_{\mathbf{F}} \mathbf{x}_{(k-1)} + \underbrace{\begin{bmatrix} \delta 0.5 T^2 \\ \delta T \end{bmatrix}}_{\mathbf{G}} \mathbf{v}_{(k)}$$

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

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 $\mathbf{Q}$[24], the measurement noise $\mathbf{w}$ is given by the covariance matrix $\mathbf{R}$.[25]

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

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

LKF - Linear Navigation Problem

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.[26]

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

[26] 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

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.

Example - MATLAB

```
%process plant model (constant velcoity with noise in acceleration)
Params.F = [1 Params.dT;
            0 1];

%process noise model (maps acceleration noise to other states)
Params.G = [Params.dT^2/2 ;Params.dT];

%actual process noise truely occuring - atmospher entry is a bumpy business
%note this noise strength - not the deacceleration of the vehicle....
Params.SigmaQ = 0.2; %ms^{-2}

%process noise strength how much acceleration (varinace) in one tick
% we expect (used to 'explain' inaccuracies in our model)
%the 3 is scale factor (set it to 1 and real and modelled noises will
%be equal
Params.Q = (1.1*Params.SigmaQ)^2; %(ms^2 std)

%observation model (explains observations in terms of state to be estimated)
Params.H = [2/Params.c_light 0];

%observation noise strength (RTrue) is how noisey the sensor really is
Params.SigmaR = 1.3e-7; %(seconds) 3.0e-7 corresponds to around 50m error....

%observation expected noise strength (we never know this parameter exactly)
%set the scale factor to 1 to make model and reallity match
Params.R = (1.1*Params.SigmaR)^2;
```

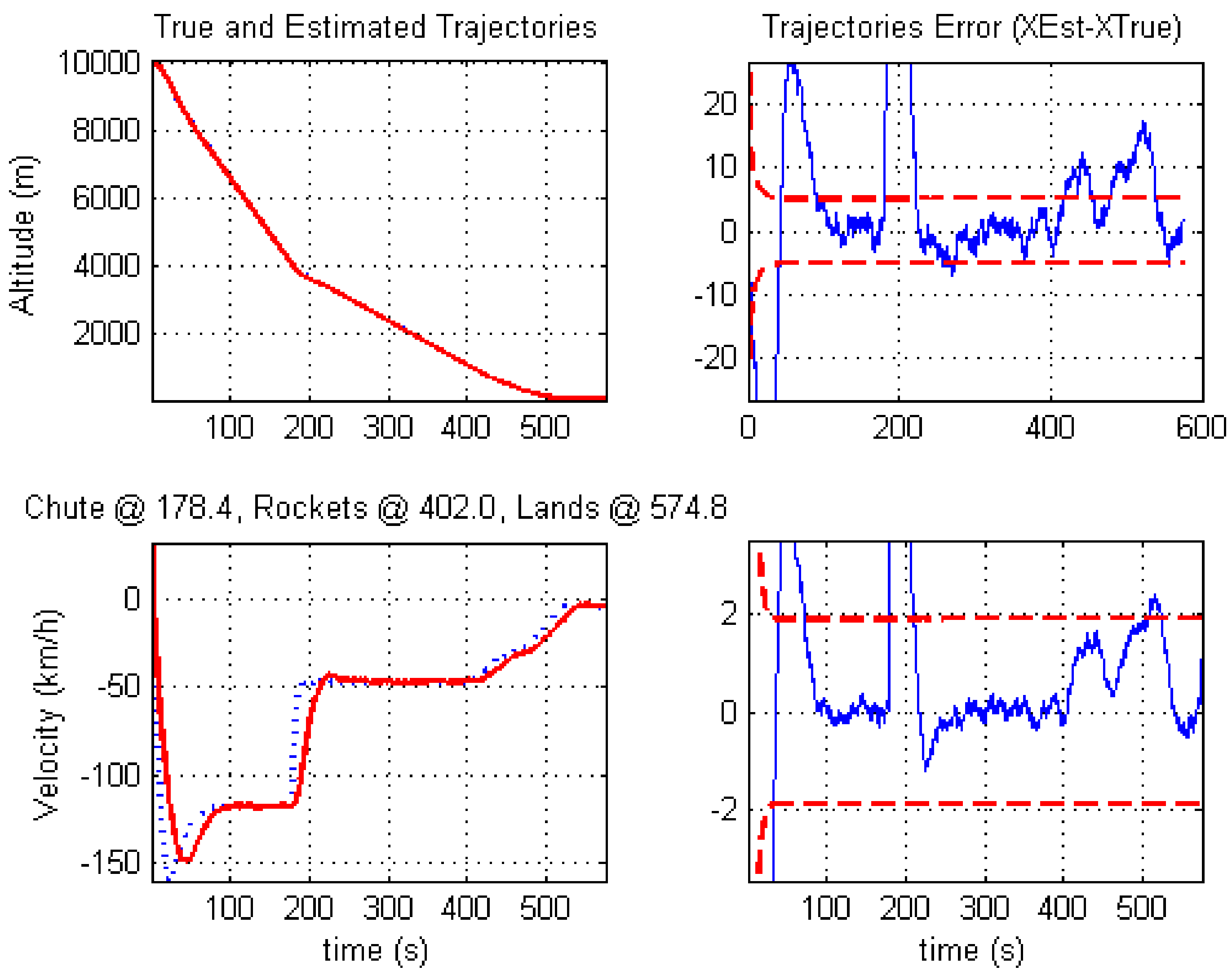
Example - MATLAB

```
%--------------- ESTIMATION KALMAN FILTER ---------------%
function [XEst,PEst,S,Innovation] = DoEstimation(XEst,PEst,z)
global Params;
F = Params.F;G = Params.G;Q = Params.Q;R = Params.R;H = Params.H;

%prediction...
XPred = F*XEst;
PPred = F*PEst*F'+G*Q*G';

% prepare for update...
Innovation = z-H*XPred;
S = H*PPred*H'+R;
W = PPred*H'*inv(S);

% do update....
XEst = XPred+W*Innovation;
PEst = PPred-W*S*W';
return;
```

