Trajectory Filter

陈烁龙

2022 年 5 月 7 日

目录

插图

表格

摘要

SLAM 中估计的手段一般有两种：基于滤波的方式和基于非线性优化的方式。本次模拟了一个用于算法开发的位姿轨迹。同时，本文基于卡尔曼滤波的方式，给出了对一个位姿轨迹的估计滤波过程。

关键词：卡尔曼滤波，PCL，位姿模拟

1 数值模拟

一般，SLAM 后端优化都使用的是非线性优化，我们认为其估计的结果优于传统滤波的方式。所以本次我们使用的卡尔曼滤波不是为了进行位姿的估计，而是用于位姿预测，以加快特征的匹配[1]。

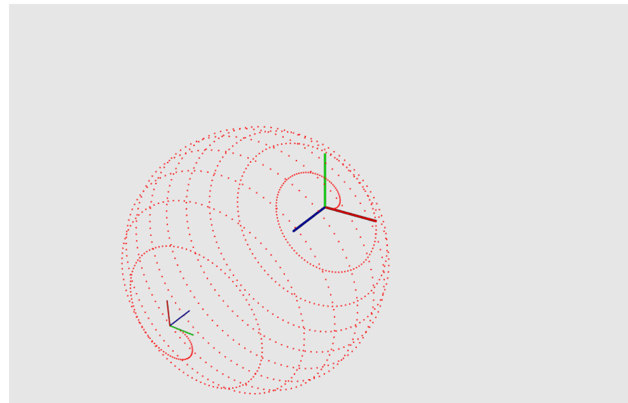
为了实验的进行，我们首先要模拟出一组位姿轨迹数据。我们知道，即使对位姿图进行了 BA 优化，但是仍然会出现累积误差[2]。基于此经验，我们会首先设计一条无误差的标准位姿轨迹[3]，此轨迹可用来后续处理结果的评估。同时对于每一帧，我们计算标准轨迹位姿变化量 (使用李代数的形式给出)：

$$\xi_{b_i b_{i-1}}^{update} = \xi_{b_i w}^{truth} (\xi_{b_{i-1} w}^{truth})^{-1}$$

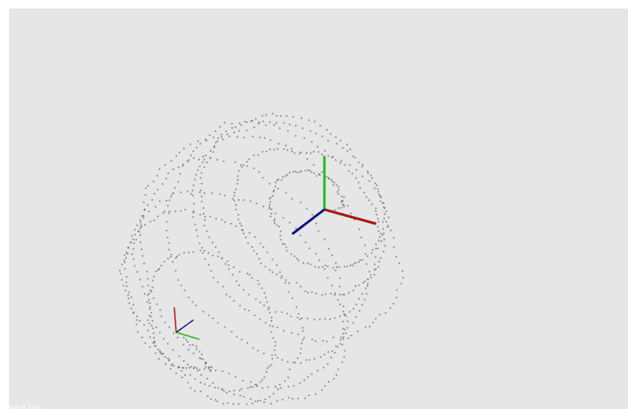
并在标准轨迹位姿变化量的基础上增加误差，得到附有误差的位姿变化量，而后在对观测值进行更新：

$$\xi_{b_i b_{i-1}}^{error} \leftarrow \begin{cases} R_{b_i b_{i-1}}^{error} = R_Z(N_R) R_Y(N_R) R_X(N_R) \\ t_{b_i b_{i-1}}^{error} = \begin{pmatrix} N_t & N_t & N_t \end{pmatrix}^T \end{cases}$$

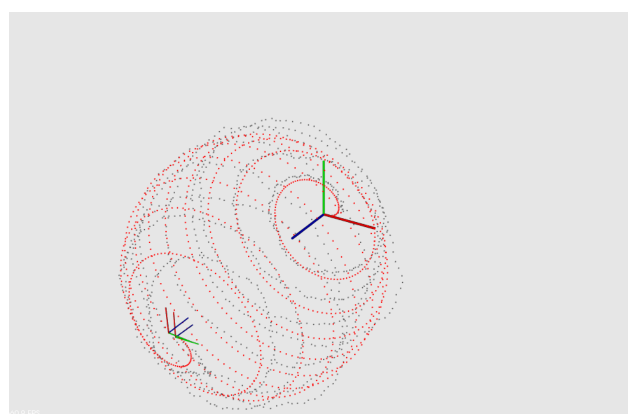
$$\rightarrow \xi_{b_i w}^{obs} = (\xi_{b_i b_{i-1}}^{error} \xi_{b_i b_{i-1}}^{update}) \xi_{b_{i-1} w}^{obs}$$



