# Lecture Note

CS572 Intelligent Robotics

Joonyoung Yi[1,3], Kwonsoo Chae[1,3],
Minh Son Cao[1,3], Donghoon Baek[2,3]

[1]KAIST School of Computing , [2]KAIST Robotics Program,
[3]KAIST 2018 Fall CS572 Team 9

**November 5th 2018**

## 1. Continuous Kalman Filter

The Model: Continuous-time linear system, with white noises state and measurement noises (not necessarily Gaussian).

Goal: Develop the continuous-time Kalman filter as the optimal linear estimator (L-MMSE) for this system.

The state-space model:

$$\frac{d}{dt}x_t = F_t x_t + w_t\,, \quad t \geq 0$$

$$z_t = H_t x_t + v_t$$

where:

- $\{w_t\}$ and $\{v_t\}$ are zero-mean white-noise processes, namely

$$E(w_t w_s^T) = Q_t \delta(t-s), \quad E(v_t v_s^T) = R_t \delta(t-s)$$

$$E(w_t v_s^T) = 0$$

- $\{w_t\}$, $\{v_t\}$ and $x_0$ are uncorrelated.
- $E(x_0) = \overline{x}_0$ and $\text{cov}(x_0) = P_0$ are given.
- We shall assume that $R_t$ is non-singular.

Let $Z_t = \{z_s,\ s < t\}$. We need to calculate $\hat{x}_t = E^L(x_t|Z_t)$.

Define the innovations process:

$$\tilde{z}_t := z_t - E^L(z_t|Z_t)$$

Observe that

$$\tilde{z}_t = z_t - H_t E^L(x_t|Z_t) = H_t \tilde{x}_t + v_t$$

where $\tilde{x}_t = x_t - \hat{x}_t$.

Properties of $\tilde{z}_t$:

$\tilde{z}_t$ is a zero-mean *white* noise process (exercise). Its covariance equals:

$$E(\tilde{z}_t \tilde{z}_s^T) = R_t \delta(t-s)\,.$$

Note: same covariance as $z_t$ !

It can also be shown that $\tilde{Z}_t$ and $Z_t$ are linearly equivalent, so that

$$E^L(\cdot|Z_t) = E^L(\cdot|\tilde{Z}_t)\,.$$

It follows that $\hat{x}_t$ can be expressed as a linear function of $\tilde{Z}_t$:

$$\hat{x}_t = \int_0^t g(t,s)\tilde{z}_s ds$$

The kernel $g(t,s)$ is easily computable via the orthogonality principle. Since $\tilde{x}_t :=$ $(x_t - \hat{x}_t) \perp \tilde{z}_s$ for $s < t$,

$$\begin{aligned} E(x_t \tilde{z}_s^T) &= \int_0^t g(t,r)E(\tilde{z}_r \tilde{z}_s^T)dr \\ &= \int_0^t g(t,r)\delta(s-r)R_s dr = g(t,s)R_s, \quad s < t\,. \end{aligned}$$

## 2. Matrix Fraction Decomposition

Let $A$ be an $n \times d$ matrix with singular vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$ and corresponding singular values $\sigma_1, \sigma_2, \ldots, \sigma_r$. Then $\mathbf{u_i} = \frac{1}{\sigma_i} A\mathbf{v_i}$, for $i = 1, 2, \ldots, r$, are the left singular vectors and by Theorem 1.5, $A$ can be decomposed into a sum of rank one matrices as

$$A = \sum_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T.$$

We first prove a simple lemma stating that two matrices $A$ and $B$ are identical if $A\mathbf{v} = B\mathbf{v}$ for all $\mathbf{v}$. The lemma states that in the abstract, a matrix $A$ can be viewed as a transformation that maps vector $\mathbf{v}$ onto $A\mathbf{v}$.

**Theorem 1.5** *Let $A$ be an $n \times d$ matrix with right singular vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$, left singular vectors $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_r}$, and corresponding singular values $\sigma_1, \sigma_2, \ldots, \sigma_r$. Then*

$$A = \sum_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T.$$

**Proof:** For each singular vector $\mathbf{v_j}$, $A\mathbf{v_j} = \sum\limits_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T \mathbf{v_j}$. Since any vector $\mathbf{v}$ can be expressed as a linear combination of the singular vectors plus a vector perpendicular to the $\mathbf{v_i}$, $A\mathbf{v} = \sum\limits_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T \mathbf{v}$ and by Lemma 1.4, $A = \sum\limits_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T$.

The decomposition is called the *singular value decomposition*, *SVD*, of $A$. In matrix notation $A = UDV^T$ where the columns of $U$ and $V$ consist of the left and right singular vectors, respectively, and $D$ is a diagonal matrix whose diagonal entries are the singular values of $A$.