LKF - Linear Navigation Problem

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

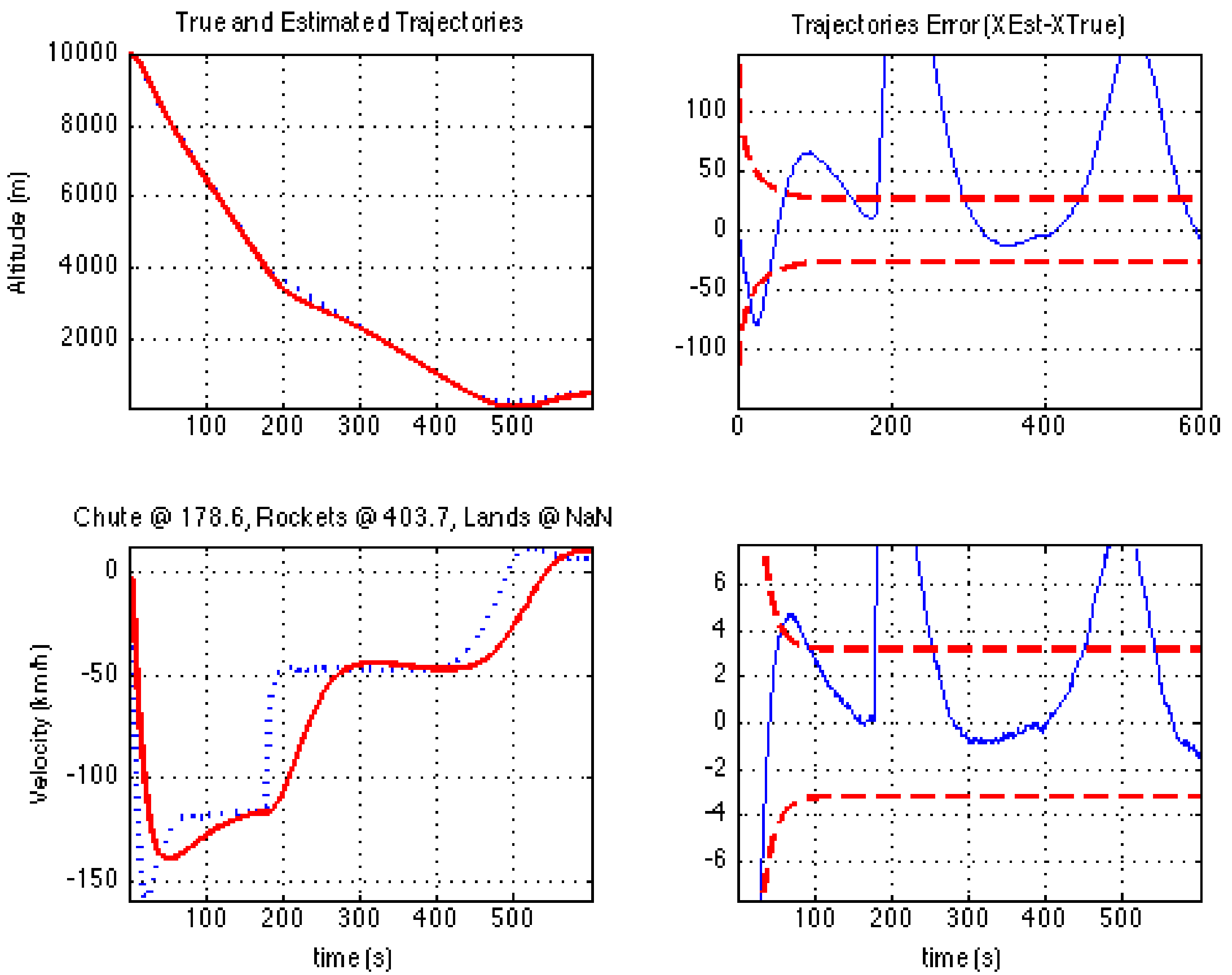
We did good modeling, errors are due to the non-linear world!