(a) 真实轨迹



(b) 观测轨迹



(c) 综合

图 1: 模拟效果

[1] 当预测了下一帧的位姿后，我们可以推算出当前地图点在下一帧影像上可能出现的位置。

[2] 这也是为什么有回环检测的原因。

[3] 至于如何设计的轨迹，后文有详细的描述。

其中 N_R、N_t 为服从在正太分布的 0 均值随机变量，我们事先假定了其标准差。

$$
\begin{cases}
N_R \sim N(0, \sigma_R = \frac{\pi}{1800}) \\
N_t \sim N(0, \sigma_t = 0.01)
\end{cases}
$$

图 1 为本次我们模拟出来的位姿轨迹图[4]。其中红色的位姿轨迹为实 (理想) 的位姿轨迹，而 h 灰色的则为我们引入累积误差的轨迹，为我们所 “观测到的” 位姿轨迹。可以看到，随着附带误差的新估计位姿变化量的到来，我们所观测的位姿轨迹逐渐偏离真实的位姿轨迹。这正是我们所需要的。

2 要求说明

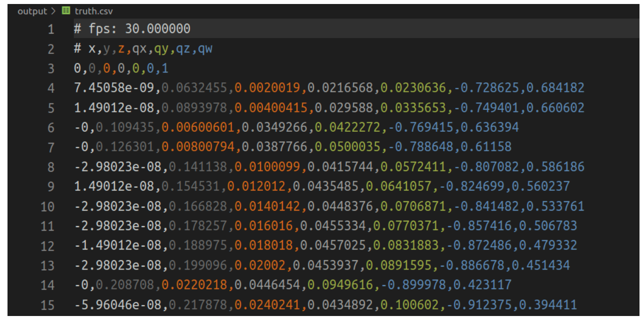
如下图 2 所示，这是我们要解算的数据文件。头两行是一些数据或格式的相关说明 (采样帧率，数据对应关系)。而后的每一行为一个位姿，其中前三维是位移量，后四维是表示旋转的单位四元数。注意每一条数据所给出的是由 w 系到当前 b_i 系的位姿变换 $\xi_{b_i w}$。

你需要做的就是，基于观测值文件，尽可能给出与真实轨迹一致的滤波或优化结果。可以通过真实轨迹数据来评估你的处理结果。例如，我们可以用绝对轨迹误差来描述：

$$
ATE_{all} = (\frac{1}{N} \sum_{i=0}^{N-1} \parallel \xi_{truth,i}^{-1} \xi_{esti,i} \parallel^2)^{\frac{1}{2}}
$$