For any matrix $A$, the sequence of singular values is unique and if the singular values are all distinct, then the sequence of singular vectors is unique also. However, when some set of singular values are equal, the corresponding singular vectors span some subspace. Any set of orthonormal vectors spanning this subspace can be used as the singular vectors.

## 3. Gradient Descent Method

**Direction and step size.** In general, one can consider a search for a stationary point as having two components: the direction and the step size. The direction decides which direction we search next, and the step size determines how far we go in that particular direction. Such methods can be generally described as starting at some arbitrary point $\mathbf{x}^{(0)}$ and then at every step $k \geq 0$ iteratively moving at direction $\Delta\mathbf{x}^{(k)}$ by step size $t_k$ to the next point $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + t_k \cdot \Delta\mathbf{x}^{(k)}$. In gradient descent, the direction we search is the negative gradient at the point, i.e. $\Delta\mathbf{x} = -\nabla f(\mathbf{x})$. Thus, the iterative search of gradient descent can be described through the following recursive rule:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - t_k \nabla f(\mathbf{x}^{(k)})$$

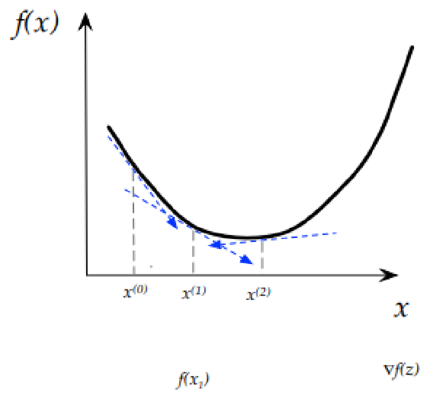
**Choosing a step size.** Given that the search for a stationary point is currently at a certain point $\mathbf{x}^{(k)}$, how should we choose our step size $t_k$? Since our objective is to minimize the function, one reasonable approach is to choose the step size in manner that will minimize the value of the new point, i.e. find the step size that minimizes $f(\mathbf{x}^{(k+1)})$. Since $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - t\nabla f(\mathbf{x}^{(k)})$ the step size $t_k^\star$ of this approach is:

$$t_k^\star = \text{argmin}_{t \geq 0}\, f(\mathbf{x}^{(k)} - t\nabla f(\mathbf{x}^{(k)}))$$

**The algorithm.** Formally, given a desired precision $\epsilon > 0$, we define the gradient descent as described below.

**Algorithm 1** Gradient Descent

```
1: Guess $\mathbf{x}^{(0)}$, set $k \leftarrow 0$
2: while $||\nabla f(\mathbf{x}^{(k)})|| \geq \epsilon$ do
3:     $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - t_k \nabla f(\mathbf{x}^{(k)})$
4:     $k \leftarrow k + 1$
5: end while
6: return $\mathbf{x}^{(k)}$
```



## 4. Perceptron Convergence Theorem

The convergence theorem is as follows:

**Theorem 1** *Assume that there exists some parameter vector $\underline{\theta}^*$ such that $||\underline{\theta}^*|| = 1$, and some $\gamma > 0$ such that for all $t = 1 \ldots n$,*

$$y_t(\underline{x}_t \cdot \underline{\theta}^*) \geq \gamma$$

*Assume in addition that for all $t = 1 \ldots n$, $||\underline{x}_t|| \leq R$.*

*Then the perceptron algorithm makes at most*

$$\frac{R^2}{\gamma^2}$$

*errors. (The definition of an error is as follows: an error occurs whenever we have $y' \neq y_t$ for some $(j, t)$ pair in the algorithm.)*

Note that for any vector $\underline{x}$, we use $||\underline{x}||$ to refer to the Euclidean norm of $\underline{x}$, i.e., $||\underline{x}|| = \sqrt{\sum_i x_i^2}$.

*Proof:* First, define $\underline{\theta}^k$ to be the parameter vector when the algorithm makes its $k$'th error. Note that we have

$$\underline{\theta}^1 = \underline{0}$$

Next, assuming the $k$'th error is made on example $t$, we have

$$\begin{aligned} \underline{\theta}^{k+1} \cdot \underline{\theta}^* &= (\underline{\theta}^k + y_t \underline{x}_t) \cdot \underline{\theta}^* \\ &= \underline{\theta}^k \cdot \underline{\theta}^* + y_t \underline{x}_t \cdot \underline{\theta}^* \\ &\geq \underline{\theta}^k \cdot \underline{\theta}^* + \gamma \end{aligned}$$