LKF - Linear Navigation Problem

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

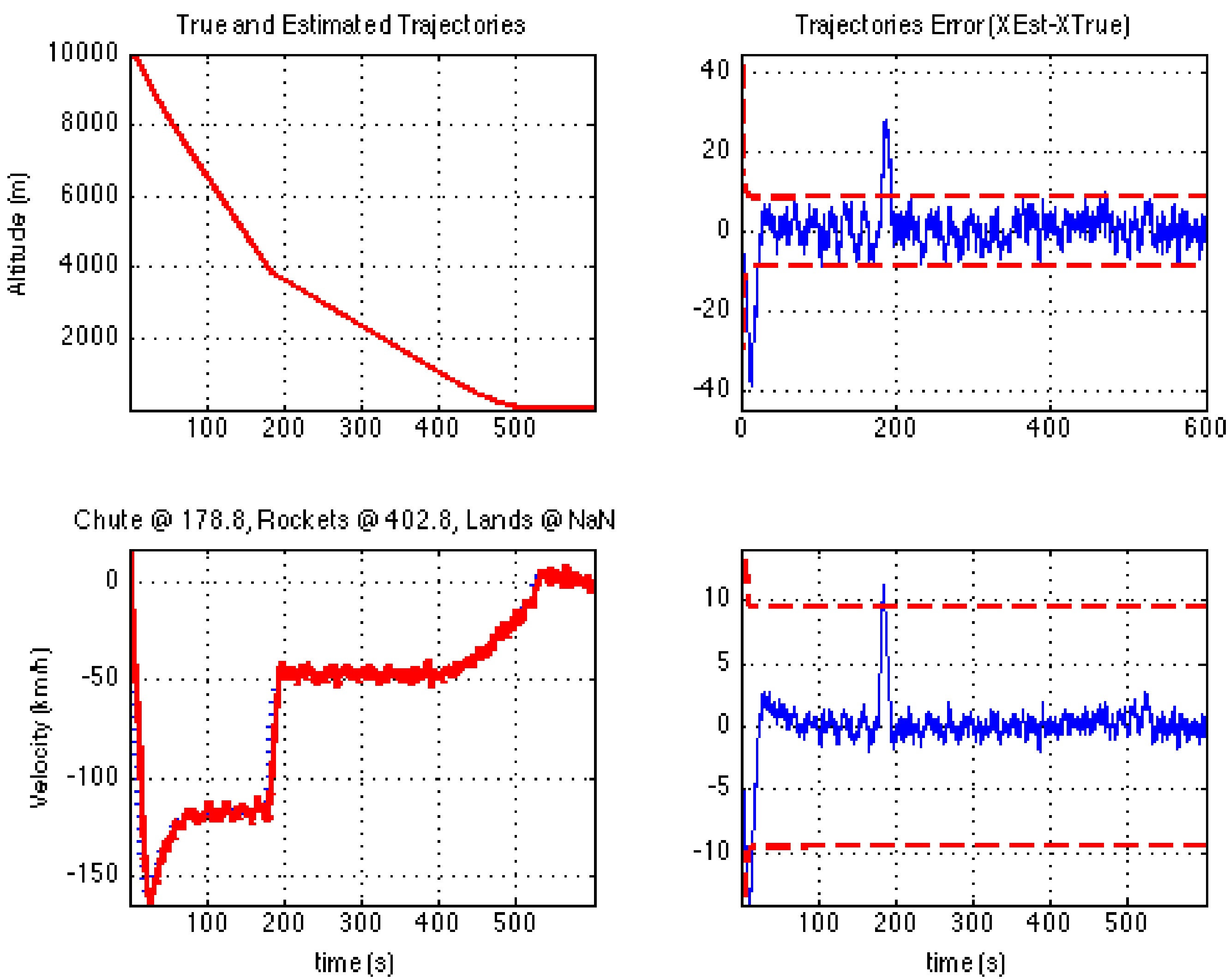
We do not trust the measurements, the good linear model alone is not enough!



