2020年“最优估计理论”课程综合作业之二

**车载组合导航滤波算法设计**

姓名： 魏君陶

学号： SY1903120

邮箱： 1017456739@qq.com



签名：

2020年 5月 4日

**一、观测序列生成**

问题1：在给定位置基础上，东向、北向均叠加8米（1倍σ）随机定位误差，生成观测序列；

首先读取数据文件中的观测数据：东向位置(米)、北向位置(米)、GPS航向(度)、GPS速度(米/秒)、陀螺角速率(度/秒)、里程计脉冲（一秒内脉冲数），其中航向和陀螺角速率需要转换为弧度制单位来表示。将每列数据分别存储在[posOfEast, posOfNorth, GPSHeading, GPSSpeed, GyroAngularRate, Odemeter]中，每个向量大小为1078×1.按照题目要求对东向、北向均叠加8米随机定位误差以便获得实际观测数值，观测数据保持不变，最终获得的动向北向观测序列如下图1.1：

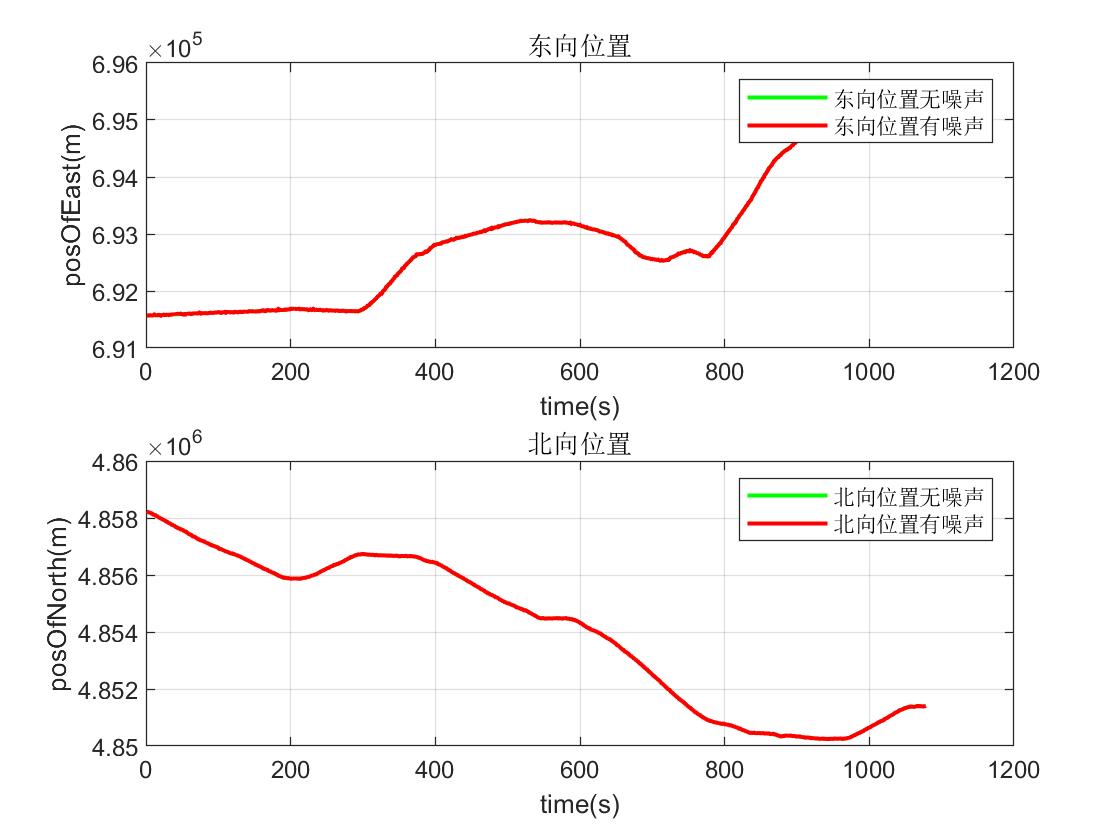


图1.1 东向、北向位置原始数据与加入噪声数据

该问题的代码如下：

%读取数据

X = load('z+n.txt');