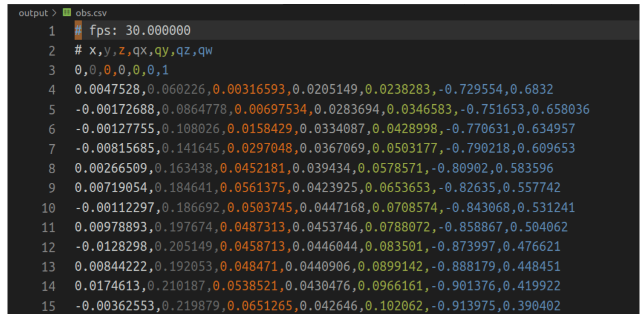
对于真实理想轨迹而言，其 ATE_{all} 为 0。

[4]此三维场景使用 PCL 绘制，其中坐标轴的对应顺序为 $rgb \to xyz$。



```
output > truth.csv
1  # fps: 30.000000
2  # x,y,z,qx,qy,qz,qw
3  0,0,0,0,0,0,1
4  7.45058e-09,0.0632455,0.0020019,0.0216568,0.0230636,-0.728625,0.684182
5  1.49012e-08,0.0893978,0.00400415,0.029588,0.0335653,-0.749401,0.660602
6  -0,0.109435,0.00600601,0.0349266,0.0422272,-0.769415,0.636394
7  -0,0.126301,0.00800794,0.0387766,0.0500035,-0.788648,0.61158
8  -2.98023e-08,0.141138,0.0100099,0.0415744,0.0572411,-0.807082,0.586186
9  1.49012e-08,0.154531,0.012012,0.0435485,0.0641057,-0.824699,0.560237
10 -2.98023e-08,0.166828,0.0140142,0.0448376,0.0706871,-0.841482,0.533761
11 -2.98023e-08,0.178257,0.016016,0.0455334,0.0770371,-0.857416,0.506783
12 -1.49012e-08,0.188975,0.018018,0.0457025,0.0831883,-0.872486,0.479332
13 -2.98023e-08,0.199096,0.02002,0.0453937,0.0891595,-0.886678,0.451434
14 -0,0.208708,0.0220218,0.0446454,0.0949616,-0.899978,0.423117
15 -5.96046e-08,0.217878,0.0240241,0.0434892,0.100602,-0.912375,0.394411
```

(a) 真实轨迹文件



```
output > obs.csv
1  # fps: 30.000000
2  # x,y,z,qx,qy,qz,qw
3  0,0,0,0,0,0,1
4  0.0047528,0.060226,0.00316593,0.0205149,0.0238283,-0.729554,0.6832
5  -0.00172688,0.0864778,0.00697534,0.0283694,0.0346583,-0.751653,0.658036
6  -0.00127755,0.108026,0.0158429,0.0334087,0.0428998,-0.770631,0.634957
7  -0.00815685,0.141645,0.0297048,0.0367069,0.0503177,-0.790218,0.609653
8  0.00266509,0.163438,0.0452181,0.039434,0.0578571,-0.80902,0.583596
9  0.00719054,0.184641,0.0561375,0.0423925,0.0653653,-0.82635,0.557742
10 -0.00112297,0.186692,0.0503745,0.0447168,0.0708574,-0.843068,0.531241
11 0.00978893,0.197674,0.0487313,0.0453746,0.0788072,-0.858867,0.504062
12 -0.0128298,0.205149,0.0458713,0.0446044,0.083501,-0.873997,0.476621
13 0.00844222,0.192053,0.048471,0.0440906,0.0899142,-0.888179,0.448451
14 0.0174613,0.210187,0.0538521,0.0430476,0.0966161,-0.901376,0.419922
15 -0.00362553,0.219879,0.0651265,0.042646,0.102062,-0.913975,0.390402
```

(b) 观测轨迹文件

图 2: 轨迹文件

3 模拟过程说明

3.1 轨迹生成

我们模拟的输入是帧率 f、总共的点数 n_p，目的是在一个球：

$$
\begin{cases}
(X - X_c^w)^2 + (Y - Y_c^w)^2 + (Z - Z_c^w)^2 = R_c^2 \\
P_c^w = \begin{pmatrix} X_c^w & Y_c^w & Z_c^w \end{pmatrix}^T = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T \\
R_c = 1
\end{cases}
$$

的表面生成一个环状的轨迹，其中 P_c^w 表示的是球的中心在世界坐标系下的投影坐标。为了让轨迹更加均匀，我们假定环绕球圈数是固定的，为 $n_c = 10$；且相邻点之间的 Z 坐标增量 ΔZ 是相同的。因此，我们有：

$$
\begin{cases}
\Delta\theta = \theta_i - \theta_{i-1} = 2\pi n_c / (n_p - 1) \\
\Delta Z = Z_i^w - Z_{i-1}^w = 2R_c / (n_p - 1) \\
i \in 0, ..., n_p - 1
\end{cases}
$$

其中 θ_i 表示原点到 P_i 在 $X - Y$ 面的投影点 $OP_{P_i}^{x-o-y}$ 的角度。如对于点 $P^w(\sqrt{2}/2, \sqrt{2}/2, 1)$，

则该值为 $\theta=\pi/4$。至此，我们可以计算每一点的坐标值：

$$\begin{cases}\theta_i=\theta_{i-1}+\Delta\theta\\ Z_i^w\ =Z_{i-1}^w+\Delta Z\\ R_i=\sqrt{R_c^2-(Z_i^w-Z_c^w)^2}\\ X_i^w=R_i\cos\theta_i\\ Y_i^w=R_i\sin\theta_i\end{cases}$$

$$\rightarrow P_i^w=\begin{pmatrix}X_i^w & Y_i^w & Z_i^w\end{pmatrix}^T\rightarrow t_{wb_i}$$

