非线性系统滤波算法

蔡远利　教授
西安交通大学自动化学院

0. Outline

1. 非线性贝叶斯滤波理论

1.1 问题描述

考虑加性噪声非线性系统:

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k, k) + \boldsymbol{w}_k \tag{1}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{h}(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}, k+1) + \boldsymbol{v}_{k+1} \tag{2}$$

假设

1. $\{\boldsymbol{w}_k\}$、$\{\boldsymbol{v}_k\}$ 和随机初始状态 $\boldsymbol{x}_0$ 三者相互独立，$\{\boldsymbol{w}_k\}$ 和 $\{\boldsymbol{v}_k\}$ 都是白噪声序列；
2. $\boldsymbol{w}_k$ 的概率分布密度函数为 $f_w(\boldsymbol{w}_k)$；
3. $\boldsymbol{v}_k$ 的概率分布密度函数为 $f_v(\boldsymbol{v}_k)$；
4. $\boldsymbol{x}_0$ 的概率分布密度函数为 $f_x(\boldsymbol{x}_0)$；

贝叶斯滤波就是基于量测 $\mathbf{Y}_1^k = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_k\}$，根据先验概率分布密度函数 $f_x(\boldsymbol{x}_0)$ 求 $\boldsymbol{x}_k$ 的后验概率分布密度函数 $f_x(\boldsymbol{x}_k|\mathbf{Y}_1^k)$。

1.2 查普曼-柯尔莫哥洛夫方程

设 $\{\boldsymbol{x}(t)\}$ 是一随机过程，如果随机向量 $[\boldsymbol{x}(t_1), \boldsymbol{x}(t_2), \cdots, \boldsymbol{x}(t_m)]^{\mathrm{T}} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m]^{\mathrm{T}}$ 的条件概率密度满足

$$f(\boldsymbol{x}_m|\boldsymbol{x}_{m-1}, \boldsymbol{x}_{m-2}, \cdots, \boldsymbol{x}_1) = f(\boldsymbol{x}_m|\boldsymbol{x}_{m-1}) \tag{3}$$

那么称 $\{\boldsymbol{x}(t)\}$ 为**一阶马尔可夫过程**。

其中 $t_1 \sim t_m$ 是任意选取的一组时间点，$f(\boldsymbol{x}_m|\boldsymbol{x}_{m-1})$ 称为一步转移概率密度函数，$f(\boldsymbol{x}_m|\boldsymbol{x}_{m-l})$ 称为 l 步转移概率密度函数。

对于马尔可夫过程，根据贝叶斯法则可知

$$\begin{aligned} & f(\boldsymbol{x}_m,\boldsymbol{x}_{m-1},\boldsymbol{x}_{m-2},\cdots,\boldsymbol{x}_1) \\ &\quad = f(\boldsymbol{x}_m|\boldsymbol{x}_{m-1},\boldsymbol{x}_{m-2},\cdots,\boldsymbol{x}_1)f(\boldsymbol{x}_{m-1},\boldsymbol{x}_{m-2},\cdots,\boldsymbol{x}_1) \\ &\quad = f(\boldsymbol{x}_m|\boldsymbol{x}_{m-1})f(\boldsymbol{x}_{m-1}|\boldsymbol{x}_{m-2})\cdots f(\boldsymbol{x}_2|\boldsymbol{x}_1)f(\boldsymbol{x}_1) \\ &\quad = f(\boldsymbol{x}_1)\prod_{k=1}^{m-1} f(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k) \end{aligned} \tag{4}$$

另外，由边缘密度函数计算可知

$$\int_{-\infty}^{+\infty} f(\boldsymbol{x}_m,\boldsymbol{x}_{m-1},\boldsymbol{x}_{m-2})\mathrm{d}\boldsymbol{x}_{m-1} = f(\boldsymbol{x}_m,\boldsymbol{x}_{m-2}) \tag{5}$$

即

$$\int_{-\infty}^{+\infty} f(\boldsymbol{x}_m|\boldsymbol{x}_{m-1})f(\boldsymbol{x}_{m-1}|\boldsymbol{x}_{m-2})f(\boldsymbol{x}_{m-2})\mathrm{d}\boldsymbol{x}_{m-1} = f(\boldsymbol{x}_m|\boldsymbol{x}_{m-2})f(\boldsymbol{x}_{m-2}) \tag{6}$$

所以

$$f(\boldsymbol{x}_m|\boldsymbol{x}_{m-2}) = \int_{-\infty}^{+\infty} f(\boldsymbol{x}_m|\boldsymbol{x}_{m-1})f(\boldsymbol{x}_{m-1}|\boldsymbol{x}_{m-2})\mathrm{d}\boldsymbol{x}_{m-1} \tag{7}$$

这就是著名的查普曼-柯尔莫哥洛夫方程。更一般地，可以表示为

$$f(\boldsymbol{x}_m|\boldsymbol{x}_{m-l}) = \int_{-\infty}^{+\infty} f(\boldsymbol{x}_m|\boldsymbol{x}_{m-1})f(\boldsymbol{x}_{m-1}|\boldsymbol{x}_{m-l})\mathrm{d}\boldsymbol{x}_{m-1} \tag{8}$$

1.3 贝叶斯滤波公式

根据贝叶斯法则，我们有

$$\begin{aligned} f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^{k+1}) &= f_x(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k+1},\mathbf{Y}_1^k) \\ &= \frac{f_{xy}(\boldsymbol{x}_{k+1},\boldsymbol{y}_{k+1},\mathbf{Y}_1^k)}{f_y(\boldsymbol{y}_{k+1},\mathbf{Y}_1^k)} \\ &= \frac{f_{xy}(\boldsymbol{x}_{k+1},\boldsymbol{y}_{k+1}|\mathbf{Y}_1^k)}{f_y(\boldsymbol{y}_{k+1}|\mathbf{Y}_1^k)} \end{aligned} \tag{9}$$

注意到（因为 $\boldsymbol{v}_k$ 的统计性质）

$$\begin{aligned} f_{xy}(\boldsymbol{x}_{k+1},\boldsymbol{y}_{k+1}|\mathbf{Y}_1^k) &= f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k) f_y(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1},\mathbf{Y}_1^k) \\ &= f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k) f_y(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1}) \end{aligned}$$

$$\Longrightarrow \qquad f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^{k+1}) = \frac{f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k) f_y(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1})}{f_y(\boldsymbol{y}_{k+1}|\mathbf{Y}_1^k)} \tag{10}$$

根据查普曼-柯尔莫哥洛夫方程，可知

$$\begin{aligned} f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k) &= \int_x f_x(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k, \mathbf{Y}_1^k) f_x(\boldsymbol{x}_k|\mathbf{Y}_1^k) \mathrm{d}\boldsymbol{x}_k \\ &= \int_x f_x(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k) f_x(\boldsymbol{x}_k|\mathbf{Y}_1^k) \mathrm{d}\boldsymbol{x}_k \end{aligned} \tag{11}$$

$$\begin{aligned} f_y(\boldsymbol{y}_{k+1}|\mathbf{Y}_1^k) &= \int_x f_y(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1}, \mathbf{Y}_1^k) f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k) \mathrm{d}\boldsymbol{x}_{k+1} \\ &= \int_x f_y(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1}) f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k) \mathrm{d}\boldsymbol{x}_{k+1} \end{aligned} \tag{12}$$

而由系统方程 (1)、(2)，可知

$$f_x(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k) = f_w(\boldsymbol{x}_{k+1} - \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k, k)) \tag{13}$$

$$f_y(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1}) = f_v(\boldsymbol{y}_{k+1} - \boldsymbol{h}(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}, k+1)) \tag{14}$$

将以上二式代入 (11)、(12)，得

$$f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k) = \int_x f_w(\boldsymbol{x}_{k+1} - \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k, k)) f_x(\boldsymbol{x}_k|\mathbf{Y}_1^k) \mathrm{d}\boldsymbol{x}_k \tag{15}$$

$$f_y(\boldsymbol{y}_{k+1}|\mathbf{Y}_1^k) = \int_x f_v(\boldsymbol{y}_{k+1} - \boldsymbol{h}(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}, k+1)) f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k) \mathrm{d}\boldsymbol{x}_{k+1} \tag{16}$$

在起始时刻

$$f_x(\boldsymbol{x}_0|\mathbf{Y}_1^0) = f_x(\boldsymbol{x}_0) \tag{17}$$

Table 1: 贝叶斯滤波算法

状态方程	$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k, k) + \boldsymbol{w}_k$
量测方程	$\boldsymbol{y}_{k+1} = \boldsymbol{h}(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}, k+1) + \boldsymbol{v}_{k+1}$
状态初始概率密度	$f_x(\boldsymbol{x}_0 \vert \mathbf{Y}_1^0) = f_x(\boldsymbol{x}_0)$
状态预测概率密度	$f_x(\boldsymbol{x}_{k+1} \vert \mathbf{Y}_1^k) = \int_x f_w(\boldsymbol{x}_{k+1} - \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k, k)) f_x(\boldsymbol{x}_k \vert \mathbf{Y}_1^k) \mathrm{d}\boldsymbol{x}_k$
量测预测概率密度	$f_y(\boldsymbol{y}_{k+1} \vert \mathbf{Y}_1^k) = \int_x f_v(\boldsymbol{y}_{k+1} - \boldsymbol{h}(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}, k+1)) f_x(\boldsymbol{x}_{k+1} \vert \mathbf{Y}_1^k) \mathrm{d}\boldsymbol{x}_{k+1}$
状态后验概率密度	$f_x(\boldsymbol{x}_{k+1} \vert \mathbf{Y}_1^{k+1}) = \frac{f_x(\boldsymbol{x}_{k+1} \vert \mathbf{Y}_1^k) f_y(\boldsymbol{y}_{k+1} \vert \boldsymbol{x}_{k+1})}{f_y(\boldsymbol{y}_{k+1} \vert \mathbf{Y}_1^k)}$

- 式 (9)、(15) ∼ (17) 便构成了计算的递推算式，即贝叶斯滤波算法，见表1。

- 有了状态的后验概率密度，可以根据需要，进一步求出状态的滤波值及误差协方差等低阶统计信息。例如

$$\hat{\boldsymbol{x}}_{k+1|k+1} = E(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^{k+1}) = \int_x \boldsymbol{x}_{k+1} f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^{k+1})\mathrm{d}\boldsymbol{x}_{k+1} \tag{18}$$

$$P_{k+1|k+1} = E(\tilde{\boldsymbol{x}}_{k+1}\tilde{\boldsymbol{x}}_{k+1}^{\mathrm{T}}) = \int_x \tilde{\boldsymbol{x}}_{k+1}\tilde{\boldsymbol{x}}_{k+1}^{\mathrm{T}} f_x(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^{k+1})\mathrm{d}\boldsymbol{x}_{k+1} \tag{19}$$

- 对于线性系统，贝叶斯滤波算法公式可以解析求解，即可导出卡尔曼滤波算法。一般情况下，需要求高维的非线性积分，通常非常困难，由此发展出了许多近似处理方法。

2. 基于标称状态的线性化滤波方法

如果系统的真实状态 $\boldsymbol{x}$ 总是围绕标称状态 $\boldsymbol{x}^*$ 附近变化，当标称状态 $\boldsymbol{x}^*$ 已知时，我们可以取 $\boldsymbol{x} = \boldsymbol{x}^* + \Delta\boldsymbol{x}$，从而建立一种可行的滤波算法。

考虑如下非线性离散时间随机系统:

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k, k) + \boldsymbol{w}_k \tag{20}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{h}(\boldsymbol{x}_{k+1}, k+1) + \boldsymbol{v}_{k+1} \tag{21}$$

其中，过程噪声 $\boldsymbol{w}_k \sim (0, Q_k)$ 与量测噪声 $\boldsymbol{v}_{k+1} \sim (0, R_{k+1})$ 相互独立，设 $Q_k \geq 0, R_{k+1} > 0$。

标称的状态及量测方程定义为

$$\boldsymbol{x}_{k+1}^{*} = \boldsymbol{g}(\boldsymbol{x}_{k}^{*}, k) \tag{22}$$

$$\boldsymbol{y}_{k+1}^{*} = \boldsymbol{h}(\boldsymbol{x}_{k+1}^{*}, k+1) \tag{23}$$

设

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_{k+1}^{*} + \Delta\boldsymbol{x}_{k+1}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_{k+1}^{*} + \Delta\boldsymbol{y}_{k+1}$$

一阶近似地

$$\Delta\boldsymbol{x}_{k+1} = F_k\Delta\boldsymbol{x}_k + \boldsymbol{w}_k \tag{24}$$

$$\Delta\boldsymbol{y}_{k+1} = H_{k+1}\Delta\boldsymbol{x}_{k+1} + \boldsymbol{v}_{k+1} \tag{25}$$

其中，$F_k = \left.\frac{\partial \boldsymbol{g}(\boldsymbol{x}_k,k)}{\partial \boldsymbol{x}_k^{\mathrm{T}}}\right|_{\boldsymbol{x}_k=\boldsymbol{x}_k^*}$，$H_k = \left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}_k,k)}{\partial \boldsymbol{x}_k^{\mathrm{T}}}\right|_{\boldsymbol{x}_k=\boldsymbol{x}_k^*}$，称为系统线性化过程中生成的雅可比矩阵。

如果 $\boldsymbol{x}_0 \sim (\boldsymbol{x}_0^*, P_0)$，即 $\Delta\boldsymbol{x}_0 \sim (0, P_0)$，而且与过程噪声 $\boldsymbol{w}_k \sim (0, Q_k)$、量测噪声 $\boldsymbol{v}_k \sim (0, R_k)$ 相互独立。那么，我们可以应用卡尔曼滤波算法于(24)、(25)，从而得到原系统的一种滤波算法，如表2所示。

Remark 2.1 在建立上述算法中，关于一步预测 $\hat{\boldsymbol{x}}_{k+1|k} = E[\boldsymbol{x}_{k+1}|\boldsymbol{Y}_1^k]$ 有

$$\begin{aligned}\hat{\boldsymbol{x}}_{k+1|k} &= E[\boldsymbol{g}(\boldsymbol{x}_k,k) + \boldsymbol{w}_k|\boldsymbol{Y}_1^k] = E[\boldsymbol{g}(\boldsymbol{x}_k,k)|\boldsymbol{Y}_1^k] \\ &\approx E[\boldsymbol{g}(\boldsymbol{x}_k^*,k) + F_k(\boldsymbol{x}_k^* - \boldsymbol{x}_k)|\boldsymbol{Y}_1^k] \\ &= \underbrace{\boldsymbol{g}(\boldsymbol{x}_k^*,k) + F_k(\boldsymbol{x}_k^* - \hat{\boldsymbol{x}}_{k|k})}_{\text{用于计算}P_{k+1|k}} \approx \boldsymbol{g}(\hat{\boldsymbol{x}}_{k|k},k)\end{aligned}$$

