第二讲-习题

姜帆

2019 年 6 月 21 日

目录

1 基础作业

1.1 Allen 方差标定

1.1.1 Allen 方差标定原理简述

目前，常用的随机误差识别方法包括功率谱密度 (PSD)、自相关函数估计、AIlan 方差估计等方法。A11an 方差分析法最初是 1966 年由美国国家标准局的 David Allan 为了研究原子钟的振荡器的稳定性而提出的，这种方法因其可以克服标准差对包含调频闪变噪声时出现的发散而得到广泛的应用。1980 年，Allan 方差被引入到陀螺仪的随机误差识别中，之后主要在中、低精度激光和光纤陀螺信号性能分析中使用。由于该方法的实用性强 1998 年 A1lan 方差被 IEEE 协会选为分析光纤陀螺随机误差的推荐方法。在 2003 年 Allan 方差第一次被应用到 MEMS 陀螺仪随机误差的分析中，并取得了预期的识别效果。由于陀螺等惯性传感器本身也具有振荡器的特征，因此该方法随后被广泛应用于各种惯性传感器的随机误差辨识中。

A11an 方差是一种基于时域的分析方法，它的特点是不仅能够确定产生数据噪声的基本随机过程的特性，而且能够识别给定噪声项的来源。它能非常容易地对误差源以及对整个噪声特性的影响程度进行细致的表征和识别，计算方便，易于分离。

A11an 方差是一种从时域上对信号频域稳定性进行分析的通用方法，也就是将随机误差作为时间序列来处理，描述其均方误差的方法。设一定的采样时间间隔对数据集进行采样，把所获数据进行分组；对每个子集求平均值；对每个不同平均时间计算 A11an 方差；作出 A11an 标准差随平均时间变化的双对数曲线。

噪声类型	参数	Allan 标准差	单位
量化噪声	Q	$\sigma_Q = \sqrt{3}Q/\tau$	μ rad
角度随机游走	N	$\sigma_N = N/\sqrt{\tau}$	$^\circ/\sqrt{h}$
零偏不稳定性	B	$\sigma_B = B/0.6648$	$^\circ/h$
角速率随机游走	K	$\sigma_K = K\sqrt{\tau/3}$	$(^\circ/h)/\sqrt{h}$
速度斜坡	R	$\sigma_R = R\tau/\sqrt{2}$	$(^\circ/h)/h$

图 1: Allan 方差与常见噪声对应关系

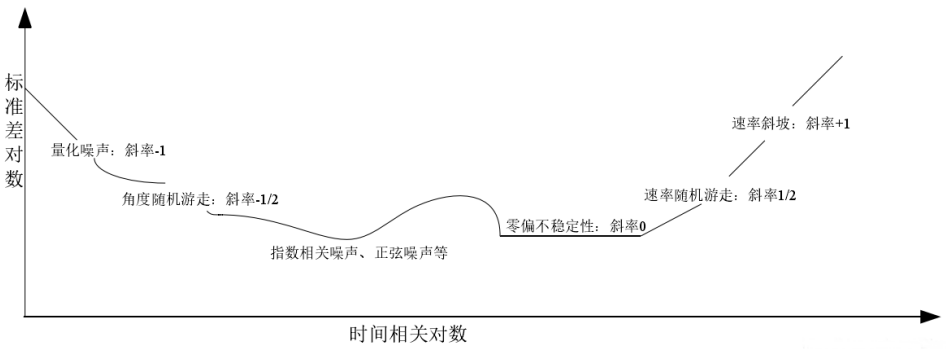

图 2: Allan 曲线示意图

1.1.2 Allen 方差标定实验

本文实验使用 imu_utils 工具生成 allan 曲线。

实验流程为:

(1) 进入 ros 工作空间, 配置 vio_data_simulation 功能包, 并运行生成 imu.bag 文件。

rosrun vio_data_simulation vio_data_simulation_node

(2) 进入 ros 工作空间,配置 imu_utils 功能包。回放 imu.bag 文件。同时运行 imu_utils。

roslaunch imu_utils simulation.launch

rosbag play -r 200 ../src/vio_data_simulation/data/imu2.bag

(3) 画出 Allan 曲线

下面为曲线绘制脚本

```
clear
close all

dt = dlmread('../data/data_simulation_imu2_gyr_t.txt');
data_x = dlmread('../data/data_simulation_imu2_gyr_x.txt');
data_y= dlmread('../data/data_simulation_imu2_gyr_y.txt');
data_z = dlmread('../data/data_simulation_imu2_gyr_z.txt');
data_drawavg=(data_x +data_y +data_z)/3/3600*(2.*3.141593653)
    /360;
data_draw=[data_x data_y data_z]/3600*(2.*3.141593653)/360 ;
data_sim_x= dlmread('../data/data_simulation_imu_sim_gyr_x.
    txt');
data_sim_y= dlmread('../data/data_simulation_imu_sim_gyr_y.
    txt');
data_sim_z= dlmread('../data/data_simulation_imu_sim_gyr_z.
    txt');
data_sim_draw=[data_sim_x data_sim_y data_sim_z
    ]/3600*(2*3.141593653)/360 ;
figure(1)
loglog(dt, data_draw(:,1) , 'r+');
hold on;
loglog(dt, data_draw(:,2) , 'b+');
hold on;
loglog(dt, data_draw(:,3) , 'k+');
hold on
loglog(dt, data_drawavg , '-');
legend('w_x','w_y','w_z','w_{avg}')
mix_where=find(dt==1);
num=dt(mix_where);
xlabel('time:sec');
ylabel('Sigma:deg/h');
```

```
27      grid on;
28      hold on;
29      %loglog(dt, data_sim_draw , 'r-');
30
31      dt = dlmread('../data/data_simulation_imu_acc_t.txt');
32      data_x = dlmread('../data/data_simulation_imu2_acc_x.txt');
33      data_y = dlmread('../data/data_simulation_imu2_acc_y.txt');
34      data_z = dlmread('../data/data_simulation_imu2_acc_z.txt');
35      data_draw=[data_x data_y data_z] ;
36      data_draw_avg=(data_x +data_y +data_z)/3 ;
37      data_sim_x= dlmread('../data/data_simulation_imu2_sim_acc_x.
            txt');
38      data_sim_y= dlmread('../data/data_simulation_imu2_sim_acc_y.
            txt');
39      data_sim_z= dlmread('../data/data_simulation_imu2_sim_acc_z.
            txt');
40      data_sim_draw=[data_sim_x data_sim_y data_sim_z] ;
41      figure(2)
42      loglog(dt, data_draw(:,1) , 'r+');
43      hold on
44      loglog(dt, data_draw(:,2) , 'b+');
45      hold on
46      loglog(dt, data_draw(:,3) , 'k+');
47      hold on
48      loglog(dt, data_draw_avg, '-');
49      legend('acc_x','acc_y','acc_z','acc_{avg}')
50      ylabel('Sigma:m/s2');
51      xlabel('time:sec');
52      hold on
53      % loglog(dt, data_sim_draw , 'b-');
54      grid on;
```

1. 第一组

连续时间的加速度的高斯白噪声方差设定为 0.019, 连续时间的陀螺仪的高斯白噪声为 0.015。连续时间的加速度 bias 的随机游走噪声设定为 5e−4 , 连续时间的陀螺仪的 bias 随机游走噪声设定为 5e−5。