而这坐标值向量即为由当前坐标系 b_i 转到 w 坐标系的平移量 t_{wb_i}。

3.2 姿态生成

对于姿态，我们假定每一个位置处坐标系的 Z 轴指向球心，X 轴沿以 R_i 为半径的截面圆切线方向。我们现在要做的就是基于此要求，求解每个点处坐标系统的姿态 ξ_{b_iw}。我们记点 $P_o^w(0,0,0)$ 为世界坐标系原点，点 $P_c^w(0,0,1)$ 为球心，点 $P_t^w(0,0,2)$ 为球的上部顶点。所以，对于位置 P_i，我们有：

- 球心 P_c 的坐标对应关系：

$$\begin{cases}P_c^w=\begin{pmatrix}0 & 0 & 1\end{pmatrix}^T\leftrightarrow P_c^{b_i}=\begin{pmatrix}0 & 0 & 1\end{pmatrix}^T\end{cases}$$

- 世界坐标系原点 P_o 的坐标对应关系：

$$\begin{cases}\overrightarrow{P_iO}^w=P_o^w-P_i^w\\ \overrightarrow{P_iC}^w=P_c^w-P_i^w\\ \cos\theta_1=\overrightarrow{P_iO}^w\cdot\overrightarrow{P_iC}^w/(\mid\overrightarrow{P_iO}^w\mid\mid\overrightarrow{P_iC}^w\mid)\end{cases}$$

$$\rightarrow\begin{cases}P_o^w=\begin{pmatrix}0 & 0 & 0\end{pmatrix}^T\leftrightarrow P_o^{b_i}=\begin{pmatrix}0 & Y_o^{b_i} & Z_o^{b_i}\end{pmatrix}^T\\ Y_o^{b_i}=\mid\overrightarrow{P_iO}^w\mid\sin\theta_1\\ Z_o^{b_i}=\mid\overrightarrow{P_iO}^w\mid\cos\theta_1\end{cases}$$

- 球的上部顶点 P_t 的坐标对应关系：

$$\begin{cases}\overrightarrow{P_iT}^w=P_t^w-P_i^w\\ \overrightarrow{P_iC}^w=P_c^w-P_i^w\\ \cos\theta_2=\overrightarrow{P_iT}^w\cdot\overrightarrow{P_iC}^w/(\mid\overrightarrow{P_iT}^w\mid\mid\overrightarrow{P_iC}^w\mid)\end{cases}$$

$$\rightarrow\begin{cases}P_t^w=\begin{pmatrix}0 & 0 & 2\end{pmatrix}^T\leftrightarrow P_t^{b_i}=\begin{pmatrix}0 & Y_t^{b_i} & Z_t^{b_i}\end{pmatrix}^T\\ Y_t^{b_i}=-\mid\overrightarrow{P_iT}^w\mid\sin\theta_2\\ Z_t^{b_i}=\mid\overrightarrow{P_iT}^w\mid\cos\theta_2\end{cases}$$

至此，基于上述三个坐标点，我们有如下的坐标变换关系：

$$R_{wb_i}\begin{pmatrix}P_c^{b_i}\\ P_o^{b_i}\\ P_t^{b_i}\end{pmatrix}+\begin{pmatrix}t_{wb_i}\\ t_{wb_i}\\ t_{wb_i}\end{pmatrix}=\begin{pmatrix}P_c^w\\ P_o^w\\ P_t^w\end{pmatrix}$$

$$\rightarrow\begin{cases}\begin{pmatrix}P^w\\ 1\end{pmatrix}=T_{wb_i}\begin{pmatrix}P^{b_i}\\ 1\end{pmatrix}\\ T_{wb_i}=\begin{pmatrix}R_{wb_i} & t_{wb_i}\\ 0 & 1\end{pmatrix}\end{cases}$$