Table 2: 基于标称状态线性化的卡尔曼滤波算法

状态方程	$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k, k) + \boldsymbol{w}_k$
量测方程	$\boldsymbol{y}_{k+1} = \boldsymbol{h}(\boldsymbol{x}_{k+1}, k+1) + \boldsymbol{v}_{k+1}$
滤波初值	$\hat{\boldsymbol{x}}_{0\|0} = E\boldsymbol{x}_0 = \boldsymbol{x}_0^*, \quad P_{0\|0} = \mathrm{var}(\boldsymbol{x}_0) = P_0$
一步预测	$\hat{\boldsymbol{x}}_{k+1\|k} = \boldsymbol{g}(\hat{\boldsymbol{x}}_{k\|k}, k)$ $P_{k+1\|k} = F_k P_{k\|k} F_k^{\mathrm{T}} + Q_k$
滤波增益	$K_{k+1} = P_{k+1\|k} H_{k+1}^{\mathrm{T}} (H_{k+1} P_{k+1\|k} H_{k+1}^{\mathrm{T}} + R_{k+1})^{-1}$
量测修正	$\hat{\boldsymbol{x}}_{k+1\|k+1} = \hat{\boldsymbol{x}}_{k+1\|k} + K_{k+1}[\boldsymbol{y}_{k+1} - \boldsymbol{h}(\hat{\boldsymbol{x}}_{k+1\|k}, k+1)]$ $P_{k+1\|k+1} = (I - K_{k+1} H_{k+1}) P_{k+1\|k}$
雅可比阵	$F_k = \left.\frac{\partial \boldsymbol{g}(\boldsymbol{x}_k, k)}{\partial \boldsymbol{x}_k^{\mathrm{T}}}\right\|_{\boldsymbol{x}_k = \boldsymbol{x}_k^*}, \quad H_k = \left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}_k, k)}{\partial \boldsymbol{x}_k^{\mathrm{T}}}\right\|_{\boldsymbol{x}_k = \boldsymbol{x}_k^*}$

Remark 2.2 关于量测预报 $\hat{\boldsymbol{y}}_{k+1|k}=E[\boldsymbol{y}_{k+1}|\boldsymbol{Y}_1^k]$ 有

$$\begin{aligned}\hat{\boldsymbol{y}}_{k+1|k}&=E[\boldsymbol{h}(\boldsymbol{x}_{k+1},k+1)+\boldsymbol{v}_{k+1}|\boldsymbol{Y}_1^k]=E[\boldsymbol{h}(\boldsymbol{x}_{k+1},k+1)|\boldsymbol{Y}_1^k]\\&\approx E[\boldsymbol{h}(\boldsymbol{x}_{k+1}^*,k+1)+H_{k+1}(\boldsymbol{x}_{k+1}^*-\boldsymbol{x}_{k+1})|\boldsymbol{Y}_1^k]\\&=\underbrace{\boldsymbol{h}(\boldsymbol{x}_{k+1}^*,k+1)+H_{k+1}(\boldsymbol{x}_{k+1}^*-\hat{\boldsymbol{x}}_{k+1|k})}_{\text{用于计算}P_{k+1|k+1}}\\&\approx\boldsymbol{h}(\hat{\boldsymbol{x}}_{k+1|k},k+1)\end{aligned}$$

3. 扩展卡尔曼滤波

- ♠ 如果滤波器已经正常工作，那么滤波器当前的输出值将非常接近实际的工作状态。
- ♠ 将滤波器每个时刻的输出作为线性化的参考，由此可以构建一种线性化滤波方法，这就是扩展卡尔曼滤波（EKF）的基本思想。

3.1 扩展卡尔曼滤波基本公式

定义 (雅可比矩阵)

$$F_k = \left.\frac{\partial \boldsymbol{g}(\boldsymbol{x}_k, k)}{\partial \boldsymbol{x}_k{}^{\mathrm{T}}}\right|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_{k|k}} \tag{26}$$

$$H_{k+1} = \left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}_{k+1}, k+1)}{\partial \boldsymbol{x}_{k+1}^{\mathrm{T}}}\right|_{\boldsymbol{x}_{k+1}=\hat{\boldsymbol{x}}_{k+1|k}} \tag{27}$$

那么

$$\boldsymbol{x}_{k+1} \approx F_k \boldsymbol{x}_k + [\boldsymbol{g}(\hat{\boldsymbol{x}}_{k|k}, k) - F_k \hat{\boldsymbol{x}}_{k|k}] + \boldsymbol{w}_k \tag{28}$$

$$\boldsymbol{y}_{k+1} \approx H_{k+1} \boldsymbol{x}_{k+1} + [\boldsymbol{h}(\hat{\boldsymbol{x}}_{k+1|k}, k+1) - H_{k+1} \hat{\boldsymbol{x}}_{k+1|k}] + \boldsymbol{v}_{k+1} \tag{29}$$

因为在求 $k+1$ 时刻状态的滤波时，$\hat{\boldsymbol{x}}_{k|k}$、$\hat{\boldsymbol{x}}_{k+1|k}$ 已经知道，可以视为确定性信号。另外，考虑到

$$\begin{aligned}\hat{\boldsymbol{x}}_{k+1|k} &= E[\boldsymbol{x}_{k+1}|\boldsymbol{Y}_1^k] \\ &= E[\boldsymbol{g}(\boldsymbol{x}_k,k)+\boldsymbol{w}_k|\boldsymbol{Y}_1^k] = E[\boldsymbol{g}(\boldsymbol{x}_k,k)|\boldsymbol{Y}_1^k] \\ &\approx E[\boldsymbol{g}(\hat{\boldsymbol{x}}_{k|k},k)+F_k(\hat{\boldsymbol{x}}_{k|k}-\boldsymbol{x}_k)|\boldsymbol{Y}_1^k] \approx \boldsymbol{g}(\hat{\boldsymbol{x}}_{k|k},k)\end{aligned}$$

$$\begin{aligned}\hat{\boldsymbol{y}}_{k+1|k} &= E[\boldsymbol{y}_{k+1}|\boldsymbol{Y}_1^k] \\ &= E[\boldsymbol{h}(\boldsymbol{x}_{k+1},k+1)+\boldsymbol{v}_{k+1}|\boldsymbol{Y}_1^k] = E[\boldsymbol{h}(\boldsymbol{x}_{k+1},k+1)|\boldsymbol{Y}_1^k] \\ &\approx E[\boldsymbol{h}(\hat{\boldsymbol{x}}_{k+1|k},k+1)+H_{k+1}(\hat{\boldsymbol{x}}_{k+1|k}-\boldsymbol{x}_{k+1})|\boldsymbol{Y}_1^k] \approx \boldsymbol{h}(\hat{\boldsymbol{x}}_{k+1|k},k+1)\end{aligned}$$

这样，我们便可以借助卡尔曼滤波基本方程建立一套有效的非线性滤波算法，称为扩展卡尔曼滤波器 (Extended Kalman Filter, EKF)，如表3所示。

Table 3: 扩展卡尔曼滤波算法

状态方程	$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k, k) + \boldsymbol{w}_k$
量测方程	$\boldsymbol{y}_{k+1} = \boldsymbol{h}(\boldsymbol{x}_{k+1}, k+1) + \boldsymbol{v}_{k+1}$
滤波初值	$\hat{\boldsymbol{x}}_{0\|0} = E\boldsymbol{x}_0 = \bar{\boldsymbol{x}}_0, \quad P_{0\|0} = \mathrm{var}(\boldsymbol{x}_0) = P_0$
一步预测	$\hat{\boldsymbol{x}}_{k+1\|k} = \boldsymbol{g}(\hat{\boldsymbol{x}}_{k\|k}, k)$ $P_{k+1\|k} = F_k P_{k\|k} F_k^{\mathrm{T}} + Q_k$
滤波增益	$K_{k+1} = P_{k+1\|k} H_{k+1}^{\mathrm{T}} (H_{k+1} P_{k+1\|k} H_{k+1}^{\mathrm{T}} + R_{k+1})^{-1}$
量测修正	$\hat{\boldsymbol{x}}_{k+1\|k+1} = \hat{\boldsymbol{x}}_{k+1\|k} + K_{k+1}[\boldsymbol{y}_{k+1} - \boldsymbol{h}(\hat{\boldsymbol{x}}_{k+1\|k}, k+1)]$ $P_{k+1\|k+1} = (I - K_{k+1} H_{k+1}) P_{k+1\|k}$
雅可比阵	$F_k = \left.\frac{\partial \boldsymbol{g}(\boldsymbol{x}_k, k)}{\partial \boldsymbol{x}_k^{\mathrm{T}}}\right\|_{\boldsymbol{x}_k = \hat{\boldsymbol{x}}_{k\|k}}, \; H_{k+1} = \left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}_{k+1}, k+1)}{\partial \boldsymbol{x}_{k+1}^{\mathrm{T}}}\right\|_{\boldsymbol{x}_{k+1} = \hat{\boldsymbol{x}}_{k+1\|k}}$

3.2 迭代扩展卡尔曼滤波

设按扩展卡尔曼滤波器已经获得了 $\hat{\boldsymbol{x}}_{k|k}$、$\hat{\boldsymbol{x}}_{k|k-1}$，记 $\hat{\boldsymbol{x}}_{k|k}^{(0)}=\hat{\boldsymbol{x}}_{k|k-1}$，可以以此进一步改善 $\hat{\boldsymbol{y}}_{k|k-1}$ 的计算。

定义 $H_k(\hat{\boldsymbol{x}}_{k|k}^{(i)})=\left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}_k,k)}{\partial \boldsymbol{x}_k^{\mathrm{T}}}\right|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_{k|k}^{(i)}}$，$i=0,1,2,\cdots$，于是

$$\boldsymbol{y}_k\approx H_k(\hat{\boldsymbol{x}}_k^{(i)})\boldsymbol{x}_k+\boldsymbol{h}(\hat{\boldsymbol{x}}_{k|k}^{(i)},k)-H_k(\hat{\boldsymbol{x}}_k^{(i)})\hat{\boldsymbol{x}}_{k|k}^{(i)}+\boldsymbol{v}_k$$

因此

$$\hat{\boldsymbol{y}}_{k|k-1}^{(i)}\approx\boldsymbol{h}(\hat{\boldsymbol{x}}_{k|k}^{(i)},k)+H_k(\hat{\boldsymbol{x}}_k^{(i)})(\hat{\boldsymbol{x}}_{k|k-1}-\hat{\boldsymbol{x}}_{k|k}^{(i)})\approx\boldsymbol{h}(\hat{\boldsymbol{x}}_{k|k}^{(i)},k)$$

从而可得比 $\hat{\boldsymbol{x}}_{k|k}$ 更好的估计 $\hat{\boldsymbol{x}}_{k|k}^{(i+1)}$，即

$$\hat{\boldsymbol{x}}_{k|k}^{(i+1)} = \hat{\boldsymbol{x}}_{k|k-1} + K_k^{(i)}[\boldsymbol{y}_k - \hat{\boldsymbol{y}}_{k|k-1}^{(i)}] \tag{30}$$

$$K_k^{(i)} = P_{k|k-1}H_k^{\mathrm{T}}(\hat{\boldsymbol{x}}_{k|k}^{(i)})[H_k(\hat{\boldsymbol{x}}_{k|k}^{(i)})P_{k|k-1}H_k^{\mathrm{T}}(\hat{\boldsymbol{x}}_{k|k}^{(i)}) + R_{k+1}]^{-1} \tag{31}$$

$$P_{k|k}^{(i+1)} = [I - K_k^{(i)}H_k(\hat{\boldsymbol{x}}_{k|k}^{(i)})]P_{k+1|k} \tag{32}$$

按式30)～ 式 (32) 反复迭代 $(i = 1, 2, 3, \cdots)$，直到不能改善为止 ($\|\hat{\boldsymbol{x}}_{k|k}^{(i+1)} - \hat{\boldsymbol{x}}_{k|k}^{(i)}\|$ 小于给定的精度)。这便是迭代扩展卡尔曼滤波器(Iterative Extended Kalman Filter, IEKF)，其精度将高于普通扩展卡尔曼滤波，但计算时间会增加。

Remark 3.1 工程应用中，为了保证滤波算法的实时性，*IEKF* 的迭代次数一般取 $3 \sim 5$。

Remark 3.2 *IEKF* 通过在扩展卡尔曼滤波中量测修正环节引入迭代计算，不断改进线性化参考点，能够更好地解决量测非线性的影响。

4. 无迹卡尔曼滤波

4.1 无迹变换 (Unscented Transformation, UT)

考虑非线性映

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}) \tag{33}$$

其中，$\boldsymbol{x} \in \mathbb{R}^n \sim \mathcal{N}(\bar{\boldsymbol{x}}, P_x)$，$\boldsymbol{y} \in \mathbb{R}^m$。由于 $\boldsymbol{x}$ 是随机变量，因此 $\boldsymbol{y}$ 也是随机变量。由于非线性的原因，尽管 $\boldsymbol{x}$ 是高斯分布的，但通常 $\boldsymbol{y}$ 不一定还是高斯分布的。

构造加权 sigma 点集

$$\boldsymbol{\chi}_0 = \bar{\boldsymbol{x}}, \qquad w_0 = \frac{\kappa}{n+\kappa}, i = 0 \tag{34}$$

$$\boldsymbol{\chi}_i = \bar{\boldsymbol{x}} + (\sqrt{(n+\kappa)P_x})_i, \qquad w_i = \frac{1}{2(n+\kappa)}, i = 1, \cdots, n \tag{35}$$

$$\boldsymbol{\chi}_i = \bar{\boldsymbol{x}} - (\sqrt{(n+\kappa)P_x})_{i-n}, \qquad w_i = \frac{1}{2(n+\kappa)}, i = n+1, \cdots, 2n \tag{36}$$

其中，κ 是可调参数。当 $\boldsymbol{x}$ 为正态分布时，$\kappa = 3 - n$。

那么，精确到 2 阶以上（泰勒级数）有

$$\bar{\boldsymbol{y}} = \sum_{i=0}^{2n} w_i \boldsymbol{y}_i, \quad \boldsymbol{y}_i = \boldsymbol{h}(\boldsymbol{\chi}_i) \tag{37}$$

$$P_y = \sum_{i=0}^{2n} w_i (\boldsymbol{y}_i - \bar{\boldsymbol{y}})(\boldsymbol{y}_i - \bar{\boldsymbol{y}})^{\mathrm{T}} \tag{38}$$

上述结论可以通过泰勒级数展开分析获得。与传统的线性化方法相比，UT 提供了计算非线性变换后 $\bar{\boldsymbol{y}}$ 和 P_y 更有效的手段，二者比较图见图1。

在 UT 的基础上，人们又进一步提出了缩放无迹变换 (Scaled Unscented Transformation, SUT)。通过缩放因子，可以提高数值稳定性。此外，在建模噪声可能不均匀分布时，特别有用。

在缩放 UT 中，sigma 点及对应的权重为

$$\boldsymbol{\chi}_0 = \bar{\boldsymbol{x}}, \quad i = 0 \tag{39}$$

$$\boldsymbol{\chi}_i = \bar{\boldsymbol{x}} + (\sqrt{(n+\lambda)P_x})_i, \quad i = 1, \cdots, n \tag{40}$$

$$\boldsymbol{\chi}_i = \bar{\boldsymbol{x}} - (\sqrt{(n+\lambda)P_x})_{i-n}, \quad i = n+1, \cdots, 2n \tag{41}$$

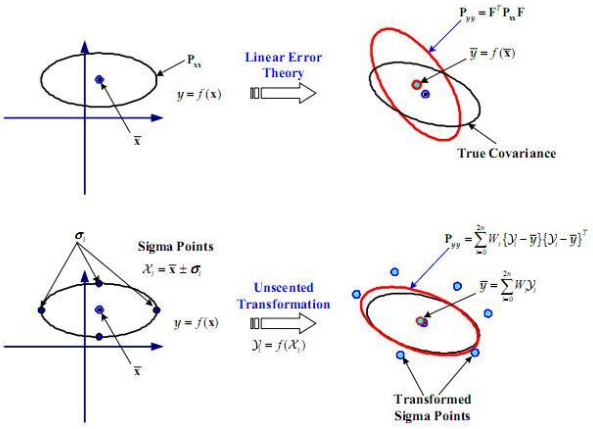

Figure 1: UT 示意图

$$w_i^{(m)} = \begin{cases} \lambda/(n+\lambda), & i=0 \\ 1/2(n+\lambda), & i=1,2,\cdots,2n \end{cases} \tag{42}$$

$$w_i^{(c)} = \begin{cases} \lambda/(n+\lambda)+(1-\alpha^2+\beta), & i=0 \\ 1/2(n+\lambda), & i=1,2,\cdots,2n \end{cases} \tag{43}$$

其中，可调参数 $0 \le \alpha \le 1$(一般可选 $\alpha = 10^{-3}$)，β 根据 $\boldsymbol{x}$ 先验知识选取(正态分布时，$\beta = 2$ 为最优)，κ 通常取为 0，$\lambda = (n+\kappa)\alpha^2 - n$.