PI=3.1415926;

state=0;

ang\_swap=8\*PI/180;

posOfEast = X(:, 1);

posOfNorth = X(:, 2);

GPSHeading =X(:, 3)\*PI/180;

GPSSpeed = X(:, 4);

GyroAngularRate = X(:, 5)\*PI/180;

Odometer = X(:, 6);

%添加高斯白噪声（平均值= 0，标准差= 8）

posOfEastNoisy=posOfEast + normrnd(0,8,[1078,1]);

posOfNorthNoisy=posOfNorth + normrnd(0,8,[1078,1]);

% 绘制有和无噪声的东向北向曲线

figure(1)

k = 1:length(posOfEast);

subplot(211);

plot(k, posOfEast, '-g','linewidth',1.5);

hold on;

plot(k, posOfEastNoisy, '-r','linewidth',1.5);

legend('东向位置无噪声', '东向位置有噪声');

xlabel('time(s)');

ylabel('posOfEast(m)');

title('东向位置');

grid on;

subplot(212);

plot(k, posOfNorth, '-g','linewidth',1.5);

hold on;

plot(k, posOfNorthNoisy, '-r','linewidth',1.5);

legend('北向位置无噪声', '北向位置有噪声');

xlabel('time(s)');

ylabel('posOfNorth (m)');

title('北向位置');

grid on;

% 生成观测值

Z = [posOfNorthNoisy';posOfEastNoisy';GPSHeading';GPSSpeed';GyroAngularRate'];

**二、Kalman**滤波器设计与滤波估计

问题2：设计Kalman滤波器，完成对东向与北向位置、车辆速度、航向的滤波估计，并能有效处理陀螺零偏及里程计标度因数误差；

问题3：绘制原始轨迹、组合导航轨迹，速度、航向估计、陀螺零偏、里程仪标度因数曲线；

**1、建立状态方程和观测方程**

首先建立状态方程，共有七个状态变量分别为北向位置、东向位置、航向角度、速度、角速率、陀螺零偏、里程计标度因数。方程列写如下：



可以看到变量之间是有相互耦合关系的，这也为后面文图的分析提供数学模型依据。

然后建立观测方程，由于运用的是组合导航设计，因此需要对观测数据进行数据融合，把里程计、陀螺和GPS的数据关联起来，其中GPS输出北向位置、东向位置、航向角度和速度，陀螺输出角速率，里程计则提供脉冲数，得到的观测方程为：



把状态方程和观测方程用矩阵形式来表示：



其中：





其中，比较困难的是确定Q阵和R阵的参数，在后面部分会进行讨论。

**2、建立****Kalman滤波方程**

应用离散系统的Kalman滤波，计算公式如下：



这里我们的状态值初值取观测第一组数据，陀螺零偏取0.1，里程计标度因数同样取0.1即

初始P矩阵取：



由稳定性内容可知，初始X和P的取值对最终结果没有影响。初始零偏设置为0.1，初始标度因数0.3，由稳定性内容可知，初始零偏和标度因数的取值同样对最终结果没有影响。

而对于系统状态噪声和观测噪声，经过滤波结果反复调试最终取值为：





基于上述讨论，编写matlab代码实现kalman最优滤波表示如下：

% 生成观测值

Z = [posOfNorthNoisy';posOfEastNoisy';GPSHeading';GPSSpeed';GyroAngularRate'];

% 构造Q和R阵

Q = diag([2^2 2^2 (0.1)^2 4^2 (0.05)^2 (0.1)^2 (0.01)^2]);

R = diag([8^2 8^2 (0.1)^2 4^2 (0.05)^2]);

% 卡尔曼滤波初值设置

% 0时刻估计值—X00:[Ny, Ex, psi, v, psi\_dot, B, S]；零偏0.1，标度因数0.3

Xk1k1\_hat = [posOfNorth(1);posOfEast(1);GPSHeading(1);GPSSpeed(1);GyroAngularRate(1);0.1;0.3];