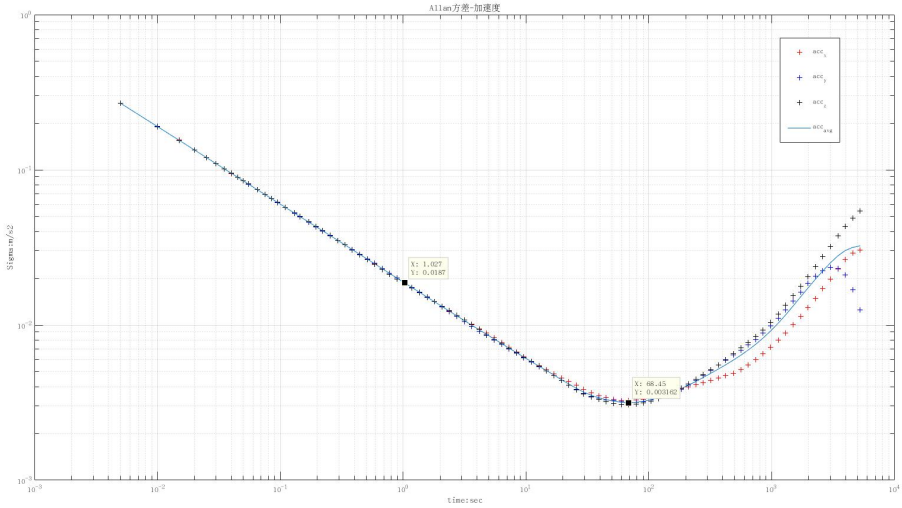

图 3: 连续时间的 Allan 方差-加速度

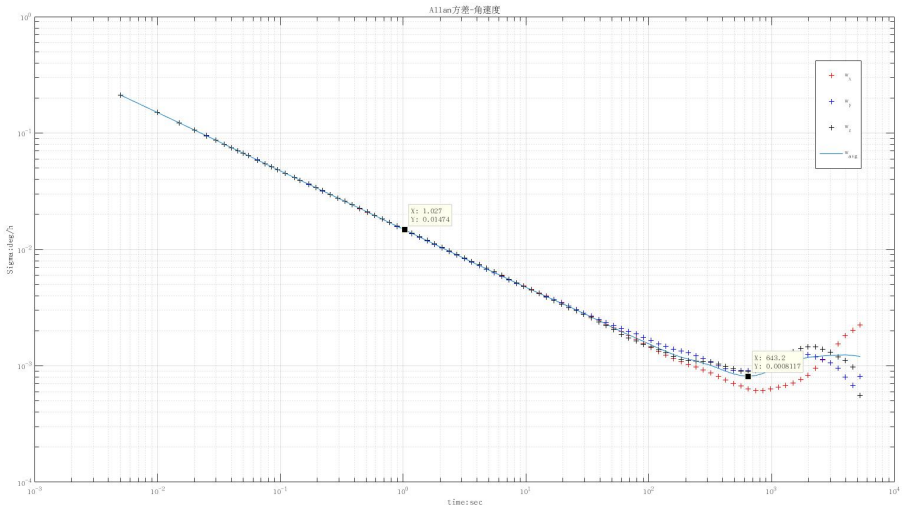

图 4: 连续时间的 Allan 方差-角速度

如图，绘制了 IMU 连续时间的加速度和角速度误差 Allan 方差曲线。从图中可以看出连续时间的加速度三个轴的平均值白噪声标定结果为 0.0187，连续时间的加速度三个轴的平均值 bias 随机游走噪声为 3.1e-3；连续时间的角速度三个轴的平均值白噪声标定结果为 0.147，连续时间的角速度三个轴的平均值 bias 噪声为 8.1e-4。

2. 第二组

连续时间的加速度的高斯白噪声设定为 0.035, 连续时间的陀螺仪的高斯白噪声为 0.025。连续时间的加速度 bias 的随机游走噪声设定为 2e−3 , 连续时间的陀螺仪的 bias 随机游走噪声设定为 2e−4。