对于 SUT，随机变量非线性变换后的均值及协方差矩阵计算公式为

$$\bar{\boldsymbol{y}} = \sum_{i=0}^{2n} w_i^{(m)} \boldsymbol{y}_i, \quad \boldsymbol{y}_i = \boldsymbol{h}(\boldsymbol{\chi}_i) \tag{44}$$

$$P_y = \sum_{i=0}^{2n} w_i^{(c)} (\boldsymbol{y}_i - \bar{\boldsymbol{y}})(\boldsymbol{y}_i - \bar{\boldsymbol{y}})^{\mathsf{T}} \tag{45}$$

4.2 EKF 另一种形式

考虑更一般的非线性系统

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{w}_k, k) \tag{46}$$

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k, k) + \boldsymbol{v}_k \tag{47}$$

简记

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_{k|k}, \quad P_k = P_{k|k}$$

$$\hat{\boldsymbol{x}}_k^- = \hat{\boldsymbol{x}}_{k|k-1}, \quad P_k^- = P_{k|k-1}$$

那么，关于非线性系统 (46)、(47) 的 EKF 可重新描述如下:

(1) 初始化: $\hat{\boldsymbol{x}}_0 = E\boldsymbol{x}_0, \quad P_0 = \mathrm{var}(\boldsymbol{x}_0)$

(2) 时间修正（time update）：

$$\hat{\boldsymbol{x}}_{k+1}^{-} = \boldsymbol{f}(\hat{\boldsymbol{x}}_k, 0, k) \tag{48}$$

$$P_{k+1}^{-} = F_k P_k F_k^{\mathrm{T}} + G_k Q_k G_k^{\mathrm{T}} \tag{49}$$

(3) 量测修正（measurement update）：

$$\hat{\boldsymbol{y}}_{k+1}^{-} = \boldsymbol{h}(\hat{\boldsymbol{x}}_{k+1}^{-}, k+1) \tag{50}$$

$$P_{k+1}^{yy} = H_{k+1} P_{k+1}^{-} H_{k+1}^{\mathrm{T}} + R_{k+1} \tag{51}$$

$$P_{k+1}^{xy} = P_{k+1}^{-} H_{k+1}^{\mathrm{T}} \tag{52}$$

$$K_{k+1} = P_{k+1}^{xy} (P_{k+1}^{yy})^{-1} \tag{53}$$

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1}^{-} + K_{k+1}(\boldsymbol{y}_{k+1} - \hat{\boldsymbol{y}}_{k+1}^{-}) \tag{54}$$

$$P_{k+1} = P_{k+1}^{-} - K_{k+1} P_{k+1}^{yy} K_{k+1}^{\mathrm{T}} \tag{55}$$

其中

$$F_k = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{w}_k, k)}{\partial \boldsymbol{x}_k^{\mathrm{T}}}\right|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_k, \boldsymbol{w}_k=0}$$

$$G_k = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{w}_k, k)}{\partial \boldsymbol{w}_k^{\mathrm{T}}}\right|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_k, \boldsymbol{w}_k=0}$$

$$H_{k+1} = \left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}_{k+1}, k+1)}{\partial \boldsymbol{x}_{k+1}^{\mathrm{T}}}\right|_{\boldsymbol{x}_{k+1}=\hat{\boldsymbol{x}}_{k+1}^{-}}$$

按简记符号，$\hat{\boldsymbol{x}}_{k+1}^{-}$、$\hat{\boldsymbol{y}}_{k+1}^{-}$ 分别表示系统状态及量测的（一步）预测估计。

4.3 无迹卡尔曼滤波算法

基于上小节讨论的 EKF 形式，可以发现系统状态及量测的一步预测及相关协方差计算都可以采用前面介绍的 SUT 技术，由此即可建立所谓的 UKF。

下面考虑式 (46) 和 (47) 描述的非线性系统，UKF 基本步骤如下：

(1) 扩展状态

$$\boldsymbol{x}_k^a = \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{w}_k \end{bmatrix}, \quad P_k^a = \begin{bmatrix} P_k & P_k^{xw} \\ P_k^{wx} & \boldsymbol{Q}_k \end{bmatrix} \tag{56}$$

(2) 选择 sigma 点

依据 $(\hat{\boldsymbol{x}}_k^a, P_k^a)$ 生成 $L(=2(n+q))+1$ 个 sigma 点 $(\boldsymbol{\chi}_{i,k}^a)$, $i = 0, 1, \cdots, L$。

(3) **时间修正**

$$\boldsymbol{\chi}_{i,k+1} = \boldsymbol{f}(\chi^a_{i,k}, k) \tag{57}$$

$$\hat{\boldsymbol{x}}^-_{k+1} = \sum_{i=0}^{L} w_i^{(m)} \boldsymbol{\chi}_{i,k+1} \tag{58}$$

$$P^-_{k+1} = \sum_{i=0}^{L} w_i^{(c)} (\boldsymbol{\chi}_{i,k+1} - \hat{\boldsymbol{x}}^-_{k+1})(\boldsymbol{\chi}_{i,k+1} - \hat{\boldsymbol{x}}^-_{k+1})^{\mathrm{T}} \tag{59}$$

(4) 量测修正

$$\boldsymbol{y}_{i,k+1} = \boldsymbol{h}(\boldsymbol{\chi}_{i,k+1}, k+1) \tag{60}$$

$$\hat{\boldsymbol{y}}_{k+1}^{-} = \sum_{i=0}^{L} w_i^{(m)} \boldsymbol{y}_{i,k+1} \tag{61}$$

$$P_{k+1}^{yy} = \sum_{i=0}^{L} w_i^{(c)} (\boldsymbol{y}_{i,k+1} - \hat{\boldsymbol{y}}_{k+1}^{-})(\boldsymbol{y}_{i,k+1} - \hat{\boldsymbol{y}}_{k+1}^{-})^{\mathrm{T}} + R_{k+1} \tag{62}$$

$$P_{k+1}^{xy} = \sum_{i=0}^{L} w_i^{(c)} (\boldsymbol{\chi}_{i,k+1} - \hat{\boldsymbol{x}}_{k+1}^{-})(\boldsymbol{y}_{i,k+1} - \hat{\boldsymbol{y}}_{k+1}^{-})^{\mathrm{T}} \tag{63}$$

$$\begin{cases} K_{k+1} = P_{k+1}^{xy} (P_{k+1}^{yy})^{-1} \\ \hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1}^{-} + K_{k+1}(\boldsymbol{y}_{k+1} - \hat{\boldsymbol{y}}_{k+1}^{-}) \\ P_{k+1} = P_{k+1}^{-} - K_{k+1} P_{k+1}^{yy} K_{k+1}^{\mathrm{T}} \end{cases} \tag{64}$$

如果系统的过程噪声和量测噪声都是加性的，则不需要上述 UKF 算法中的扩展状态环节。

例如对于式 (1)、(2) 描述的加性噪声系统，对应的 UKF 算法主要步骤简化如下:

(1) 选择 sigma 点

依据 $(\hat{\boldsymbol{x}}_k, P_k)$ 生成 $L(=2n+1)$ 个 sigma 点 $\boldsymbol{\chi}_{i,k}$，$i=0,1,\cdots,L$。

(2) **时间修正**

$$\boldsymbol{\chi}_{i,k+1} = \boldsymbol{f}(\boldsymbol{\chi}_{i,k}, k) \tag{65}$$

$$\hat{\boldsymbol{x}}_{k+1}^{-} = \sum_{i=0}^{L} \boldsymbol{w}_i^{(m)} \boldsymbol{\chi}_{i,k+1} \tag{66}$$

$$P_{k+1}^{-} = \sum_{i=0}^{L} \boldsymbol{w}_i^{(c)} (\boldsymbol{\chi}_{i,k+1} - \hat{\boldsymbol{x}}_{k+1}^{-})(\boldsymbol{\chi}_{i,k+1} - \hat{\boldsymbol{x}}_{k+1}^{-})^{\mathrm{T}} + Q_k \tag{67}$$

(3) 量测修正

$$\boldsymbol{y}_{i,k+1} = \boldsymbol{h}(\boldsymbol{\chi}_{i,k+1}, k+1) \tag{68}$$

$$\hat{\boldsymbol{y}}_{k+1}^{-} = \sum_{i=0}^{L} \boldsymbol{w}_i^{(m)} \boldsymbol{y}_{i,k+1} \tag{69}$$

$$P_{k+1}^{yy} = \sum_{i=0}^{L} \boldsymbol{w}_i^{(c)} (\boldsymbol{y}_{i,k+1} - \hat{\boldsymbol{y}}_{k+1}^{-})(\boldsymbol{y}_{i,k+1} - \hat{\boldsymbol{y}}_{k+1}^{-})^{\mathrm{T}} + R_{k+1} \tag{70}$$

$$P_{k+1}^{xy} = \sum_{i=0}^{L} \boldsymbol{w}_i^{(c)} (\boldsymbol{\chi}_{i,k+1} - \hat{\boldsymbol{x}}_{k+1}^{-})(\boldsymbol{y}_{i,k+1} - \hat{\boldsymbol{y}}_{k+1}^{-})^{\mathrm{T}} \tag{71}$$

$$\begin{cases} K_{k+1} = P_{k+1}^{xy} (P_{k+1}^{yy})^{-1} \\ \hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1}^{-} + K_{k+1}(\boldsymbol{y}_{k+1} - \hat{\boldsymbol{y}}_{k+1}^{-}) \\ P_{k+1} = P_{k+1}^{-} - K_{k+1} P_{k+1}^{yy} K_{k+1}^{\mathrm{T}} \end{cases} \tag{72}$$

- ♢ UKF 中核心是构造 sigma 点，需要计算协方差矩阵的平方根，可以采用 Cholesky 分解算法。
- ♢ 大量研究表明，UKF 与 EKF 计算量相当或更小，但不需要计算雅可比矩阵，滤波精度及计算稳定性均更好。
- ♢ 关于 UKF 有许多改进，例如平方根滤波算法、自适应算法等。
- ♢ UKF 除应用于非线性状态估计外，也可以应用于参数估计、机器学习、信号处理、时间序列预测、图像处理等领域。
- ♢ 关于 UT、UKF 的详细推导、分析及应用，可参见 Julier、Farina、Jargani 等学者的论文。

5. 容积卡尔曼滤波

为了便于描述，考虑时不变非线性系统，即

$$\boldsymbol{x}_k = \boldsymbol{g}(\boldsymbol{x}_{k-1}) + \boldsymbol{w}_{k-1} \tag{73}$$

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k \tag{74}$$

在时刻 $k-1$，假设状态 $\boldsymbol{x}_{k-1}$ 近似服从高斯分布，已经获得了 $\boldsymbol{x}_{k-1} \sim f(\boldsymbol{x}_{k-1}|\mathbf{Y}_1^{k-1}) = \mathcal{N}(\hat{\boldsymbol{x}}_{k-1}, P_{k-1})$。同前，$\mathbf{Y}_1^k$ 表示 $\boldsymbol{y}_1$ 到 $\boldsymbol{y}_k$ 的量测集合，即 $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_k\}$。那么

$$\begin{aligned}\hat{\boldsymbol{x}}_k^- &= E[\boldsymbol{x}_k|\mathbf{Y}_1^{k-1}] = E[\boldsymbol{g}(\boldsymbol{x}_{k-1}) + \boldsymbol{w}_{k-1}|\mathbf{Y}_1^{k-1}] = E[\boldsymbol{g}(\boldsymbol{x}_{k-1})|\mathbf{Y}_1^{k-1}] \\ &= \int_{\mathbb{R}^{n_x}} \boldsymbol{g}(\boldsymbol{x}_{k-1}) f(\boldsymbol{x}_{k-1}|\mathbf{Y}_1^{k-1}) \mathrm{d}\boldsymbol{x}_{k-1} = \int_{\mathbb{R}^{n_x}} \boldsymbol{g}(\boldsymbol{x}_{k-1}) \mathcal{N}(\hat{\boldsymbol{x}}_{k-1}, P_{k-1}) \mathrm{d}\boldsymbol{x}_{k-1}\end{aligned} \tag{75}$$

$$\begin{aligned}P_k^- &= E[(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-)(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-)^{\mathrm{T}}|\mathbf{Y}_1^{k-1}] \\ &= \int_{\mathbb{R}^{n_x}} \boldsymbol{g}(\boldsymbol{x}_{k-1}) \boldsymbol{g}^{\mathrm{T}}(\boldsymbol{x}_{k-1}) \mathcal{N}(\hat{\boldsymbol{x}}_{k-1}, P_{k-1}) \mathrm{d}\boldsymbol{x}_{k-1} - \hat{\boldsymbol{x}}_k^- (\hat{\boldsymbol{x}}_k^-)^{\mathrm{T}} + Q_{k-1}\end{aligned} \tag{76}$$

同时，近似地认为 $\boldsymbol{x}_k \sim \mathcal{N}(\hat{\boldsymbol{x}}_k^-, P_k^-)$，有

$$\hat{\boldsymbol{y}}_k^- = E[\boldsymbol{y}_k|\mathbf{Y}_1^{k-1}] = \int_{\mathbb{R}^{n-x}} \boldsymbol{h}(\boldsymbol{x}_{k-1})\mathcal{N}(\hat{\boldsymbol{x}}_k^-, P_k^-)\mathrm{d}\boldsymbol{x}_k \tag{77}$$

$$P_{yy,k} = \int_{\mathbb{R}^{n_x}} \boldsymbol{h}(\boldsymbol{x}_{k-1})\boldsymbol{h}^{\mathrm{T}}(\boldsymbol{x}_{k-1})\mathcal{N}(\hat{\boldsymbol{x}}_k^-, P_k^-)\mathrm{d}\boldsymbol{x}_k - \hat{\boldsymbol{y}}_k^-(\hat{\boldsymbol{y}}_k^-)^{\mathrm{T}} + R_k \tag{78}$$