% 0时刻P—P00:7\*7 matrix

Pk1k1 = 10\*eye(7);

% 卡尔曼滤波

% X\_hat 行表示状态，列表示观测次数共1078次

X\_hat = zeros(7, length(posOfEast));

X\_hat(:,1) = Xk1k1\_hat;

for idx = 2:length(posOfEast)

if Z(4,idx)<1

R(1,1)=2000;

R(2,2)=2000;

R(4,4)=4;

else

R(1,1)=36;

R(2,2)=36;

R(4,4)=0.005;

end

if X\_hat(3,idx-1) >2\*PI-ang\_swap && Z(3,idx)<ang\_swap

Z(3,idx)=2\*PI+ Z(3,idx);

state=1;

else

if state==1 && Z(3,idx) <2\*PI-ang\_swap

X\_hat(3,idx-1)=X\_hat(3,idx-1)-2\*PI;

end

state=0;

end

% P(k+1|k)

Pk1k = phi(Xk1k1\_hat(3)) \* Pk1k1 \* phi(Xk1k1\_hat(3))' +gamma() \* Q \* gamma()';

% K(k+1)

Kk1 = Pk1k \* h(Odometer(idx))' \* ((h(Odometer(idx)) \* Pk1k \* h(Odometer(idx))' + R) ^ -1);

% X(k+1|k)

Xk1k\_hat = phi(Xk1k1\_hat(3)) \* Xk1k1\_hat;

% X(k+1|k+1)

Xk1k1\_hat = Xk1k\_hat + Kk1 \* (Z(:,idx) - h(Odometer(idx)) \* Xk1k\_hat);

%´æ´¢

X\_hat(:,idx) = Xk1k1\_hat;

% P(k+1|k+1)

Pk1k1 = Pk1k - Kk1 \* h(Odometer(idx)) \* Pk1k;

end

% 绘制有噪声和卡尔曼滤波估计后的南向北向曲线

figure(2);

subplot(211);

k = 1:length(posOfEastNoisy);

plot(k, posOfEastNoisy, '-b','linewidth',1.5);

hold on;

plot(k, X\_hat(2, :), '-r','linewidth',1.5);

legend('东向位置有噪声', '滤波作用东向位置');

xlabel('time(s)');

ylabel('posOfEast(m)');

title(' '东向位置估计曲线');

grid on;

subplot(212);

k = 1:length(posOfNorthNoisy);

plot(k, posOfNorthNoisy, '-b','linewidth',1.5);

hold on;

plot(k, X\_hat(1, :), '-r','linewidth',1.5);

legend('北向位置有噪声', '滤波作用北向位置');

xlabel('time(s)');

ylabel('posOfNorth(m)');

title('北向位置估计曲线');

grid on;

% 绘制有噪声,没有噪声和卡尔曼滤波估计后的南向北向曲线

% x是东向位置，y是北向位置figure(3)

hold on

plot(posOfEast, posOfNorth, '-g','linewidth',1.5);

plot(posOfEastNoisy, posOfNorthNoisy, '--b','linewidth',1.5);

plot(X\_hat(2, :), X\_hat(1, :), '-r','linewidth',1.5);

xlabel('Pos of East(m)');

ylabel('Pos of North(m)');

legend('没有噪声', '有噪声', '卡尔曼滤波');

title('Trajectory');

axis equal;

grid on;

% 航向角度

figure(4)

hold on

k = 1:length(posOfNorthNoisy);

plot(k,GPSHeading\*180/PI,'-b','linewidth',1);

hold on;

plot(k,X\_hat(3, :)\*180/PI,'-r','linewidth',1);

title('航向角估计曲线');

xlabel('time(s)');

ylabel('heading degree(°)');

legend('没有噪声航向角', '卡尔曼滤波航向角');

grid on;

%速度

figure(5)

hold on

k = 1:length(posOfNorthNoisy);

plot(k,GPSSpeed,'-b','linewidth',1);

hold on;

plot(k,X\_hat(4, :),'-r','linewidth',1);

title('速度估计曲线');

xlabel('time(s)');

ylabel('speed(m/s)');

legend('没有噪声速度', '卡尔曼滤波速度');

grid on;