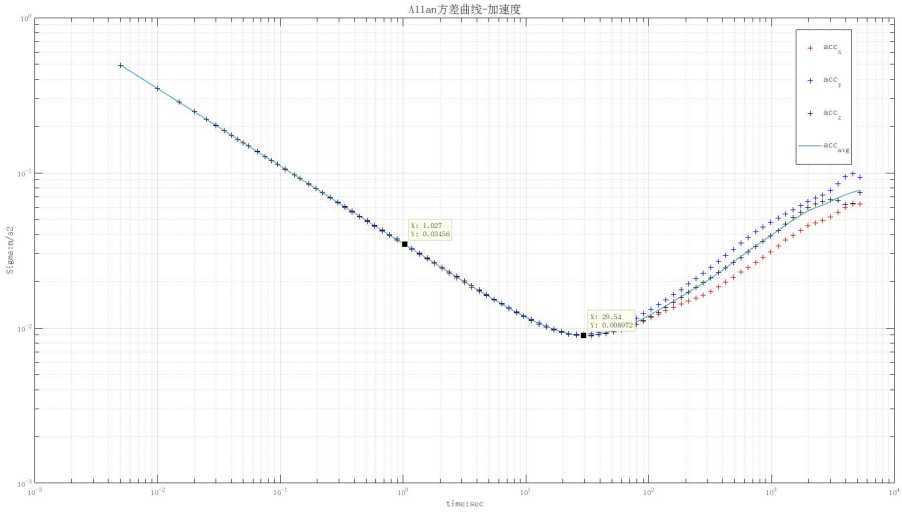

图 5: 连续时间 Allan 方差-加速度

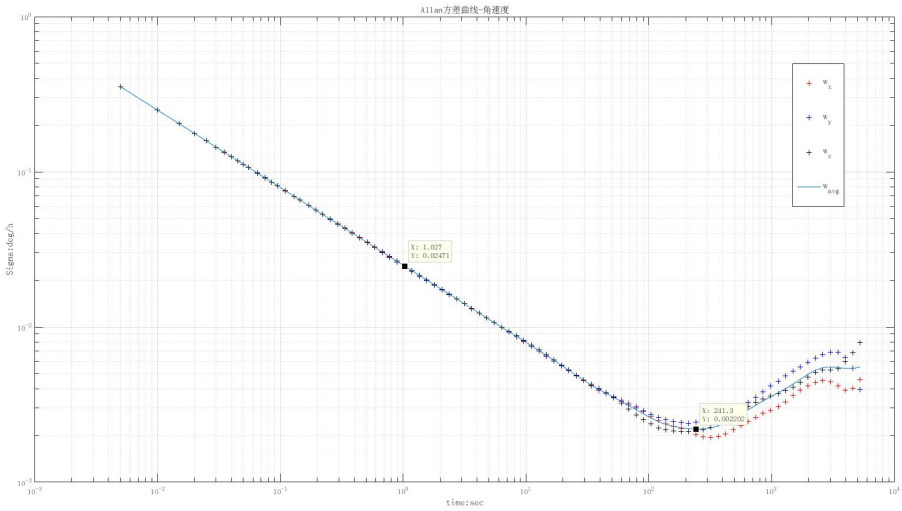

图 6: 连续时间的 Allan 方差-角速度

如图，绘制了 IMU 连续时间的加速度和角速度误差 Allan 方差曲线。从图中可以看出连续时间的加速度三个轴的平均值白噪声标定结果为 0.0345，连续时间的加速度三个轴的平均值 bias 随机游走噪声为 8.9e-3；连续时间的角速度三个轴的平均值白噪声标定结果为 0.247，连续时间的角速度三个轴的平均值 bias 噪声为 2.2e-3。

根据两次实验结果，可以发现 Allan 曲线标定的加速度高斯白噪声，角速度高斯白噪声与预设的基本吻合，但是加速度角速度 bias 随机游走标定结果与预设差一个数量级。