因此，可以认为似然密度函数服从高斯分布 $f(\boldsymbol{y}_k|\mathbf{Y}_1^k) = \mathcal{N}(\hat{\boldsymbol{y}}_k^-, P_{yy,k})$。

$$P_{xy,k} = \int_{\mathbb{R}^{n_x}} \boldsymbol{x}_k\boldsymbol{h}^{\mathrm{T}}(\boldsymbol{x}_k)\mathcal{N}(\hat{\boldsymbol{x}}_k^-, P_k^-)\mathrm{d}\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-\hat{\boldsymbol{y}}_k^- \tag{79}$$

$$K_k = P_{xy,k}(P_{yy,k})^{-1} \tag{80}$$

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + K_k(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k^-) \tag{81}$$

$$P_k = P_k^- - K_k P_{yy,k} K_k^{\mathrm{T}} \tag{82}$$

最后，时刻 k 的后验概率密度近似地为 $f(\boldsymbol{x}_k|\mathbf{Y}_1^k) = \mathcal{N}(\hat{\boldsymbol{x}}_k, P_k)$。

5.1 容积数值积分

考虑如下非线性函数的带权积分问题

$$I(\boldsymbol{g})=\int_{\mathbb{R}^{n_x}}\boldsymbol{g}(\boldsymbol{x})e^{-\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}}\mathrm{d}\boldsymbol{x} \tag{83}$$

式中 $I(\boldsymbol{g})$ 为所求积分，$\boldsymbol{x}\in\mathbb{R}^{n_x}$，$\mathbb{R}^{n_x}$ 为积分域，$\boldsymbol{g}(\boldsymbol{x})$ 为非线性函数 (映射)。式 (83) 的积分可使用基于容积原则的数值积分方法进行求解。

令 $\boldsymbol{x}=r\boldsymbol{s}$, $n=n_x$，取 $\boldsymbol{s}^{\mathrm{T}}\boldsymbol{s}=1$, 那么 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}=r^2, r\in[0,\infty)$，于是式 (83) 变换为球形-径向积分形式，即

$$I(\boldsymbol{g})=\int_0^{+\infty}\int_{U_n}\boldsymbol{g}(r\boldsymbol{s})r^{n-1}\exp(-r^2)\mathrm{d}\sigma(\boldsymbol{s})\mathrm{d}r \tag{84}$$

式中，$U_n=\left\{\boldsymbol{s}\in\mathbb{R}^n|\boldsymbol{s}^{\mathrm{T}}\boldsymbol{s}=1\right\}$ (球面)，$\sigma(\cdot)$ 为 U_n 上的面积单元。

令

$$S(r)=\int_{U_n}\boldsymbol{g}(r\boldsymbol{s})\mathrm{d}\sigma(\boldsymbol{s}) \tag{85}$$

式 (84) 可表示为

$$I(\boldsymbol{g})=\int_0^{+\infty}S(r)r^{n-1}\exp(-r^2)\mathrm{d}r \tag{86}$$

该式称为径向积分。

式 (85) 的球形积分可使用基于球形容积原则的数值积分方法近似为

$$S(r)=\int_{U_n}\boldsymbol{g}(r\boldsymbol{s})\mathrm{d}\sigma(\boldsymbol{s})=\sum_{j=1}^{m_{\boldsymbol{s}}}b_j\boldsymbol{g}(r\boldsymbol{s}_j) \tag{87}$$

式中 m_s 为积分点数。若使用三阶球形容积原则，相关参数为 $m_s = 2n_x$，$b_j = \frac{A_n}{2n}$，其中 $A_n = \frac{2\sqrt{\pi^n}}{\Gamma(n/2)}$，$\Gamma(n/2) = \int_0^{+\infty} x^{n-1}\exp(-x)\mathrm{d}x$。

$\boldsymbol{s}_j = [\mathbf{1}]_j$ 表示积分点。记 n_x 维单位向量为 $\boldsymbol{e} = [1, 0, \cdots 0]^{\mathrm{T}}$，使用 $[\mathbf{1}]$ 表示对 $\boldsymbol{e}$ 的元素进行全排列和改变元素符号产生的点集，称为完整全对称点集，$[\mathbf{1}]_j$ 表示点集中 $[\mathbf{1}]$ 的第 j 个点。

以 $n_x = 3$ 为例，$[\mathbf{1}]$ 表示的点集为

$$\left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \right\}$$

使用基于高斯积分原则的数值积分方法来近似计算径向积分 (86)，可表示为

$$I(\boldsymbol{g}) = \int_0^{+\infty} S(r) r^{n-1} \exp(-r^2) \mathrm{d}r = \sum_{i=1}^{m_r} a_i S(r_i) \tag{88}$$

式中 m_r 为积分点数。如果采用一阶高斯-赛德尔积分，那么 $m_r = 1$，$a_1 = \Gamma(n/2)/2$，$r_1 = \sqrt{n/2}$。

基于 (87) 和 (88) 可获得 (84) 的 3 阶球形-径向数值积分近似

$$\int_{\mathbb{R}^{n_x}} \boldsymbol{g}(\boldsymbol{x}) \exp(-\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \approx \sum_{i=1}^{2n_x} \frac{\sqrt{\pi^{n_x}}}{2n_x} \boldsymbol{g}(\sqrt{\frac{n_x}{2}}[\mathbf{1}]_i) \tag{89}$$

使用 (89) 式可求得标准正态分布函数的 3 阶球形-径向数值积分如下:

$$I_{\mathcal{N}}(\boldsymbol{g}) = \int_{\mathbb{R}^{n_x}} \boldsymbol{g}(\boldsymbol{x})\mathcal{N}_x(0, I)\mathrm{d}\boldsymbol{x} = \sum_{i=1}^{m} \omega_i \boldsymbol{g}(\boldsymbol{\xi}_i) \tag{90}$$

式中 $\boldsymbol{\xi}_i$ 为基本容积点，ω_i 为容积点对应的权值。$\boldsymbol{\xi}_i$ 和 ω_i 的取值分别为

$$\boldsymbol{\xi}_i = \sqrt{\frac{m}{2}}[\mathbf{1}]_i, \quad \omega_i = \frac{1}{m}, \quad i = 1, 2, \cdots, m = 2n_x \tag{91}$$

(90) 式中，权值为标准正态分布函数。对于一般的正态分布函数 $\mathcal{N}_x(\boldsymbol{\mu}, \Sigma)$，对应的非线性积分为

$$\int_{\mathbb{R}^{n_x}} \boldsymbol{g}(\boldsymbol{x})\mathcal{N}_x(\boldsymbol{\mu}, \Sigma)\mathrm{d}\boldsymbol{x} = \frac{1}{\sqrt{|(2\pi)^{n_x}\Sigma|}} \int_{\mathbb{R}^{n_x}} \boldsymbol{g}(\boldsymbol{x}) e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathrm{T}}\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}\mathrm{d}\boldsymbol{x} \tag{92}$$

分解 $\Sigma=\sqrt{\Sigma}\sqrt{\Sigma}^{\mathrm{T}}$，然后进行变换 $\boldsymbol{x}=\sqrt{\Sigma}\boldsymbol{y}+\boldsymbol{\mu}$，上式可化为

$$\begin{aligned}\int_{\mathbb{R}^{n_x}}\boldsymbol{g}(\boldsymbol{x})\mathcal{N}_x(\boldsymbol{\mu},\Sigma)\mathrm{d}\boldsymbol{x}&=\frac{1}{\sqrt{|(2\pi)^{n_x}\Sigma|}}\int_{\mathbb{R}^{n_x}}\boldsymbol{g}(\sqrt{\Sigma}\boldsymbol{y}+\boldsymbol{\mu})\sqrt{|\Sigma|}e^{-\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}}\mathrm{d}\boldsymbol{y}\\&=\int_{\mathbb{R}^{n_x}}\boldsymbol{g}(\sqrt{\Sigma}\boldsymbol{y}+\boldsymbol{\mu})\mathcal{N}_y(0,I)\mathrm{d}\boldsymbol{y}\end{aligned}\tag{93}$$

由 (90) 式，可得上式的 3 阶球形-径向数值积分近似解

$$\int_{\mathbb{R}^{n_x}}\boldsymbol{g}(\boldsymbol{x})\mathcal{N}_x(\boldsymbol{\mu},\Sigma)\mathrm{d}\boldsymbol{x}\approx\sum_{i=1}^{m}\omega_i\boldsymbol{g}(\sqrt{\Sigma}\boldsymbol{\xi}_i+\boldsymbol{\mu})\tag{94}$$

5.2 容积卡尔曼滤波算法 (Cubature Kalman Filter, CKF)

设时刻 $k-1$ 的后验概率密度为

$$f(\boldsymbol{x}_{k-1}|\,\mathbf{Y}_1^{k-1}) \simeq \mathcal{N}(\hat{\boldsymbol{x}}_{k-1}, P_{k-1})$$

容积卡尔曼滤波算法同样由时间修正和量测修正两部分组成。

5.2.1 时间修正

协方差矩阵平方根和容积点计算

$$S_{k-1} = \mathrm{chol}\{P_{k-1}\} \tag{95}$$

$$\boldsymbol{X}_{j,k-1} = S_{k-1}\boldsymbol{\xi}_j + \hat{\boldsymbol{x}}_{k-1} \tag{96}$$

式中，chol 表示矩阵的 Cholesky 分解。

容积点传播计算

$$\boldsymbol{X}^*_{j,k} = \boldsymbol{f}(\boldsymbol{X}_{j,k-1}) \tag{97}$$

一步预测及其协方差矩阵计算

$$\hat{\boldsymbol{x}}_k^- = \sum_{j=1}^{m} \omega_j \boldsymbol{X}_{j,k}^* \tag{98}$$

$$P_k^- = \sum_{j=1}^{m} \omega_j \boldsymbol{X}_{j,k}^* \boldsymbol{X}_{j,k}^{*\mathrm{T}} - \hat{\boldsymbol{x}}_k^- \hat{\boldsymbol{x}}_k^{-\mathrm{T}} + Q_{k-1} \tag{99}$$

5.2.2 量测修正

一步预测协方差矩阵平方根和容积点计算

$$S_k^- = \mathrm{chol}(P_k^-) \tag{100}$$

$$\boldsymbol{X}_{\boldsymbol{j},\boldsymbol{k}} = S_k^- \boldsymbol{\xi}_j + \hat{\boldsymbol{x}}_k^- \tag{101}$$

一步预测容积点传播计算

$$\boldsymbol{Z}_{j,k} = \boldsymbol{h}(\boldsymbol{X}_{j,k}) \tag{102}$$

量测一步预测、新息协方差矩阵及滤波增益计算

$$\hat{\boldsymbol{y}}_k^- = \sum_{j=1}^{m} \omega_j \boldsymbol{Z}_{j,k} \tag{103}$$

$$P_{yy,k} = \sum_{j=1}^{m} \omega_j \boldsymbol{Z}_{j,k} \boldsymbol{Z}_{j,k}^{\mathrm{T}} - \hat{\boldsymbol{y}}_k^- \hat{\boldsymbol{y}}_k^{-\mathrm{T}} + R_k \tag{104}$$

$$P_{xy,k} = \sum_{j=1}^{m} \omega_j \boldsymbol{X}_{j,k} \boldsymbol{Z}_{j,k}^{\mathrm{T}} - \hat{\boldsymbol{x}}_k^- \hat{\boldsymbol{y}}_k^{-\mathrm{T}} \tag{105}$$

$$K_k = P_{xy,k}[P_{yy,k}]^{-1} \tag{106}$$

滤波估计及其协方差矩阵计算

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + K_k(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k^-) \tag{107}$$

$$P_k = P_k - K_k P_{yy,k} K_k^{\mathrm{T}} \tag{108}$$

上述公式 (95) $\sim$ (108) 便构成了基本的容积卡尔曼滤波算法，初始条件为 $\hat{\boldsymbol{x}}_0 = \bar{\boldsymbol{x}}_0$, $P_0 = \mathrm{cov}(\boldsymbol{x}_0)$。

5.3 平方根容积卡尔曼滤波算法

- 容积卡尔曼滤波算法中，每一步滤波计算都需要对协方差矩阵进行 Cholesky 分解。为了提高算法的数值稳定性和鲁棒性，可建立平方根容积卡尔曼滤波算法。
- 若时刻 $k-1$ 的后验概率密度为 $f(\boldsymbol{x}_{k-1}|\mathbf{Y}_1^{k-1}) \simeq \mathcal{N}(\hat{\boldsymbol{x}}_{k-1}, P_{k-1})$，且获得了协方差矩阵 P_{k-1} 的平方根矩阵 S_{k-1}，即 $P_{k-1} = S_{k-1}S_{k-1}^{\mathrm{T}}$，则平方根容积卡尔曼滤波算法也由时间修正和量测修正两部分组织。

5.3.1 时间修正

容积点计算

$$\boldsymbol{X}_{j,k-1} = S_{k-1}\boldsymbol{\xi}_j + \hat{\boldsymbol{x}}_{k-1} \tag{109}$$

容积点传播计算

$$\boldsymbol{X}^*_{j,k} = \boldsymbol{g}(\boldsymbol{X}_{j,k-1}) \tag{110}$$

一步预测及其互协方差矩阵平方根计算

$$\hat{\boldsymbol{x}}_k^- = \sum_{i=1}^{m} \omega_i \boldsymbol{X}_{i,k}^* \tag{111}$$

$$S_k^- = \mathrm{Tria}([\chi_k^*, S_{Q_{k-1}}]) \tag{112}$$

式中 $S_{Q_{k-1}} = \mathrm{chol}(Q_{k-1})$，$\mathrm{Tria}()$ 表示对矩阵进行三角化，获得一方阵。矩阵 χ_k^* 定义为

$$\chi_k^* = m^{-\frac{1}{2}}[\boldsymbol{X}_{1,k}^* - \hat{\boldsymbol{x}}_k^-, \boldsymbol{X}_{2,k}^* - \hat{\boldsymbol{x}}_k^-, \cdots, \boldsymbol{X}_{m,k}^* - \hat{\boldsymbol{x}}_k^-] \tag{113}$$

5.3.2 量测修正

一步预测容积点计算

$$\boldsymbol{X}_{j,k} = S_k^- \boldsymbol{\xi}_j + \hat{\boldsymbol{x}}_k^- \tag{114}$$

一步预测容积点传播及量测一步预测计算

$$\boldsymbol{Z}_{j,k} = \boldsymbol{h}(\boldsymbol{X}_{j,k}) \tag{115}$$

$$\hat{\boldsymbol{y}}_k^- = \sum_{i=1}^{m} \omega_i \boldsymbol{Z}_{i,k} \tag{116}$$

新息协方差平方根及互协方差矩阵计算

$$S_{zz,k} = \mathrm{Tria}([A_k, S_{R_k}]) \tag{117}$$

$$P_{xy,k} = \chi_k A_k^{\mathrm{T}} \tag{118}$$

其中，$S_{R_k} = \mathrm{chol}(R_k)$，矩阵 χ_k 和 A_k 定义为

$$\chi_k = m^{-\frac{1}{2}}[\boldsymbol{X}_{1,k} - \hat{\boldsymbol{x}}_k^-, \boldsymbol{X}_{2,k} - \hat{\boldsymbol{x}}_k^-, \cdots, \boldsymbol{X}_{m,k} - \hat{\boldsymbol{x}}_k^-] \tag{119}$$

$$A_k = m^{-\frac{1}{2}}[\boldsymbol{Z}_{1,k} - \hat{\boldsymbol{y}}_k^-, \boldsymbol{Z}_{2,k} - \hat{\boldsymbol{y}}_k^-, \cdots, \boldsymbol{Z}_{m,k} - \hat{\boldsymbol{y}}_k^-] \tag{120}$$