%航向角速率

figure(6)

hold on

k = 1:length(posOfNorthNoisy);

plot(k,GyroAngularRate\*180/PI,'-b','linewidth',1);

hold on;

plot(k,X\_hat(5, :)\*180/PI,'-r','linewidth',1);

title('陀螺角速率估计曲线');

xlabel('time(s)');

ylabel('yaw rate(°/s)');

legend('没有噪声角速率', '卡尔曼滤波角速率');

grid on;

%%陀螺仪零偏

figure(7)

hold on

k = 1:length(posOfNorthNoisy);

plot(k,X\_hat(6, :),'-r','linewidth',1);

title('陀螺零偏估计曲线');

xlabel('time(s)');

ylabel('B(°)');

legend('陀螺零偏');

grid on;

%里程计标度因数

figure(8)

hold on

k = 1:length(posOfNorthNoisy);

plot(k,X\_hat(7, :),'-r','linewidth',1);

title('里程计标度因数估计曲线');

xlabel('time(s)');

ylabel('S');

legend('里程计标度因数');

grid on;

function [PHI] = phi(psi)

PHI = eye(7);

%步长为1

T = 1;

PHI(1, 4) = T \* cos(psi);

PHI(2, 4) = T \* sin(psi);

PHI(3, 5) = T;

end

function [GAMMA] = gamma()

GAMMA = eye(7);

end

function [H] = h(n)

H = zeros(5, 7);

H(1, 1) = 1;

H(2, 2) = 1;

H(3, 3) = 1;

H(4, 7) = n;

H(5, 5) = 1;

H(5, 6) = 1;