LKF - Linear Navigation Problem

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

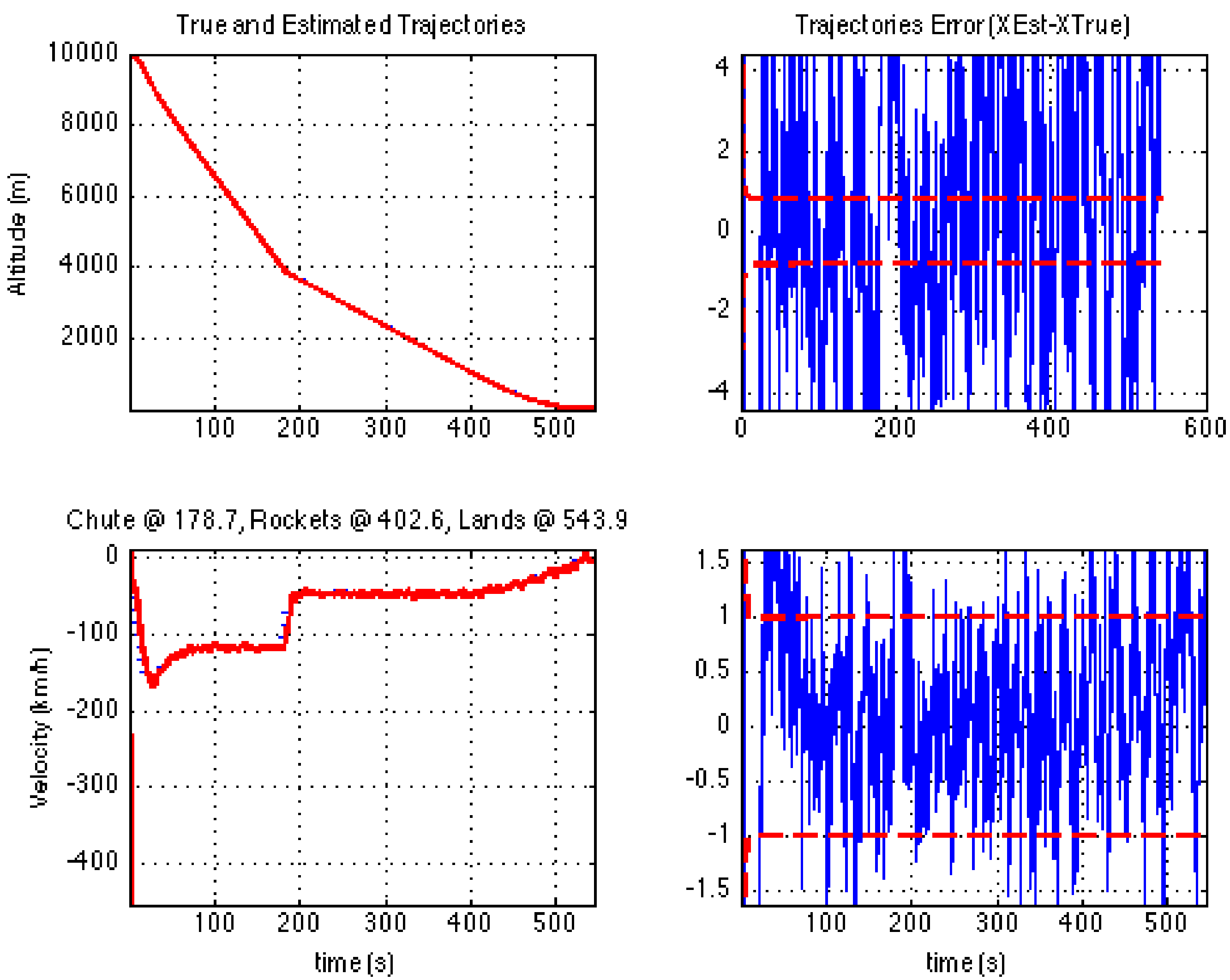
We do not trust our model, the estimates have good mean but are too noisy!



LKF - Linear Navigation Problem

Example - Results for: $\sigma_{\mathrm{R}}^{\mathrm{model}} = 0.1\sigma_{\mathrm{R}}^{\mathrm{true}}$, $\sigma_{\mathrm{Q}}^{\mathrm{model}} = 1.1\sigma_{\mathrm{Q}}^{\mathrm{true}}$