滤波增益、状态估计及协方差矩阵平方根计算

$$K_k = (P_{xy,k}/S_{yy,k}^{\mathrm{T}})/S_{zz,k} \tag{121}$$

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + K_k(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k^-) \tag{122}$$

$$S_k = \mathrm{Tria}([\chi_k - K_k A_k, K_k S_{R_k}]) \tag{123}$$

式中，符号 “/” 为矩阵的右除运算符。

5.4 迭代容积卡尔曼滤波算法

定义代价函数

$$C(\boldsymbol{x}_k) = (\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-)^{\mathrm{T}}(P_k^-)^{-1}(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-) + (\boldsymbol{y}_k - \boldsymbol{h}(\hat{\boldsymbol{x}}_k^-))^{\mathrm{T}}R_k^{-1}(\boldsymbol{y}_k - \boldsymbol{h}(\hat{\boldsymbol{x}}_k^-)) \tag{124}$$

基于高斯-牛顿 (Gauss-Newton) 迭代算法，可导出状态的递推迭代求解公式 (求解 $C(\boldsymbol{x}_k)$ 极小点)

$$\hat{\boldsymbol{x}}_k^{(i+1)} = \hat{\boldsymbol{x}}_k^- + P_k^- H_k^{(i)\mathrm{T}}[H_k^{(i)} P_k^- H_k^{(i)\mathrm{T}} + R_k]^{-1}[\boldsymbol{y}_k - \boldsymbol{h}(\hat{\boldsymbol{x}}_k^{(i)}) - H_k^{(i)}(\hat{\boldsymbol{x}}_k^- - \hat{\boldsymbol{x}}_k^{(i)})] \tag{125}$$

其中，$H_k^{(i)} = \left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}_k)}{\partial \boldsymbol{x}_k}\right|_{x_k=\hat{\boldsymbol{x}}_k^{(i)}}$，$(H_k^{(i)} P_k^- H_k^{(i)\mathrm{T}} + R_k)^{-1}$ 和 $P_k^- H_k^{(i)\mathrm{T}}$ 是量测方程线性化后获得的近似协方差和互协方差矩阵。即

$$P_{yy}^{(i)} = H_k^{(i)} P_k^- H_k^{(i)\mathrm{T}} + R_k$$
$$P_{xy}^{(i)} = P_k^- H_k^{(i)\mathrm{T}}$$

由此可近似获得

$$H_k^{(i)} = (P_{xy}^{(i)})^{\mathrm{T}} (P_k^-)^{-1} \tag{126}$$

Remark 5.1 *(125)* 式的迭代计算仅考虑了泰勒级数的线性项，对于高度非线性的量测方程，会增大估计误差。

Remark 5.2 结合迭代计算、统计线性化和容积卡尔曼滤波算法，可建立下述迭代容积卡尔曼滤波算法（*ICKF*），从而提高状态估计的精度。

Remark 5.3 和 *IEKF* 类似，在迭代容积卡尔曼滤波算法中，时间修正步骤保持不变，即假设已经获得了状态的一步预测及其协方差矩阵 $\hat{\boldsymbol{x}}_k^-$ 和 P_k^-。

假设第 i 次迭代的状态估计及其协方差矩阵分别为 $\hat{\boldsymbol{x}}_k^{(i)}$ 和 $P_k^{(i)}$，记

$$
\begin{aligned}
S_k^{(i)} &= \mathrm{chol}(P_k^{(i)}) \\
\boldsymbol{X}_{j,k}^{(i)} &= \hat{S}_k^{(i)} \xi_j + \hat{\boldsymbol{x}}_k^{(i)} \\
\boldsymbol{Z}_{j,k}^{(i)} &= \boldsymbol{h}(\boldsymbol{X}_{j,k}^{(i)})
\end{aligned}
$$

那么

$$\hat{\boldsymbol{y}}_k^{(i)} = \sum_{j=1}^{m} \omega_j \boldsymbol{Z}_{j,k}^{(i)} \tag{127}$$

$$P_{yy,k}^{(i)} = \sum_{j=1}^{m} \omega_j \boldsymbol{Z}_{j,k}^{(i)} (\boldsymbol{Z}_{j,k}^{(i)})^{\mathrm{T}} - \hat{\boldsymbol{y}}_k^{(i)} (\hat{\boldsymbol{y}}_k^{(i)})^{\mathrm{T}} + R_k$$

$$P_{xy,k}^{(i)} = \sum_{j=1}^{m} \omega_j \boldsymbol{X}_{j,k}^{(i)} (\boldsymbol{Z}_{j,k}^{(i)})^{\mathrm{T}} - \hat{\boldsymbol{x}}_k^{(i)} (\hat{\boldsymbol{y}}_k^{(i)})^{\mathrm{T}}$$

$$K_k^{(i)} = P_{xy,k}^{(i)} (P_{yy,k}^{(i)})^{-1}$$

$$\hat{\boldsymbol{x}}_k^{(i+1)} = \hat{\boldsymbol{x}}_k^{-} + K_k^{(i)} [\boldsymbol{y}_k - \boldsymbol{h}(\hat{\boldsymbol{x}}_i) - (P_{xy,k}^{(i)})^{\mathrm{T}} (P_k^{-})^{-1} (\hat{\boldsymbol{x}}_k^{-} - \hat{\boldsymbol{x}}_k^{(i)})] \tag{128}$$

$$P_k^{(i+1)} = P_k^{-} - K_k^{(i)} (P_{yy,k}^{(i)}) (K_k^{(i)})^{\mathrm{T}} \tag{129}$$

$$(i = 0, 1, 2, \cdots, N_{\max})$$

迭代终止条件为

$$\left\| \boldsymbol{x}_k^{(i+1)} - \boldsymbol{x}_k^{(i)} \right\| \leq \varepsilon \quad \text{或} \quad i = N_{\max} \tag{130}$$

式中 ε 和 $N_{\max}$ 为预先设置的阈值和最大迭代次数。

若迭代终止时迭代次数为 N，时刻 k 的状态估计及其协方差矩阵分别为

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^{(N)} \tag{131}$$

$$P_k = P_k^{(N)} \tag{132}$$

6. 粒子滤波

- ♢ EKF 估计非线性系统均值的精度是 1 阶的，而 UKF 可以到达 2 阶以上。当系统非线性非常严重时，两者都可能出现滤波发散问题。
- ♢ CKF 隐含假设了系统状态时刻都近似满足高斯分布，对于严重非线性的系统或噪声本身就是非高斯的系统，该假设显然不太容易满足。
- ♢ 粒子滤波（Particle Filter, PF）方法可以有效地克服 EKF 等可能遇到的滤波发散问题。
- ♢ 粒子滤波是一种“暴力”滤波方法，它以计算代价换取滤波性能。
- ♢ 粒子滤波又称为序贯重要性采样、蒙特-卡罗滤波等。

6.1 粒子滤波的基本思想

考虑一般的非线性系统

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}_k(\boldsymbol{x}_k, \boldsymbol{w}_k) \tag{133}$$

$$\boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{x}_k, \boldsymbol{v}_k) \tag{134}$$

假设 $\{\boldsymbol{w}_k\}$ 和 $\{\boldsymbol{v}_k\}$ 是相互独立的纯随机序列；$\boldsymbol{x}_0 \sim f_{x_0}(\boldsymbol{x}_0) = f(\boldsymbol{x}_0) = f(\boldsymbol{x}_0|\boldsymbol{y}_0)$，而且与 $\{\boldsymbol{w}_k\}$ 和 $\{\boldsymbol{v}_k\}$ 相互独立。

仍然记 $\mathbf{Y}_1^k = (\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_k)$，那么

$$f(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k) = \int f(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k) f(\boldsymbol{x}_k|\mathbf{Y}_1^k) \mathrm{d}\boldsymbol{x}_k \tag{135}$$

$$f(\boldsymbol{y}_{k+1}|\mathbf{Y}_1^k) = \int f(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1}) f(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k) \mathrm{d}\boldsymbol{x}_{k+1} \tag{136}$$

$$f(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^{k+1}) = \frac{f(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1}) f(\boldsymbol{x}_{k+1}|\mathbf{Y}_1^k)}{f(\boldsymbol{y}_{k+1}|\mathbf{Y}_1^k)} \tag{137}$$

以上三式便构成了递推求解后验概率密度 $f(\boldsymbol{x}_k|\mathbf{Y}_1^k)$ 的核心公式，称为贝叶斯递推滤波算法。一般情况下，通常无法获得解析解。对于复杂的非线性系统，求解其中的高维非线性积分是贝叶斯递推滤波的本质困难。

在 Monte Carlo (MC) 仿真中，以若干足够多的离散样本（称为粒子）对期望的概率密度进行近似，即

$$f(\boldsymbol{x}_k|\mathbf{Y}_1^k) \approx \hat{f}(\boldsymbol{x}_k|\mathbf{Y}_1^k) = \sum_{i=1}^{N} w_k^{(i)} \delta(\boldsymbol{x} - \boldsymbol{x}_k^{(i)}) \tag{138}$$

其中，$w_k^{(i)} \geq 0$，而且 $\sum_{i=1}^{N} w_k^{(i)} = 1$。考虑到 δ 函数的宝贵性质，上式为求解贝叶斯递推滤波所面临的高维非线性积分提供了技术途径，也是粒子滤波的基本出发点。

如何生成粒子 $\{\boldsymbol{x}_k^{(i)}\}_{i=1,2,\cdots,N}$ 并确定 $\{w_k^{(i)}\}_{i=1,2,\cdots,N}$，这就是粒子滤波的基本问题。

6.2 蒙特卡罗采样技术

如上所述，粒子滤波的本质是蒙特-卡罗仿真，力图通过大量的样本（粒子）来近似需要的概率密度函数。这里我们分别介绍其中的完备采样 (perfect sampling)、重要性采样 (importance sampling) 及重采样 (resampling) 等技术。

6.2.1 完备采样

对于目标密度函数 $f(\boldsymbol{x})$，设 $\{\boldsymbol{x}^{(i)}, i = 1, 2, \cdots, N\}$ 是根据 $f(\boldsymbol{x})$ 采样到的独立同分布粒子，那么

$$f(\boldsymbol{x}) \approx \hat{f}(\boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} \delta(\boldsymbol{x} - \boldsymbol{x}^{(i)}) \tag{139}$$

上述根据给定的概率密度函数获取大量独立同分布的粒子，并以此构造便于积分计算的概率密度函数近似表达式，即称为完备采样。

通过完备采样，显然有

$$\bar{\boldsymbol{x}} = \int \boldsymbol{x} f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \approx \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}^{(i)}$$

$$\mathrm{var}(\boldsymbol{x}) = \int (\boldsymbol{x} - \bar{\boldsymbol{x}})(\boldsymbol{x} - \bar{\boldsymbol{x}})^{\mathrm{T}} f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \approx \frac{1}{N} \sum_{i=1}^{N} (\boldsymbol{x}^{(i)} - \bar{\boldsymbol{x}})(\boldsymbol{x}^{(i)} - \bar{\boldsymbol{x}})^{\mathrm{T}}$$

更加一般地，对于比较任意的函数 $\boldsymbol{g}(\cdot)$，有

$$E\boldsymbol{g}(\boldsymbol{x}) \approx \hat{E}_N \boldsymbol{g}(\boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{g}(\boldsymbol{x}^{(i)}) \tag{140}$$

理论上

$$\lim_{N \to +\infty} \hat{E}_N \boldsymbol{g}(\boldsymbol{x}) \xrightarrow{a.s.} E\boldsymbol{g}(\boldsymbol{x}) \tag{141}$$

对于非线性滤波问题，几乎不可能直接从密度函数 $f(\boldsymbol{x}_k|\mathbf{Y}_1^k)$ 进行采样(即完备采样)，需要下述重要性采样方法。

6.2.2 重要性采样

通常，目标密度函数 $f(\boldsymbol{x})$ 非常复杂，不容易直接产生符合密度函数 $f(\boldsymbol{x})$ 的粒子。设 $q(\boldsymbol{x})$ 是较之 $f(\boldsymbol{x})$ 容易实现采样的概率分布密度函数，在支撑覆盖条件下，如果 $\{\boldsymbol{x}^{(i)}, i = 1, 2, \cdots, N\} \sim q(\boldsymbol{x})$，那么粒子 $\boldsymbol{x}^{(i)}$ 属于 $f(\boldsymbol{x})$ 的概率为 $w^{(i)}$，称为接受概率，有

$$f(\boldsymbol{x}^{(i)}) \propto w^{(i)} q(\boldsymbol{x}^{(i)}) \tag{142}$$

上式可以简单论证如下：对于任意的函数 $g(\boldsymbol{x})$，有

$$
\begin{aligned}
Eg(\boldsymbol{x}) &= \int g(\boldsymbol{x}) f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \int [g(\boldsymbol{x}) \frac{f(\boldsymbol{x})}{q(\boldsymbol{x})}] q(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\
&\approx \frac{1}{N} \sum_{i=1}^{N} g(\boldsymbol{x}^{(i)}) \frac{f(\boldsymbol{x}^{(i)})}{q(\boldsymbol{x}^{(i)})} = \sum_{i=1}^{N} w^{(i)} g(\boldsymbol{x}^{(i)})
\end{aligned}
$$

上式定义了 $w^{(i)}$，并验证了 (142) 式。换言之，我们有

$$
f(\boldsymbol{x}) \approx \sum_{i=1}^{N} w^{(i)} \delta(\boldsymbol{x} - \boldsymbol{x}^{(i)}) \tag{143}
$$

上式说明了 $w^{(i)}$ 为接受概率的含义，同时暗喻着 $\sum_{i=1}^{N} w^{(i)} = 1$。

6.2.3 重采样

重采样（resampling）是指根据（143）式进行重新采样，获取若干个近似服从 $f(\boldsymbol{x})$ 分布的粒子。有许多重采样方法，它决定了粒子滤波算法的计算复杂度。

对于目标密度函数的估计

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{N} \bar{w}^{(i)} \delta(\boldsymbol{x} - \boldsymbol{x}^{(i)}) \tag{144}$$

以 $\bar{w}^{(i)}$ 为接受概率重新产生 N 个粒子，即（离散概率密度）

$$\Pr(\hat{\boldsymbol{x}}^{(i)} = \boldsymbol{x}^{(j)}) = \bar{w}^{(j)}, \quad i = 1, 2, \cdots, N \tag{145}$$

重采样后，目标密度函数近似为

$$\tilde{f}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N}\delta(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(i)}) \tag{146}$$

重采样过程如图2所示。显然，重采样可能带来粒子贫化的问题。

常用的重采样方法有简单随机采样和系统采样法 (systematic sampling) 两种，前者简单、明了，后者计算效率更高。

简单随机采样（轮盘赌算法）

取 $i = 1, 2, \cdots, N$，执行如下两步:

(1) 生成随机数 $r \sim U[0, 1]$（均匀分布）;

(2) 逐个累加 $w^{(i)}$，直到大于 r，即 $\sum_{i=1}^{j} w^{(i)} \geq r$ 且 $\sum_{i=1}^{j-1} < r$，置旧粒子 $\boldsymbol{x}^{(j)}$ 为重采样粒子 $\hat{\boldsymbol{x}}^{(j)}$。

系统采样法

(1) 根据下式生成 N 个随机数:

$u_i = \frac{(i-1)+r}{N}, r \sim U[0,1]$

(2) 如果 $\sum_{j=1}^{m-1} \bar{w}^{(j)} < u_i \leq \sum_{j=1}^{m} \bar{w}^{(j)}$，直接拷贝 m 个粒子 $\{\boldsymbol{x}^{(i)}\}$ 为重采样粒子 $\{\hat{\boldsymbol{x}}^{(i)}\}$。

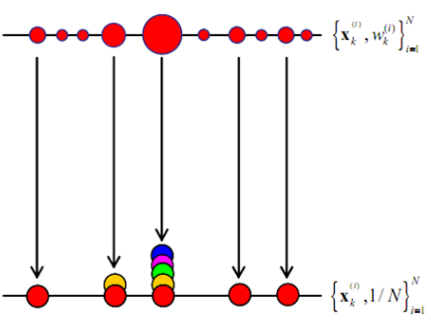

Figure 2: 重采样示意图

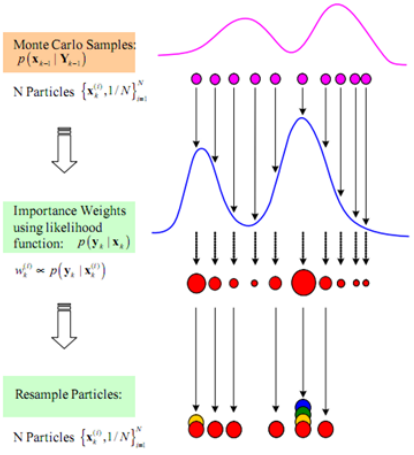

Figure 3: 粒子滤波示意图

6.3 粒子滤波算法

粒子滤波是贝叶斯递推滤波算法的一种数值实现。首先根据系统状态的初始概率密度函数 $f(\boldsymbol{x}_0)$（假设是已知的）产生 N 个状态向量（称为粒子），即 $\boldsymbol{x}_{0|0}^{(i)} \sim f(\boldsymbol{x}_0)(i = 1, 2, \cdots, N)$。

在每一时刻 $k(\geq 1)$，按照状态方程传播粒子，即

$$\boldsymbol{x}_{k|k-1}^{(i)} = \boldsymbol{g}_k(\boldsymbol{x}_{k-1|k-1}^{(i)}, \boldsymbol{w}_{k-1}^{(i)}), \qquad i = 1, 2, \cdots, N \tag{147}$$

式中，$\boldsymbol{w}_{k-1}^{(i)}$ 是根据过程噪声统计特性生成的噪声向量。上式表明，我们获得了服从 $f(\boldsymbol{x}_{k|k-1}) = f(\boldsymbol{x}_k|\mathbf{Y}_1^{k-1})$ 分布的 N 个粒子，由此可以计算一步预测的均值和方差等。换句话说，只要 N 足够大，由完全采样可知

$$f(\boldsymbol{x}_{k|k-1}) \approx \hat{f}(\boldsymbol{x}_{k|k-1}) = \frac{1}{N}\sum_{i=1}^{N}\delta(\boldsymbol{x}_{k|k-1} - \boldsymbol{x}_{k|k-1}^{(i)}) \tag{148}$$

当获得新的量测值 $\boldsymbol{y}_k$ 后，$f(\boldsymbol{x}_{k|k})$ 的近似估计可以表达为

$$\hat{f}(\boldsymbol{x}_{k|k}) = \sum_{i=1}^{N} w^{(i)}\delta(\boldsymbol{x}_{k|k} - \boldsymbol{x}_{k|k-1}^{(i)}) \tag{149}$$

其中 $w^{(i)} = \Pr(\boldsymbol{x}_k = \boldsymbol{x}_{k|k-1}^{(i)}|\boldsymbol{y}_k)$，即对 (148) 中 $1/N$ 的修正。

由于

$$\Pr(\boldsymbol{x}_k = \boldsymbol{x}_{k|k-1}^{(i)}|\boldsymbol{y}_k) = \frac{\Pr(\boldsymbol{y}_k|\boldsymbol{x}_k = \boldsymbol{x}_{k|k-1}^{(i)})\Pr(\boldsymbol{x}_k = \boldsymbol{x}_{k|k-1}^{(i)})}{\Pr(\boldsymbol{y}_k)}$$

因此

$$w^{(i)} \propto \Pr(\boldsymbol{y}_k|\boldsymbol{x}_k = \boldsymbol{x}_{k|k-1}^{(i)})$$

总结上面的讨论，根据量测方程计算每个粒子 $\boldsymbol{x}_{k|k-1}^{(i)}$ 的条件似然值，并归一化，即

$$w_k^{(i)} = f(\boldsymbol{y}_k|\boldsymbol{x}_{k|k-1}^{(i)}) \tag{150}$$

$$\bar{w}_k^{(i)} = \frac{w_k^{(i)}}{\sum_{i=1}^{N} w_k^{(i)}}, i = 1, 2, \cdots, N \tag{151}$$

那么

$$\hat{f}(\boldsymbol{x}_{k|k}) = \sum_{i=1}^{N} \bar{w}_k^{(i)} \delta(\boldsymbol{x}_{k|k} - \boldsymbol{x}_{k|k-1}^{(i)}) \tag{152}$$

为了下一步递推计算，需要进行重采样。重采样可以认为按下式获得 N 个新的粒子 $\{\boldsymbol{x}_{k|k}^{(i)}\}_{i=1}^{N}$:

$$\Pr(\boldsymbol{x}_{k|k}^{(i)} = \boldsymbol{x}_{k|k-1}^{(j)}) = \bar{w}_k^{(j)}, \quad i = 1, 2, \cdots, N \tag{153}$$

重采样后粒子权重均为 $1/N$。上述粒子滤波算法示意图见图3，实现步骤见算法1和算法2列表。

♠ 这里的粒子滤波算法实现中的输出均采用了最常规的后验概率均值，也可以采用后验概率对应的中值、模值等。

♠ 对于加性噪声情况，重要性权重计算和预测粒子计算可以简化：

♡ 如果量测噪声是加性噪声，即 $\boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{v}_k$，那么

$$w_k^{(i)} = f(\boldsymbol{y}_k|\boldsymbol{x}_{k|k-1}^{(i)}) = f_{\boldsymbol{v}_k}(\boldsymbol{y}_k - h(\boldsymbol{x}_{k|k-1}^{(i)})) \tag{154}$$

♡ 如果过程噪声是加性的，那么

$$\boldsymbol{x}_{k|k-1}^{(i)} = \boldsymbol{g}_k(\boldsymbol{x}_{k-1|k-1}^{(i)}) + \boldsymbol{w}_{k-1}^{(i)} \tag{155}$$

其中噪声粒子 $\boldsymbol{w}_{k-1}^{(i)} \sim f_{\boldsymbol{w}_{k-1}}(\boldsymbol{w}_{k-1})$。

Algorithm 1 粒子滤波算法 1

初始化: $\{\boldsymbol{x}_{0|0}^{(i)}\}_{i=1}^{N} \sim f_{\boldsymbol{x}_0}(\boldsymbol{x}_0)$，置 $k=1$;

1: **while** {实验进行中} **do**

2: 预测: $\boldsymbol{x}_{k|k-1}^{(i)} \sim f(\boldsymbol{x}_k|\boldsymbol{x}_{k-1|k-1}^{(i)})$，$i=1,2,\cdots,N$;

3: 滤波:

a. 计算重要性权重: $w_k^{(i)} = f(\boldsymbol{y}_k|\boldsymbol{x}_{k|k-1}^{(i)})$，$i=1,2,\cdots,N$;

b. 归 1 化得到接受概率: $\bar{w}_k^{(i)} = \dfrac{w_k^{(i)}}{\sum\limits_{i=1}^{N} w_k^{(i)}}$，$i=1,2,\cdots,N$;

c. 输出 (根据需要):

$$\hat{\boldsymbol{x}}_{k|k} = \sum_{i=1}^{N} \bar{w}_k^{(i)} \boldsymbol{x}_{k|k-1}^{(i)}$$

$$P_{k|k} = \sum_{i=1}^{N} \bar{w}_k^{(i)} (\boldsymbol{x}_{k|k-1}^{(i)} - \hat{\boldsymbol{x}}_{k|k})(\boldsymbol{x}_{k|k-1}^{(i)} - \hat{\boldsymbol{x}}_{k|k})^{\mathrm{T}}$$

d. 根据 $\Pr(\boldsymbol{x}_{k|k}^{(i)} = \boldsymbol{x}_{k|k-1}^{(j)}) = \bar{w}_k^{(j)}$ 重采样 N 个新的粒子 $\boldsymbol{x}_{k|k}^{(i)}$;

4: 置 $k := k+1$，或实验结束。

5: **end while**

Algorithm 2 粒子滤波算法 2

初始化: $\{\boldsymbol{x}_{0|-1}^{(i)}\}_{i=1}^{N} \sim f_{\boldsymbol{x}_0}(\boldsymbol{x}_0)$，置 $k=0$;

1: **while** {实验进行中} **do**

2: 滤波:

a. 计算重要性权重: $w_k^{(i)} = f(\boldsymbol{y}_k|\boldsymbol{x}_{k|k-1}^{(i)})$，$i=1,2,\cdots,N$;

b. 归 1 化得到接受概率: $\bar{w}_k^{(i)} = \dfrac{w_k^{(i)}}{\sum\limits_{i=1}^{N} w_k^{(i)}}$，$i=1,2,\cdots,N$;

c. 输出 (根据需要):

$$\hat{\boldsymbol{x}}_{k|k} = \sum_{i=1}^{N} \bar{w}_k^{(i)} \boldsymbol{x}_{k|k-1}^{(i)}$$

$$P_{k|k} = \sum_{i=1}^{N} \bar{w}_k^{(i)} (\boldsymbol{x}_{k|k-1}^{(i)} - \hat{\boldsymbol{x}}_{k|k})(\boldsymbol{x}_{k|k-1}^{(i)} - \hat{\boldsymbol{x}}_{k|k})^{\mathrm{T}}$$

3: 重采样: 根据 $\Pr(\boldsymbol{x}_{k|k}^{(i)} = \boldsymbol{x}_{k|k-1}^{(j)}) = \bar{w}_k^{(j)}$ 重采样 N 个新的粒子 $\boldsymbol{x}_{k|k}^{(i)}$;

4: 预测: $\boldsymbol{x}_{k+1|k}^{(i)} \sim f(\boldsymbol{x}_{k+1}|\boldsymbol{x}_{k|k}^{(i)})$，$i=1,2,\cdots,N$;

5: 置 $k := k+1$，或实验结束。

6: **end while**

6.4 改进粒子滤波

6.4.1 正则化粒子滤波

粒子滤波除了计算量大外，主要还存在粒子退化和粒子贫化问题。

粒子退化: 少量粒子的重要性权重变得很大，而若干粒子的重要性权重变得很小，甚至趋于 0。

粒子贫化: 进入下一步滤波计算的不同粒子数目越来越少，使得粒子的多样性越来越差。

正则化粒子滤波（Regularied Particle Filtering, RPF）用连续的概率密度函数代替前面粒子滤波算法中的离散概率密度函数，从而保证重采样后的粒子具有多样性。

回顾前面讨论的粒子滤波算法，重采样本质是依据概率密度函数

$$\hat{f}(\boldsymbol{x}_{k|k}) = \sum_{i=1}^{N} \bar{w}_k^{(i)} \delta(\boldsymbol{x}_{k|k} - \boldsymbol{x}_{k|k-1}^{(i)})$$

进行采样。而 RPF 中，将上式修改为如下连续函数

$$\hat{f}(\boldsymbol{x}_{k|k}) = \sum_{i=1}^{N} \bar{w}_k^{(i)} K_h(\boldsymbol{x}_{k|k} - \boldsymbol{x}_{k|k-1}^{(i)}) \tag{156}$$

其中

$$K_h(\boldsymbol{x}) = h^{-n}K(\boldsymbol{x}/h) \tag{157}$$

h 称为带宽，是一个设计参数；$K(\boldsymbol{x})$ 为一个对称的核函数，满足如下条件

$$\int \boldsymbol{x}K(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = 0 \tag{158}$$

$$\int ||\boldsymbol{x}||_2^2 K(\boldsymbol{x})\mathrm{d}\boldsymbol{x} < \infty \tag{159}$$

可用的核函数很多，例如高斯函数、三角函数、窗口函数等，都是可能的选择。

6.4.2 组合粒子滤波

粒子滤波可以与其他滤波方法组合起来使用，从而可以改善滤波的效果。例如，可以将 PF 与 EKF 组合，也可以将 PF 与 UKF 组合，前者可以称为 EKPF，后者可以称为 UKPF 或 SPPF。

EKPF 基本原理如下：

(1) 对每个粒子进行 EKF：

$$[\hat{\boldsymbol{x}}_{k|k}^{(i)}, P_{k|k}^{(i)}] = \mathrm{EKF}(\hat{\boldsymbol{x}}_{k-1|k-1}^{(i)}, P_{k-1|k-1}^{(i)}), i = 1, 2, \cdots, N$$

(2) 计算重采样概率：

$$w_k^{(i)} = f(\boldsymbol{y}_k|\hat{\boldsymbol{x}}_{k|k}^{(i)}), \bar{w}_k^{(i)} = \frac{w_k^{(i)}}{\sum_{i=1}^{N} w_k^{(i)}}$$

(3) 重采样:

$$[\hat{\boldsymbol{x}}_{k|k}^{(i)}, P_{k|k}^{(i)}] = \mathrm{Resample}[\hat{\boldsymbol{x}}_{k|k}^{(i)}, \bar{\boldsymbol{w}}_k^{(i)}], i = 1, 2, \cdots, N$$

将上面的 EKF 换为 UKF 便是 SPPF 的原理。

7. 递推克拉默-拉奥下界

如果 $\boldsymbol{\theta}=[\theta_1,\theta_2,\cdots,\theta_r]^{\mathrm{T}}$ 是一个 r 维的待估参数，假设 $\hat{\boldsymbol{\theta}}$ 是基于量测数据 $\boldsymbol{y}$ 的无偏估计。那么

$$P=E\{[\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}][\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}]^{\mathrm{T}}\}\geq J^{-1} \tag{160}$$

式中，Fisher 信息矩阵 J 的维数为 $r\times r$。

若 $\boldsymbol{\theta}$ 还是随机的，那么

$$[J_{ij}]=\Big[-E\frac{\partial^2\log f_{y,\theta}(Y,\Theta)}{\partial\theta_i\partial\theta_j}\Big],\quad i,j=1,\cdots,r \tag{161}$$

式中，$f_{y,\theta}(Y,\Theta)$ 为 $(\boldsymbol{y},\boldsymbol{\theta})$ 的联合概率分布密度。

引入如下一阶和二阶算子 ∇ 和 Δ

$$\nabla_\theta = [\frac{\partial}{\partial\theta_1}, \cdots, \frac{\partial}{\partial\theta_r}]^{\mathrm{T}}$$

$$\Delta_\psi^\theta = \nabla_\psi \nabla_\theta^{\mathrm{T}} = \begin{bmatrix} \frac{\partial^2}{\partial\psi_1\partial\theta_1} & \frac{\partial^2}{\partial\psi_1\partial\theta_2} & \cdots & \frac{\partial^2}{\partial\psi_1\partial\theta_r} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2}{\partial\psi_r\partial\theta_1} & \frac{\partial^2}{\partial\psi_r\partial\theta_2} & \cdots & \frac{\partial^2}{\partial\psi_r\partial\theta_r} \end{bmatrix}$$