1.2 中值积分

程序如下:

```
1     double dt = param_.imu_timestep;
2     Eigen::Vector3d Pwb = init_twb_;              // position :
          from imu measurements
3     Eigen::Quaterniond Qwb(init_Rwb_);            // quaterniond:
          from imu measurements
4     Eigen::Vector3d Vw = init_velocity_;          // velocity :
          from imu measurements
5     Eigen::Vector3d gw(0,0,-9.81);    // ENU frame
6     Eigen::Vector3d temp_a;
7     Eigen::Vector3d theta;
8     for (int i = 1; i < imudata.size(); ++i) {
9
10        /// 中值积分
11        MotionData imupose = imudata[i];
12        MotionData imuposenext = imudata[i-1];
13        Eigen::Vector3d imu_gyro_mid = (imupose.imu_gyro +
              imuposenext.imu_gyro) / 2.0;        //w=1/2(wk-wk_biase
              +wk+1-wk+1_bias)
14        Eigen::Quaterniond dq;
15        Eigen::Vector3d dtheta_half = imu_gyro_mid * dt /2.0;
16        dq.w() = 1;
17        dq.x() = dtheta_half.x();
18        dq.y() = dtheta_half.y();
19        dq.z() = dtheta_half.z();
20        Eigen::Quaterniond Qwbnext= Qwb * dq;              // qk
              +1=qk*deltaq
21        Qwbnext=Qwbnext.normalized();
22        Eigen::Vector3d acc_w = ((Qwb*(imupose.imu_acc)+gw) + (
              Qwbnext*(imuposenext.imu_acc)+gw)) /2.0;    //aw = ((
              Rkwb * ( acck_body - acck_bias ) + gw)+(Rk+1wb * (
              acck+!_body - acck+1_bias ) + gw))/2
23        Qwb = Qwbnext;
24        Vw = Vw + acc_w * dt;
25        Pwb = Pwb + Vw * dt + dt * dt * acc_w /2.0;
26
27
28    }
```

下面为欧拉积分与中值积分得到的轨迹图:

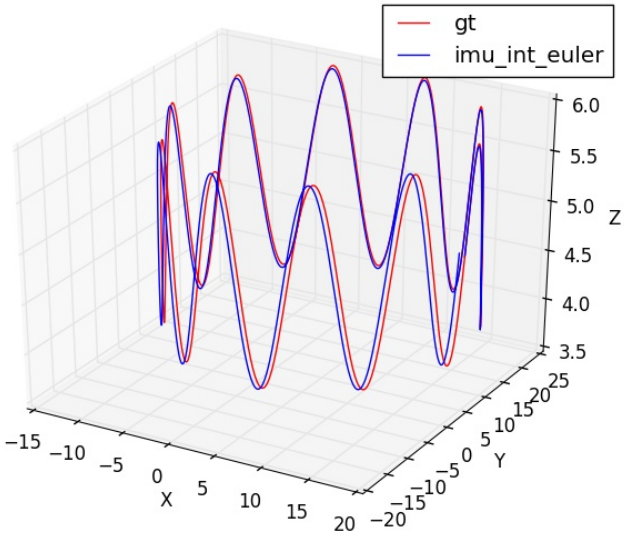

图 7: 欧拉积分结果

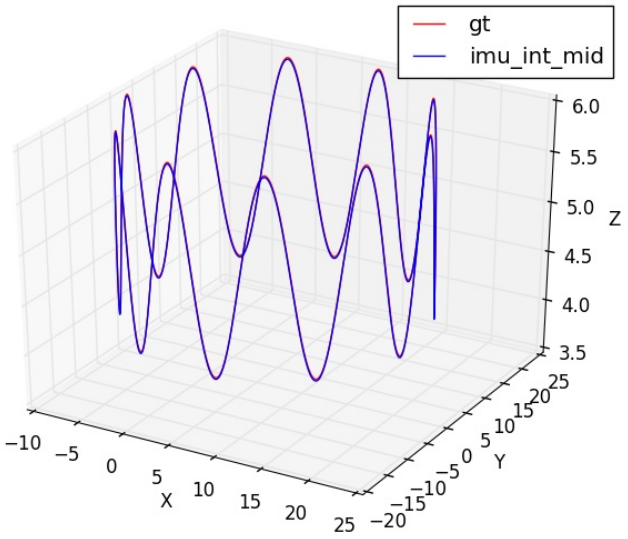

图 8: 中值积分结果

定性来看，由欧拉积分结果与中值积分结果图对比，我们可以看出 IMU 中值积分的结果比欧拉积分结果更加准确，更加接近真值。

定量来看，通过计算得到欧拉积分结果与真值的均方根误差为 0.00481，中值积分结果与真值的均方根误差为 0.00070，因此 IMU 中值积分结果要好于欧拉积分的结果