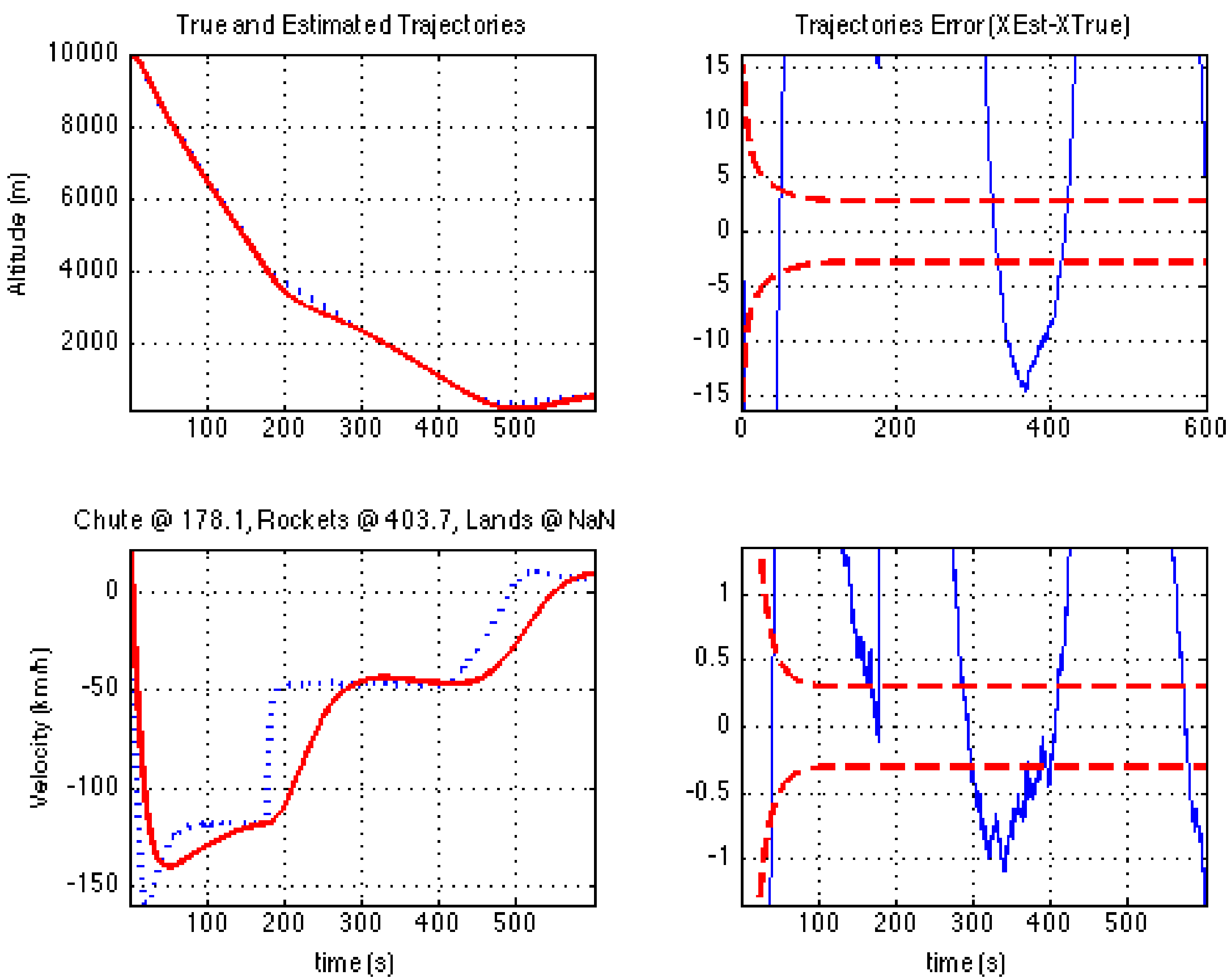
We are overconfident measurements—fortunately, the sensor is not more noisy!