那么 Fisher 信息矩阵 J 可表示为

$$J = E[-\Delta_\theta^\theta \log f_{y,\theta}(Y, \Theta)] \tag{162}$$

若将 $\boldsymbol{\theta}$ 分为两个部分 $\boldsymbol{\theta} = [\boldsymbol{\theta}_\alpha^{\mathrm{T}}, \boldsymbol{\theta}_\beta^{\mathrm{T}}]^{\mathrm{T}}$，信息矩阵 J 相应地写为 $J =$

$\begin{bmatrix} J_{\alpha\alpha} & J_{\alpha\beta} \\ J_{\beta\alpha} & J_{\beta\beta} \end{bmatrix}$，那么

$$P_\beta = E\{[\hat{\boldsymbol{\theta}}_\beta - \boldsymbol{\theta}_\beta][\hat{\boldsymbol{\theta}}_\beta - \boldsymbol{\theta}_\beta]^{\mathrm{T}}\} \geq [J_{\beta\beta} - J_{\beta\alpha}J_{\alpha\alpha}^{-1}J_{\alpha\beta}]^{-1} \tag{163}$$

若 $J_{\alpha\alpha}^{-1}$ 存在，则矩阵 $J_{\beta\beta} - J_{\beta\alpha}J_{\alpha\alpha}^{-1}J_{\alpha\beta}$ 称为参数 $\boldsymbol{\theta}_\beta$ 的信息矩阵。

令 k 时刻所有状态为 $\boldsymbol{X}_k = (\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_k)$，所有量测为 $\boldsymbol{Y}_k = (\boldsymbol{y}_0, \boldsymbol{y}_1, \cdots, \boldsymbol{y}_k)$。记 $J(\boldsymbol{X}_k)$ 为状态集合 $\boldsymbol{X}_k$ 的信息矩阵，J_k 为状态 $\boldsymbol{x}_k$ 的信息矩阵，则其联合状态过程为马尔可夫过程。联合概率密度函数 $f(\boldsymbol{X}_k, \boldsymbol{Y}_k)$ 满足等式

$$f(\boldsymbol{X}_k, \boldsymbol{Y}_k) = f(\boldsymbol{x}_0)\prod_{i=1}^{k} f(\boldsymbol{y}_i|\,\boldsymbol{x}_i)\prod_{j=1}^{k} f(\boldsymbol{x}_j|\,\boldsymbol{x}_{j-1}) \tag{164}$$

将 $\boldsymbol{X}_k$ 写成 $\boldsymbol{X}_k=[\boldsymbol{X}_{k-1}^{\mathrm{T}},\boldsymbol{x}_k^{\mathrm{T}}]^{\mathrm{T}}$，则 $J(\boldsymbol{X}_k)$ 为

$$J(\boldsymbol{X}_k)=\begin{bmatrix}A_k & B_k\\ B_k^{\mathrm{T}} & C_k\end{bmatrix}=\begin{bmatrix}E\{\Delta_{X_{k-1}}^{X_{k-1}}\log f_k\} & E\{\Delta_{X_{k-1}}^{x_k}\log f_k\}\\ E\{\Delta_{x_k}^{X_{k-1}}\log f_k\} & E\{\Delta_{x_k}^{x_k}\log f_k\}\end{bmatrix} \tag{165}$$

式中 $f_k=f(\boldsymbol{X}_k,\boldsymbol{Y}_k)$。根据 (163) 式，可知

$$J_k=C_k-B_k^{\mathrm{T}}A_k^{-1}B_k \tag{166}$$

$\boldsymbol{X}_{k+1}$ 的信息矩阵 $J(\boldsymbol{X}_{k+1})$ 为

$$J(\boldsymbol{X}_{k+1})=\begin{bmatrix}E\{-\Delta_{X_k}^{X_k}\log f_{k+1}\} & E\{-\Delta_{X_k}^{x_{k+1}}\log f_{k+1}\}\\ E\{-\Delta_{x_{k+1}}^{X_k}\log f_{k+1}\} & E\{-\Delta_{x_{k+1}}^{x_{k+1}}\log f_{k+1}\}\end{bmatrix} \tag{167}$$

式中 f_{k+1} 为 $\boldsymbol{X}_{k+1}$ 和 $\boldsymbol{Y}_{k+1}$ 的联合概率密度函数。因此可得

$$-\log f_{k+1} = -\log f_k - \log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k) - \log f(\boldsymbol{y}_{k+1}|\,\boldsymbol{x}_{k+1})$$

把上式带入 (167) 式可得

$$J(\boldsymbol{X}_{k+1}) = \begin{bmatrix} E\{-\Delta_{X_k}^{X_k}\log f_{k+1}\} & E\{-\Delta_{X_k}^{x_{k+1}}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\} \\ E\{-\Delta_{x_{k+1}}^{X_k}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\} & E\{-\Delta_{x_{k+1}}^{x_{k+1}}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{X}_k) + E\{-\Delta_{x_{k+1}}^{x_{k+1}}\log f(\boldsymbol{y}_{k+1}|\,\boldsymbol{x}_k)\} \end{bmatrix} \tag{168}$$

式中

$$E\{-\Delta_{X_k}^{X_k}\log f_{n+1}\}=$$

$$\begin{bmatrix} E\{-\Delta_{X_{k-1}}^{X_k}\log f_k\} & E\{-\Delta_{X_{k-1}}^{x_k}\log f_k\} \\ E\{-\Delta_{x_k}^{X_{k-1}}\log f_k\} & E\{-\Delta_{x_k}^{x_k}\log f_k\}+E\{-\Delta_{x_k}^{x_k}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\} \end{bmatrix}$$

$$E\{-\Delta_{X_k}^{x_{k+1}}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\}=\begin{bmatrix} 0 \\ E\{-\Delta_{x_k}^{x_{k+1}}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\} \end{bmatrix}$$

$$E\{-\Delta_{x_{k+1}}^{X_k}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\}=\begin{bmatrix} 0 & E\{-\Delta_{x_{k+1}}^{x_k}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\} \end{bmatrix}$$

记

$$
\begin{aligned}
D_k^{11} &= E\{-\Delta_{x_k}^{x_k}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\}\\
D_k^{12} &= E\{-\Delta_{x_k}^{x_{k+1}}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\}\\
D_k^{21} &= E\{-\Delta_{x_{k+1}}^{x_k}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\} = [D_k^{12}]^{\mathrm{T}}\\
D_k^{22} &= E\{-\Delta_{x_{k+1}}^{x_{k+1}}\log f(\boldsymbol{x}_{k+1}|\,\boldsymbol{x}_k)\} + E\{-\Delta_{x_{k+1}}^{x_{k+1}}\log f(\boldsymbol{z}_{k+1}|\,\boldsymbol{x}_{k+1})\}
\end{aligned}
$$

由此可得

$$
J(\boldsymbol{X}_{k+1}) = \begin{bmatrix} A_k & B_k & 0\\ B_k^{\mathrm{T}} & C_k + D_k^{11} & D_k^{12}\\ 0 & D_k^{21} & D_k^{22}\end{bmatrix} \tag{169}
$$

根据 (163) 式，可推出 $\boldsymbol{x}_{k+1}$ 的信息子矩阵 J_{k+1} 的表达式为

$$\begin{aligned} J_{k+1} &= D_k^{22} - \begin{bmatrix} 0 & D_k^{21} \end{bmatrix} \begin{bmatrix} A_k & B_k \\ B_k^T & C_k + D_k^{11} \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ D_k^{12} \end{bmatrix} \\ &= D_k^{22} - D_k^{21} \left[C_k + D_k^{11} - B_k^T A_k^{-1} B_k \right]^{-1} D_k^{12} \end{aligned}$$

结合 (166) 式，可得递推公式

$$J_{k+1} = D_k^{22} - D_k^{21} [D_k^{11} + J_k]^{-1} D_k^{12} \tag{170}$$

若考虑加性高斯噪声的系统 (73)、(74)，可知

$$-\log f(\boldsymbol{x}_{k+1} | \boldsymbol{x}_k) = c_1 + \frac{1}{2} [\boldsymbol{x}_{k+1} - \boldsymbol{g}(\boldsymbol{x}_k)]^{\mathrm{T}} Q_k^{-1} [\boldsymbol{x}_{k+1} - \boldsymbol{g}(\boldsymbol{x}_k)]$$

$$-\log f(\boldsymbol{y}_{k+1} | \boldsymbol{x}_{k+1}) = c_2 + \frac{1}{2} [\boldsymbol{y}_{k+1} - \boldsymbol{h}(\boldsymbol{x}_{k+1})]^{\mathrm{T}} R_{k+1}^{-1} [\boldsymbol{y}_{k+1} - \boldsymbol{h}(\boldsymbol{x}_{k+1})]$$

式中 c_1，c_2 为常数。由此可得 D_k^{11}，D_k^{12}，D_k^{21} 和 D_k^{22} 的表达式为

$$
\begin{aligned}
D_k^{11} &= E\{[\nabla_{x_k}\boldsymbol{g}^{\mathrm{T}}(\boldsymbol{x}_k)]Q_k^{-1}[\nabla_{x_k}\boldsymbol{g}^{\mathrm{T}}(\boldsymbol{x}_k)]^{\mathrm{T}}\} \\
D_k^{12} &= -E\{\nabla_{x_k}\boldsymbol{g}^{\mathrm{T}}(\boldsymbol{x}_k)\}Q_k^{-1} \\
D_k^{21} &= (D_k^{12})^{\mathrm{T}} \\
D_k^{22} &= Q_k^{-1} + E\{[\nabla_{x_{k+1}}\boldsymbol{h}^{\mathrm{T}}(\boldsymbol{x}_{k+1})]R_{k+1}^{-1}[\nabla_{x_{k+1}}\boldsymbol{h}^{\mathrm{T}}(\boldsymbol{x}_{k+1})]^{\mathrm{T}}\}
\end{aligned}
$$

递推公式 (170) 的初始信息矩阵 J_0 定义为

$$
J_0 = E\{-\Delta_{x_0}^{x_0}\log f(\boldsymbol{x}_0)\} \tag{171}
$$

上述结论对于线性系统当然也成立。

7. 仿真算例

7.1 常见估计性能评价指标

7.1.1 均方根误差

均方根误差 (Root Mean Squared Error, RMSE) 也称为标准误差，这是用得最多的一类估计算法评价指标。定义为

$$RMSE = \sqrt{\frac{1}{M}\sum_{k=1}^{M}\|\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k\|^2} \tag{172}$$

其中，M 是总的数据个数 (仿真或实验总时长)。

假设总共进行了 N 次蒙特卡洛仿真，记第 i 次蒙特卡洛仿真中系统在 k 时刻的状态真值为 $\boldsymbol{x}_k^{(i)}$，对应的状态估计为 $\hat{\boldsymbol{x}}_k^{(i)}$，那么 k 时刻状态估计的均方根误差定义为

$$RMSE_k = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{x}_k^{(i)} - \hat{\boldsymbol{x}}_k^{(i)}\|^2} \tag{173}$$

由此我们可以评估每个时刻的估计效果，并绘制 $RMSE_k \sim k$ 变化曲线，从而考察不同时间段的估计精度。

在式 (173) 基础上，我们还可以定义平均累计均方根误差 (ARMSE)，即

$$ARMSE = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\frac{1}{M}\sum_{k=1}^{M}\|\boldsymbol{x}_k^{(i)} - \hat{\boldsymbol{x}}_k^{(i)}\|^2} \tag{174}$$

RMSE 在数量级上直接反映出系统状态真实值与估计值之间的误差，取值范围为 $[0,+\infty)$，取值越小则表明估计误差越小。

7.1.2 平均绝对误差

平均绝对误差 (Mean Absolute Error, MAE) 是系统状态每一个时刻的估计值与真实值的偏差的绝对值的平均，定义为

$$MAE=\frac{1}{M}\sum_{k=1}^{M}\|\boldsymbol{x}_k-\hat{\boldsymbol{x}}_k\| \tag{175}$$

对于 N 次蒙特卡洛仿真，第 i 次实验 k 时刻的绝对误差为 $\|\boldsymbol{x}_k^{(i)} - \hat{\boldsymbol{x}}_k^{(i)}\|$，于是我们也可以定义每个时刻的平均绝对误差，即

$$MAE_k = \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{x}_k^{(i)} - \hat{\boldsymbol{x}}_k^{(i)}\| \tag{176}$$

同样地，我们可以绘制 $MAE_k \sim k$ 变化曲线，考察平均绝对误差随时间的变化情况。

MAE 取值范围为 $[0, +\infty)$，取值越小则表明估计算法越好。

7.1.3 平均绝对百分比误差

平均绝对百分比误差 (Mean Absolute Percentage Error, MAPE) 是在 MAE 的基础上改进来的，利用状态真实值幅度作为参考值来表示误差的相对大小，定义为

$$MAPE = 100\% \cdot \frac{1}{M} \sum_{k=1}^{M} \frac{\|\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k\|}{\|\boldsymbol{x}_k\|} \tag{177}$$

类似地，对于蒙特卡洛实验，我们也可以定义每个时刻的平均绝对百分比误差。容易发现，MAPE 限制了被估计量不能取值为 0，这是 MAPE 的局限性。对于明确知道非零的被估计量，MAPE 不失为一种有意义的指标。

7.1.4 决定系数

决定系数 (Determknation Coeffkckent) 也被称为判定系数或拟合优度，常缩写为 $\mathcal{R}^2$，它反映了被估计量与估计量之间关系密切程度的统计指标，定义如下：

$$\mathcal{R}^2 = 1 - \frac{1}{M}\sum_{k=1}^{M}\frac{\|\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k\|^2}{\|\boldsymbol{x}_k - \bar{\boldsymbol{x}}\|^2} \tag{178}$$

其中，$\bar{\boldsymbol{x}}$ 是被估计量的平均值。

$\mathcal{R}^2$ 常用来表示数据的拟合程度，当 $\mathcal{R}^2$ 越接近于 1 时，则认为拟合情况越好，估计值（预测值）越接近真实值。

7.2 典型非线性系统状态估计算例

Example 7.1 考虑非线性系统

$$x_{k+1} = 1 + \sin\frac{k\pi}{25} + \frac{1}{2}x_k + w_k$$

$$y_k = \begin{cases} \frac{1}{2}x_k^2 + v_k, k \le 30 \\ \frac{1}{2}x_k + v_k, k > 30 \end{cases}$$

其中，$w_k \sim \mathcal{N}(0, 10^{-5})$，$v_k \sim \Gamma(3, 0.5)$，$x_0 \sim \mathcal{N}(1, \frac{3}{4})$。

分别采用 EKF 和 UKF 进行状态估计仿真。UKF 参数选取为 $\alpha = 1$，$\beta = 2$，$\kappa = 0$。EKF 状态估计结果见图4，UKF 状态估计结果见图5, EKF 与 UKF 均方根误差 (RMSE，含义见 (172) 式) 比较见表4。结果表明，UKF 要优于 EKF。