它描述的是 b_i 坐标系下的坐标 P^{b_i}，通过从 b_i 系到 w 系的位姿变换关系，得到了其在世界坐标系下的球心 P^w。

通过上式，我们可以通过非线性优化的方式 (也可以通过解析解的方式) 来求解旋转量 ξ_{b_iw}。非线性优化更为方便，我们使用李代数进行优化。我们定义误差项：

$$e_i=R_{wb_i}P^{b_i}+t_{wb_i}-P^w$$

对旋转量求导：

$$J=\partial e_i/\partial\delta\varphi=-(R_{wb_i}P^{b_i})^\wedge$$

通过高斯-牛顿法求解：

$$J^TJ\Delta x=-J^Te_i$$

Δx 为每次迭代对位姿的更新量。当更新量达到阈值要求时，我们停止迭代。

4 Kalman 滤波公式

此部分只作参考，可以有其他方法滤波或优化处理。

4.1 运动方程

首先我们得取定我们所研究系统的运动方程。不失一般性，我们假设我们到估计的位姿是均匀变化的，同时考虑模型的系统误差，我们有：

$$\ddot{\xi}(t)=\begin{pmatrix}\ddot{\xi}_R(t)\\ \ddot{\xi}_t(t)\end{pmatrix}=\begin{pmatrix}e_R(t)\\ e_t(t)\end{pmatrix}$$

其中 $\ddot{\xi}(t)$ 为 SE3 对应的李代数对时间的二次导数，是一个有六个元素的列向量[5]。$e(t)$ 为白噪声，满足：

$$\begin{cases}E(e(t))=0\\ Cov(e(t),e(\tau))=D_e(t)\delta(t-\tau)\\ D_e(t)=\begin{pmatrix}D_{e_R}(t) & \mathbf{0_{3\times3}}\\ \mathbf{0_{3\times3}} & D_{e_t}(t)\end{pmatrix}\end{cases}$$

其中 $\delta(t-\tau)$ 为狄拉克-δ 函数。我们设我们的状态参量为：

$$X(t)=\begin{pmatrix}X_1(t) & X_2(t)\end{pmatrix}^T=\begin{pmatrix}\xi(t) & \dot{\xi}(t)\end{pmatrix}^T$$

则有：

$$\begin{pmatrix}\dot{\xi}(t)\\ \ddot{\xi}(t)\end{pmatrix}=\begin{pmatrix}\mathbf{0_{6\times6}} & \mathbf{1_{6\times6}}\\ \mathbf{0_{6\times6}} & \mathbf{0_{6\times6}}\end{pmatrix}\begin{pmatrix}\xi(t)\\ \dot{\xi}(t)\end{pmatrix}+\begin{pmatrix}\mathbf{0_{6\times6}}\\ \mathbf{1_{6\times6}}\end{pmatrix}e(t)$$

我们将其记为：

$$\dot{X(t)}=A(t)X(t)+C(t)e(t)$$

通过解这个微分方程，并将其离散化，我们可以得到：

$$X(t_k)=\Phi(t_k,t_{k-1})X(t_{k-1})+w(t_{k-1})$$

[5]其中三维与旋转有关，三维与平移有关。

其中 $\Phi(t_k,\tau)$ 为状态转移矩阵（我们记 $f(\tau)=t_k-\tau$）：

$$\Phi(t_k,\tau)=\begin{pmatrix}\mathbf{1_{6\times6}} & f(\tau)\mathbf{1_{6\times6}}\\ \mathbf{0_{6\times6}} & \mathbf{1_{6\times6}}\end{pmatrix}$$

而 $w(t_{k-1})$ 为噪声项（系统噪声）：

$$w(t_{k-1})=\int_{t_{k-1}}^{t_k}\Phi(t_k,\tau)C(\tau)D_e(\tau)d\tau$$

其方差为：

$$D_w(t_{k-1})=\int_{t_{k-1}}^{t_k}\Phi(t_k,\tau)C(\tau)D_e(\tau)C(\tau)^T\Phi(t_k,\tau)^Td\tau$$

易知：

$$\Phi(t_k,\tau)C(\tau)=\begin{pmatrix}f(\tau)\mathbf{1_{3\times3}} & \mathbf{0_{3\times3}}\\ \mathbf{0_{3\times3}} & f(\tau)\mathbf{1_{3\times3}}\\ \mathbf{1_{3\times3}} & \mathbf{0_{3\times3}}\\ \mathbf{0_{3\times3}} & \mathbf{1_{3\times3}}\end{pmatrix}$$