LKF - Linear Navigation Problem

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

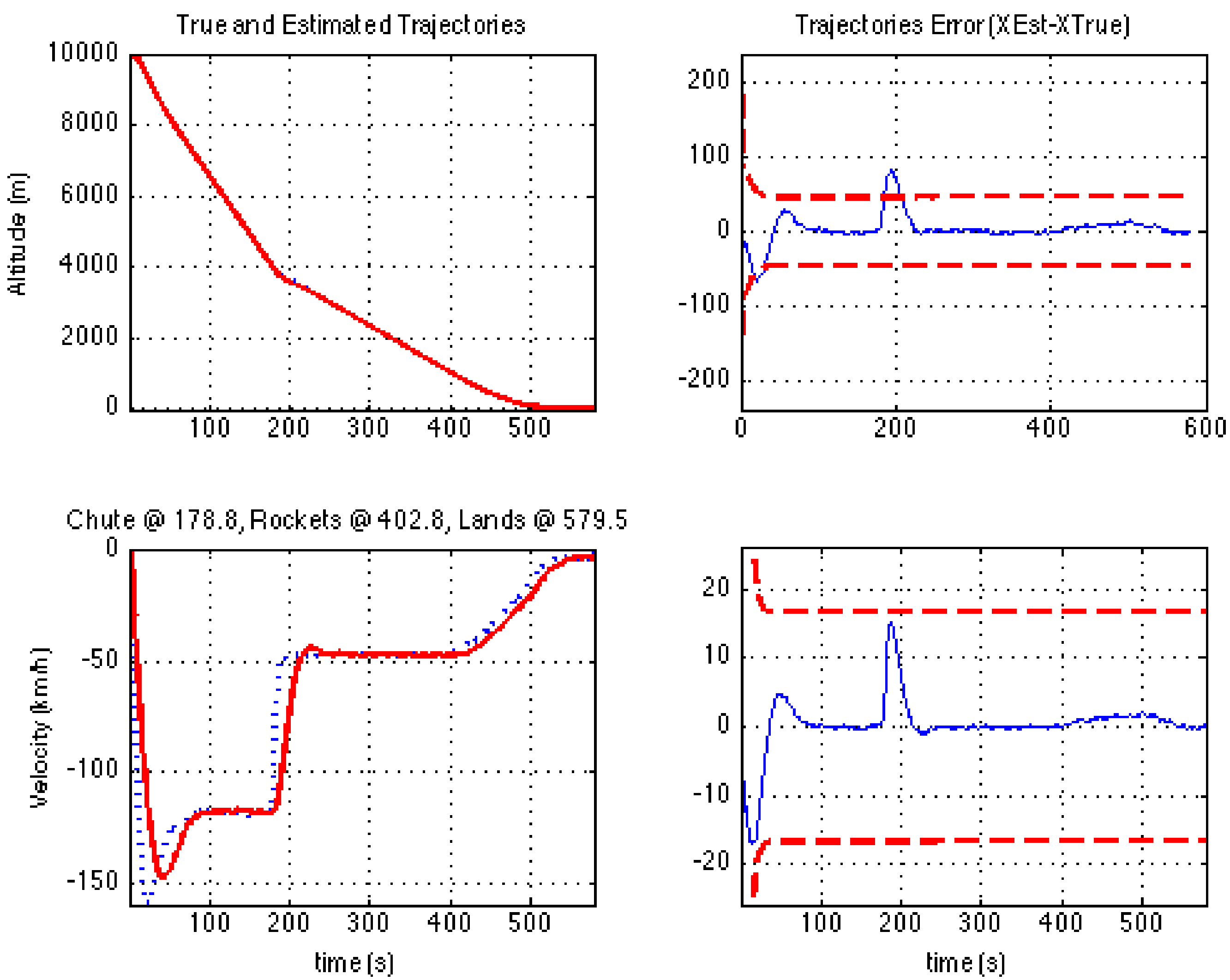
We are overconfident in our model, but the world is really not linear ...



LKF - Linear Navigation Problem

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

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[27]: The main idea behind EKF is to linearize the non-linear model around the „best" current estimate[28].

This is realized using a Taylor series expansion[29].

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 $\hat{\mathbf{x}}_{(k-1|k-1)}$:

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

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

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

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

EKF - Non-linear Observation

Without proof[30]: 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[31].

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 $\hat{\mathbf{x}}_{(k|k-1)}$:

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

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

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

EKF - Algorithm (1)

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

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

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

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

EKF - Algorithm (2)

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

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

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

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

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

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

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[32] 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 $p(\mathbf{x}_v|\mathbf{M}, \mathbf{Z^k})$.

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

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

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

[32] 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 $\mathbf{x}_v$, [33] and a sequence of vehicle-relative observations $\mathbf{Z^k}$ described by likelihood $p(\mathbf{Z^k}|\mathbf{M}, \mathbf{x}_v)$.

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

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

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

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

[33] 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.).