2 提升作业

利用 B 样条曲线可以对已有轨迹仿真 IMU 数据。

2.1 B 样条曲线简介

贝塞尔曲线于 1962，由法国工程师皮埃尔 · 贝塞尔（Pierre Bézier）所广泛发表，他运用贝塞尔曲线来为汽车的主体进行设计。贝塞尔曲线最初由 Paul de Casteljau 于 1959 年运用 de Casteljau 演算法开发，以稳定数值的方法求出贝塞尔曲线。但是贝塞尔曲线存在一些问题：缺乏局部修改性；幂次太高难以修改；n 较大时，特征多边形的边数较多，对曲线的控制减弱。

1972 年 Riesenfeld 提出了 B 样条曲线，用 B 样条基函数代替了贝塞尔曲线中的 Bernstein 基函数。

设有控制顶点 $p_0, p_1, ..., p_n$，则 k 阶 $(k-1)$B 样条曲线的数学表达式为：

$$p(t) = \sum_{i=1}^{n} p_i B_{i,k}(t) \tag{1}$$

其中 $B_{i,k}(t)$ 为 B 样条曲线的基函数。可以通过 De Boor - Cox 递归公式计算。

B 样条基函数是一个称为节点矢量的非递减的参数 t 的序列所决定的 k 阶分段多项式，也即为 k 阶 $(k-1)$ 此多项式样条。

2.2 B 样条曲线生成 IMU 仿真数据

式 (1) 可重组为其累积形式:

$$p(t) = p_0 \tilde{B_0}(t) + \sum_{i=1}^{n} (p_i - p_{i-1}) \tilde{B_{i,k}}(t) \tag{2}$$

其中 $\tilde{B_{i,k}}(t) = \sum_{j=i}^{n} B_{j,k}(t)$ 为累积函数。　　我们需要描述的是控制点位姿 (旋转，平移矩阵)，可以拓展上式为：

$$T_{w,s}(t) = exp(\tilde{B_{0,k}}(t)) log(T_{w,0}) \prod_{i=1}^{n} exp(\tilde{B_{i,k}}(t) \Omega_i) \tag{3}$$

其中 $\Omega_i = log(T_{w,s}^{-1} T_{w,i})$，$T_{w,s}(t)$ 是时间 t 处 B 样条的值，$T_{w,i}(t)$ 是控制点 i 的位姿。

此论文采用三次 B 样条曲线，即 $k = 4$。分段三次 B 样条曲线由相邻四个顶点　定义。基函数为：

$$\begin{aligned} F_{0,3}(t) &= \frac{1}{6}(1 - t^3)^3 \\ F_{1,3}(t) &= \frac{1}{6}(3t^3 - 6t^2 + 4) \\ F_{2,3}(t) &= \frac{1}{6}(-3t^3 + 3t^3 + 3t + 1) \\ F_{3,3}(t) &= \frac{1}{6}(t^3) \end{aligned} \tag{4}$$

最后的形式为

$$p(t) = p_0(F_{0,3}(t)) + p_0(F_{0,3}(t)) + p_1(F_{1,3}(t)) + p_2(F_{2,3}(t)) + p_3(F_{3,3}(t)) \tag{5}$$

这里定义 $u = \frac{t-t_i}{\delta t}$, 其中 t_i 为控制点。因为把控制点编号换为 u, 则有:

$$T_{w,s}(t) = T_{w,i-1} \prod_{j=1}^{3} exp(\tilde{B}(u)_j \Omega_i + j) \tag{6}$$

计算 IMU 加速度，角速度仿真公式为:

$$\begin{aligned} Gyro(u) &= R_{w,s}^T(u)\dot{R}_{w,s}(u) + bias \\ Accel(u) &= R_{w,s}^T(u)(\ddot{s}_{w,s}(u) + g_w) + bias \end{aligned} \tag{7}$$

其中 $\dot{R}_{w,s}$ 和 $\ddot{s}_{w,s}$ 分别为 $\dot{T}_{w,s}$ 和 $\ddot{T}_{w,s}$ 的旋转部分与平移部分。