所以被积分的矩阵为：

$$\begin{pmatrix}f^2(\tau)D_{e_R} & \mathbf{0_{3\times3}} & f(\tau)D_{e_R} & \mathbf{0_{3\times3}}\\ \mathbf{0_{3\times3}} & f^2(\tau)D_{e_t} & \mathbf{0_{3\times3}} & f(\tau)D_{e_t}\\ f(\tau)D_{e_R} & \mathbf{0_{3\times3}} & D_{e_R} & \mathbf{0_{3\times3}}\\ \mathbf{0_{3\times3}} & f(\tau)D_{e_t} & \mathbf{0_{3\times3}} & D_{e_t}\end{pmatrix}$$

积分求解可得方差矩阵为：

$$D_w(t_{k-1})=\begin{pmatrix}\frac{(\Delta t)^3D_{e_R}}{3} & \mathbf{0_{3\times3}} & \frac{(\Delta t)^2D_{e_R}}{2} & \mathbf{0_{3\times3}}\\ \mathbf{0_{3\times3}} & \frac{(\Delta t)^3D_{e_t}}{3} & \mathbf{0_{3\times3}} & \frac{(\Delta t)^2D_{e_t}}{2}\\ \frac{(\Delta t)^2D_{e_R}}{2} & \mathbf{0_{3\times3}} & \Delta tD_{e_R} & \mathbf{0_{3\times3}}\\ \mathbf{0_{3\times3}} & \frac{(\Delta t)^2D_{e_t}}{2} & \mathbf{0_{3\times3}} & \Delta tD_{e_t}\end{pmatrix}$$

其中 $\Delta t=t_k-t_{k-1}$。

4.2 观测方程

相较于运动方程。我们的观测方程就变得比较简单了，因为我们所观测的参量中的一部分正是我们所估计的参数。所以，我们有：

$$Z(t_k)=\begin{pmatrix}\mathbf{0_{6\times6}} & \mathbf{1_{6\times6}}\end{pmatrix}\begin{pmatrix}X_1(t_k)\\ X_2(t_k)\end{pmatrix}+\Delta(t_k)$$

我们将其记为：

$$Z(t_k) = H(t_k)X(t_k) + \Delta(t_k)$$

其中，$\Delta(t)$ 为白噪声，满足：

$$Cov(\Delta(t), \Delta(\tau)) = D_\Delta(t)\delta(t-\tau)$$

4.3 Kalman 滤波

基于上文，我们有如下的公式：

4.3.1 一步预测

$$\begin{cases} \hat{X}(t_k, t_{k-1}) = \Phi(t_k, t_{k-1})\hat{X}(t_{k-1}) \\ D_{\hat{X}}(t_k, t_{k-1}) = \Phi(t_k, t_{k-1})D_{\hat{X}}(t_{k-1})\Phi(t_k, t_{k-1})^T \\ \qquad\qquad + D_w(t_{k-1}) \end{cases}$$

我们基于上一时刻 t_{k-1} 的状态量 $\hat{X}(t_{k-1})$，通过 t_{k-1} 到 t_k 的状态转移矩阵 $\Phi(t_k, t_{k-1})$，预测当前时刻 t_k 的状态量 $\hat{X}(t_k, t_{k-1})$，并对给出当前预测状态的方差矩阵 $D_{\hat{X}}(t_k, t_{k-1})$。

4.3.2 二步更新

$$\begin{cases} K(t_k) = D_{\hat{X}}(t_k, t_{k-1})H(t_k)^T \\ \qquad (H(t_k)D_{\hat{X}}(t_k, t_{k-1})H(t_k)^T + D_\Delta(t_k))^{-1} \\ \hat{X}(t_k) = \hat{X}(t_k, t_{k-1}) \\ \qquad + K(t_k)(Z(t_k) - H(t_k)\hat{X}(t_{k-1})) \\ D_{\hat{X}}(t_k) = (I - H(t_k)K(t_k))D_{\hat{X}}(t_k, t_{k-1}) \end{cases}$$

我们基于当前预测状态的方差矩阵 $D_{\hat{X}}(t_k, t_{k-1})$ 和观测矩阵 $H(t_k)$、观测方差矩阵 $D_\Delta(t_k)$，计算卡尔曼增益 $K(t_k)$，而后更新 t_k 时刻的状态量 $\hat{X}(t_k)$ 和方差矩阵 $D_{\hat{X}}(t_k)$。