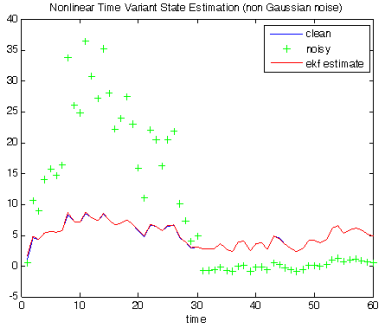

Figure 4: EKF 估计结果

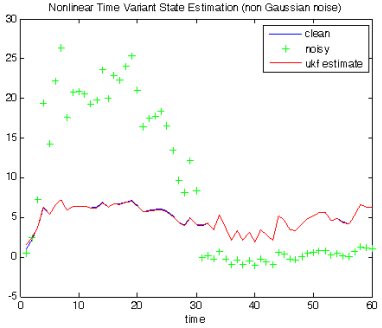

Figure 5: UKF 估计结果

Table 4: 均方根误差比较

算法	RMSE	备注
EKF	0.113	图4
UKF	0.043	图5

Example 7.2 考虑非线性系统

$$x_{k+1} = \frac{x_k}{2} + \frac{5x_k}{1+x_k^2} + 8\cos(0.4k) + w_k$$

$$y_k = \frac{x_k^2}{20} + v_k$$

其中，$x_0 \sim \mathcal{N}(0,5)$；$w_k \sim \mathcal{N}(0,10)$，$v_k \sim \mathcal{N}(0,1)$，二者均为白噪声，相互独立。*UKF* 状态估计仿真结果见图*6*。

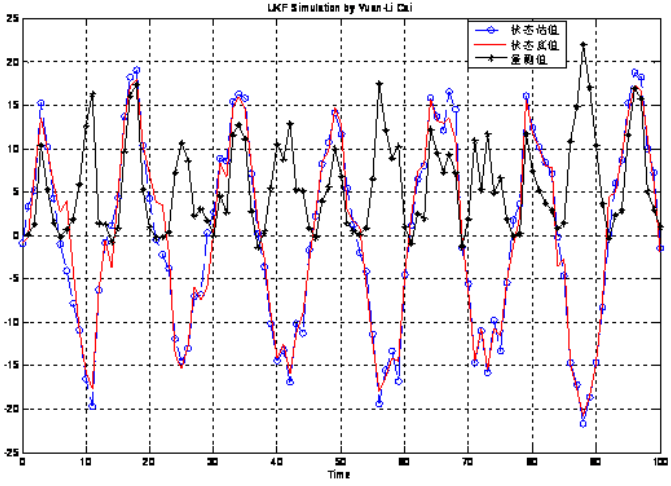

Figure 6: UKF 仿真 (RMSE=2.0)

UKF MATLAB 算法实现代码 (见教材)

仿真算例 MATLAB 代码 (见教材)

Example 7.3 考虑如下非线性系统

$$x_{k+1} = \frac{x_k}{2} + \frac{25x_k}{1+x_k^2} + 8\cos(1.2k) + w_k$$

$$y_k = \frac{x_k^2}{20} + v_k$$

其中，$x_0 \sim \mathcal{N}(0,5)$；$w_k \sim \mathcal{N}(0,10)$，$v_k \sim \mathcal{N}(0,1)$，二者均为白噪声，相互独立。

由于严重的非线性特性，使得 *EKF* 面临发散现象。这里采样粒子滤波方法进行状态估计仿真，选取 *1000* 个粒子，部分粒子滤波实验结果见图 *7* ∼ 图*9*。其中，图 *7*和图 *8* 分别是粒子滤波仿真实验中的两次状态估计对比图，可见粒子滤波能够较好地估计不同初始条件下的状态变化。图 *9*是粒子滤波过程中不同时刻粒子的分布图，可见该系统的中间状态无法用高斯分布来刻画。

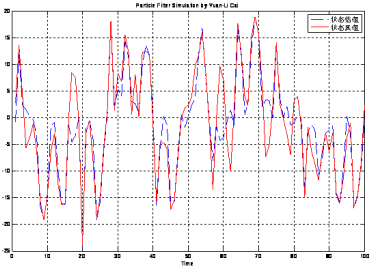

Figure 7: 例题7.3粒子滤波仿真结果 (a)

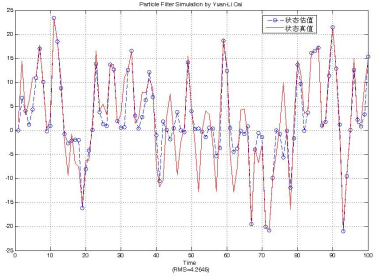

Figure 8: 例题7.3粒子滤波仿真结果 (b)

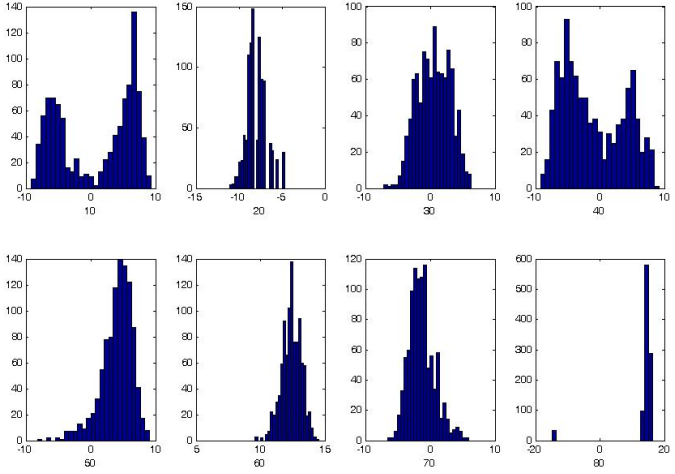

Figure 9: 例题7.3系统状态粒子分布图

粒子滤波 MATLAB 仿真程序 (见教材)

粒子滤波算法 MATLAB 实现代码 (见教材)

Example 7.4 考虑非线性系统

$$x_{k+1} = 1 + \sin\frac{k\pi}{25} + \frac{1}{2}x_k + w_k$$

$$y_k = \begin{cases} \frac{1}{2}x_k^2 + v_k, & k \le 30 \\ \frac{1}{2}x_k + v_k, & k > 30 \end{cases}$$

其中，$w_k \sim \mathcal{N}(0, 10^{-5})$，$v_k \sim \Gamma(3, 0.5)$，$x_0 \sim \mathcal{N}(1, \frac{3}{4})$。

显然，该问题中量测有非线性和突变特性，而且量测噪声是非高斯的，使得传统的非线性滤波方法效果都不太好。这里我们采样粒子数为 200 的粒子滤波和 SPPF 进行仿真，一组结果见图10和图11。简单观察可见，粒子滤波算法能够有效地完成这类复杂系统的状态估计，改进的粒子滤波方法（SPPF）估计精度更高。

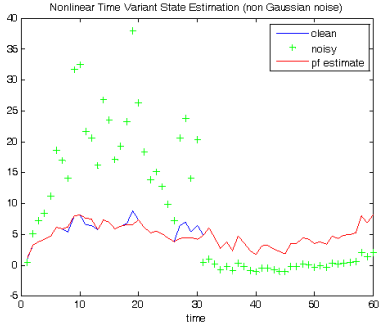

Figure 10: 例题7.4 PF 仿真结果

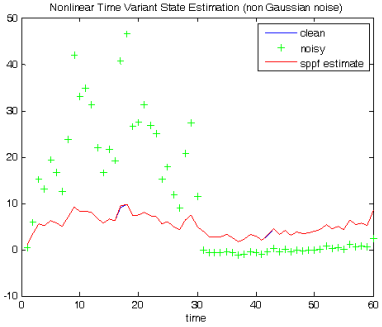

Figure 11: 例题7.4 SPPF 仿真结果

7.3 再入弹道目标状态估计仿真

7.3.1 坐标系与动力学方程

在雷达站东北天 (ENU) 坐标系下，再入飞行器动力学方程为

$$\ddot{x} = -\frac{1}{2\beta}\rho(r - R_e)V\dot{x} - \frac{\mu x}{r^3} \tag{179}$$

$$\ddot{y} = -\frac{1}{2\beta}\rho(r - R_e)V\dot{y} - \frac{\mu y}{r^3} \tag{180}$$

$$\ddot{z} = -\frac{1}{2\beta}\rho(r - R_e)V\dot{z} - \frac{\mu(z+R_e)}{r^3} \tag{181}$$

式中，$\beta = \frac{m}{C_d S}$ $(\mathrm{kg/m^2})$ 为弹道系数 (和飞行器外形尺寸等有关)。

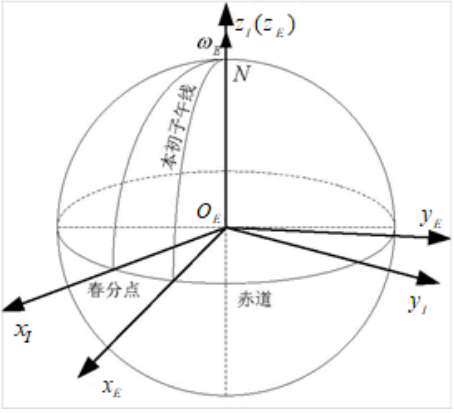

Figure 12: 地心惯性坐标系和地心固连坐标系

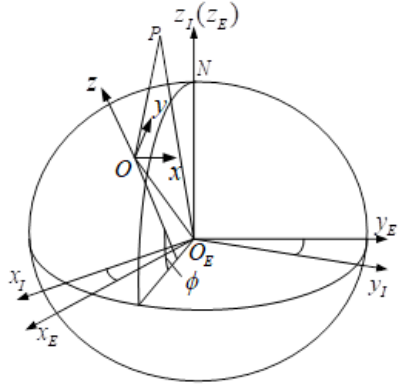

Figure 13: 东北天坐标系与地心固连坐标系之间的关系

7.3.2 状态方程与量测方程

弹道系数是个未知量，扩充为一个待估计状态。在雷达天线为原点的东北天坐标系中，再入目标运动的状态方程离散化形式为

$$\boldsymbol{x}_k = F\boldsymbol{x}_{k-1} + G\boldsymbol{\psi}(\boldsymbol{x}_{k-1}) + \boldsymbol{w}_{k-1} \tag{182}$$

其中，$\boldsymbol{x}_k = [x_k, \dot{x}_k, y_k, \dot{y}_k, z_k, \dot{z}_k, \beta_k]^{\mathrm{T}}$ 为再入目标 k 时刻的状态。此外

$$F = \begin{bmatrix} \phi & 0 & 0 & 0 \\ 0 & \phi & 0 & 0 \\ 0 & 0 & \phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \phi = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, G = \begin{bmatrix} \tau & 0 & 0 \\ 0 & \tau & 0 \\ 0 & 0 & \tau \\ 0 & 0 & 0 \end{bmatrix}, \tau = \begin{bmatrix} \frac{T^2}{2} \\ T \end{bmatrix}$$

$$\boldsymbol{\psi}(\boldsymbol{x}_{k-1}) = \begin{bmatrix} -\frac{\rho(h_{k-1})}{2\beta_{k-1}} V_{k-1}\dot{x}_{k-1} - \frac{\mu x_{k-1}}{r_{k-1}^3} \\ -\frac{\rho(h_{k-1})}{2\beta_{k-1}} V_{k-1}\dot{y}_{k-1} - \frac{\mu y_{k-1}}{r_{k-1}^3} \\ -\frac{\rho(h_{k-1})}{2\beta_{k-1}} V_{k-1}\dot{z}_{k-1} - \frac{\mu(z_{k-1}+R_e)}{r_{k-1}^3} \end{bmatrix}$$

假设过程噪声 $\boldsymbol{w}_k$ 是均值为零的白噪声，协方差矩阵 Q_k 近似为

$$Q_k = \begin{bmatrix} q_1\boldsymbol{\theta} & 0 & 0 & 0 \\ 0 & q_1\boldsymbol{\theta} & 0 & 0 \\ 0 & 0 & q_1\boldsymbol{\theta} & 0 \\ 0 & 0 & 0 & q_2 T \end{bmatrix}, \qquad \boldsymbol{\theta} = \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^2}{2} & T \end{bmatrix}$$

式中 T 为连续两个量测间隔时间（采样步长），q_1、q_2 为位置速度和弹道系数相关的噪声强度。

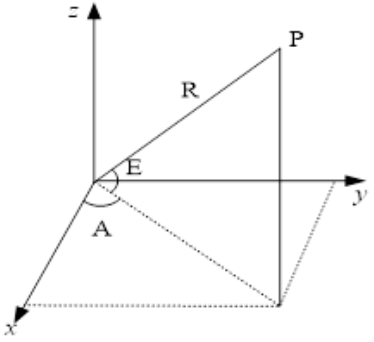

Figure 14: 再入飞行器测量关系 (ENU 坐标系)

如图14所示，雷达测量方程为

$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k \tag{183}$$

式中，$\boldsymbol{h}(\boldsymbol{x}_k) = [R_k, E_k, A_k]^{\mathrm{T}}$，且

$$\begin{aligned} R_k &= \sqrt{x_k + y_k + z_k} + v_R \\ E_k &= \arctan \frac{z_k}{\sqrt{x_k^2 + y_k^2}} + v_E \\ A_k &= \arctan \frac{y_k}{x_k} + v_A \end{aligned}$$

假设量测噪声 $\boldsymbol{v}_k = [v_R, v_E, v_A]^{\mathrm{T}}$ 是均值为零的白噪声，协方差矩阵为 $R_k = \mathrm{diag}[\sigma_R^2, \sigma_E^2, \sigma_A^2]$, σ_R， σ_E,σ_A 分别为距离、高低角和方位角的量测误差标准差。

7.3.3 仿真实验

设雷达量测间隔 $T = 0.1\text{s}$，跟踪目标 60s。进行 100 次蒙特卡罗仿真实验，UKF、CKF、ICKF 三种滤波算法的位置、速度和弹道系数估计的均方根误差如图15、图16和图17所示。

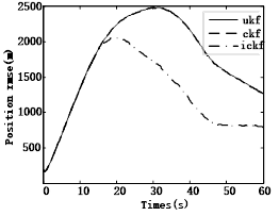

Figure 15: 位置均方根误差

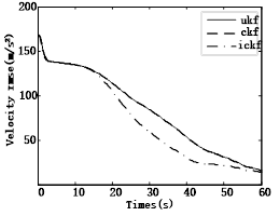

Figure 16: 速度均方根误差

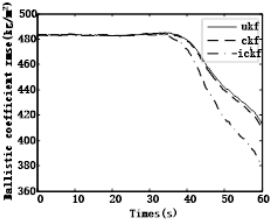

Figure 17: 弹道系数均方根误差

三种滤波算法的位置、速度和弹道系数平均累计均方根误差（ARMSE）和平均计算时间如表5所示。

Table 5: 三种滤波器的平均累计均方根误差及平均计算时间比较

滤波算法	E_f (m)	E_v (m/s)	E_β (kg/m^2)	t (s)
UKF	1698.57	86.20	397.95	1.02
CKF	1696.58	86.16	397.84	0.30
ICKF	1293.36	79.93	390.70	0.42

上述结果表明，对于本问题来说，ICKF 算法是一个有效的状态估计方法。

7. 本章小结

- ♠ 非线性系统的状态估计问题比较困难和麻烦，但具有重要的理论和实际意义，是动态估计理论的前沿领域和热门研究主题。
- ♠ 本章基于贝叶斯递推滤波框架，系统地介绍了经典和目前主流的非线性系统滤波方法，包括基于标称状态的线性化、EKF、UKF、CKF 以及粒子滤波等算法。

Questions?