end

生成的东向、北向位置Kalman估计结果为：

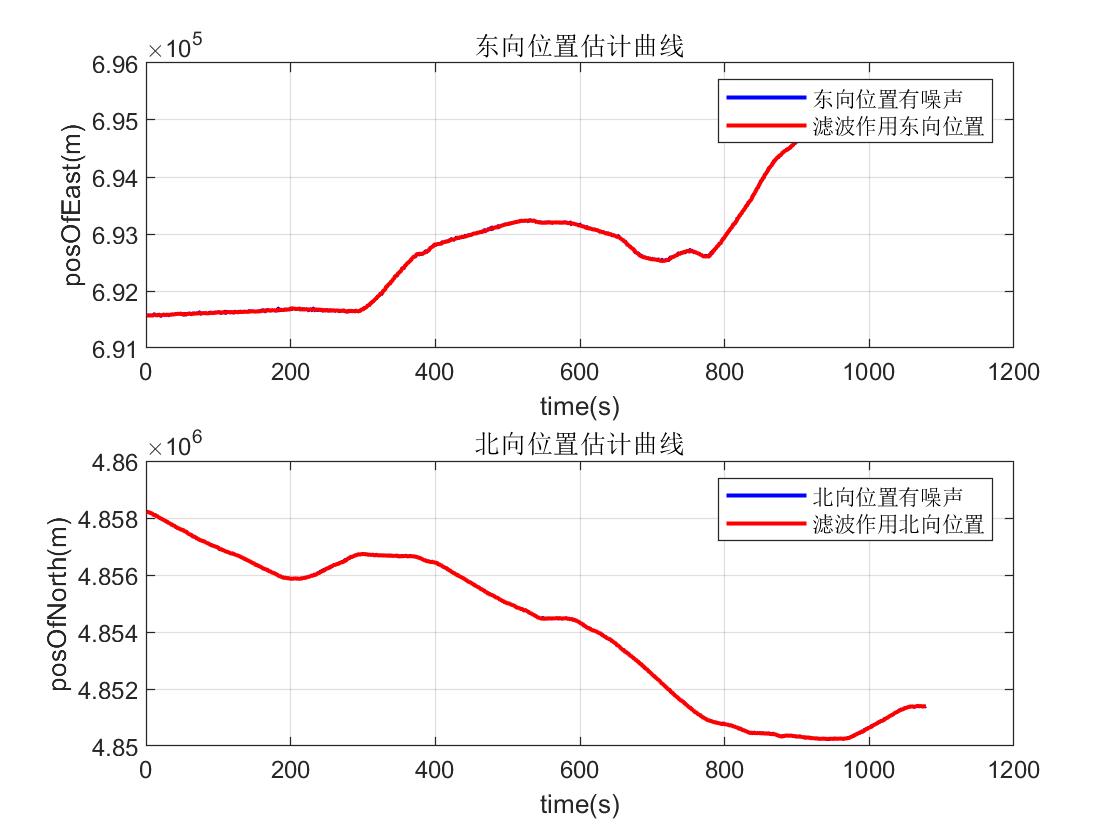


图2.1 东向、北向位置加入噪声数据与滤波估计结果

生成的运动轨迹为：

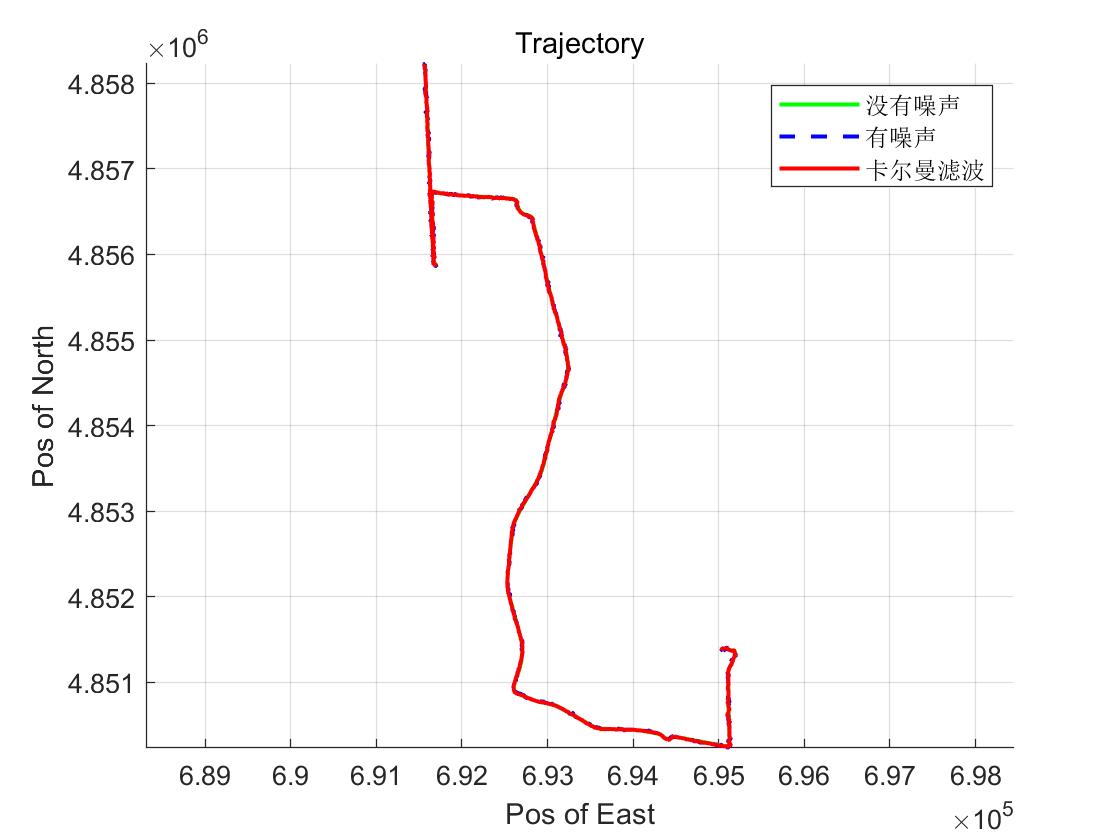


图2.2 组合导航运动轨迹

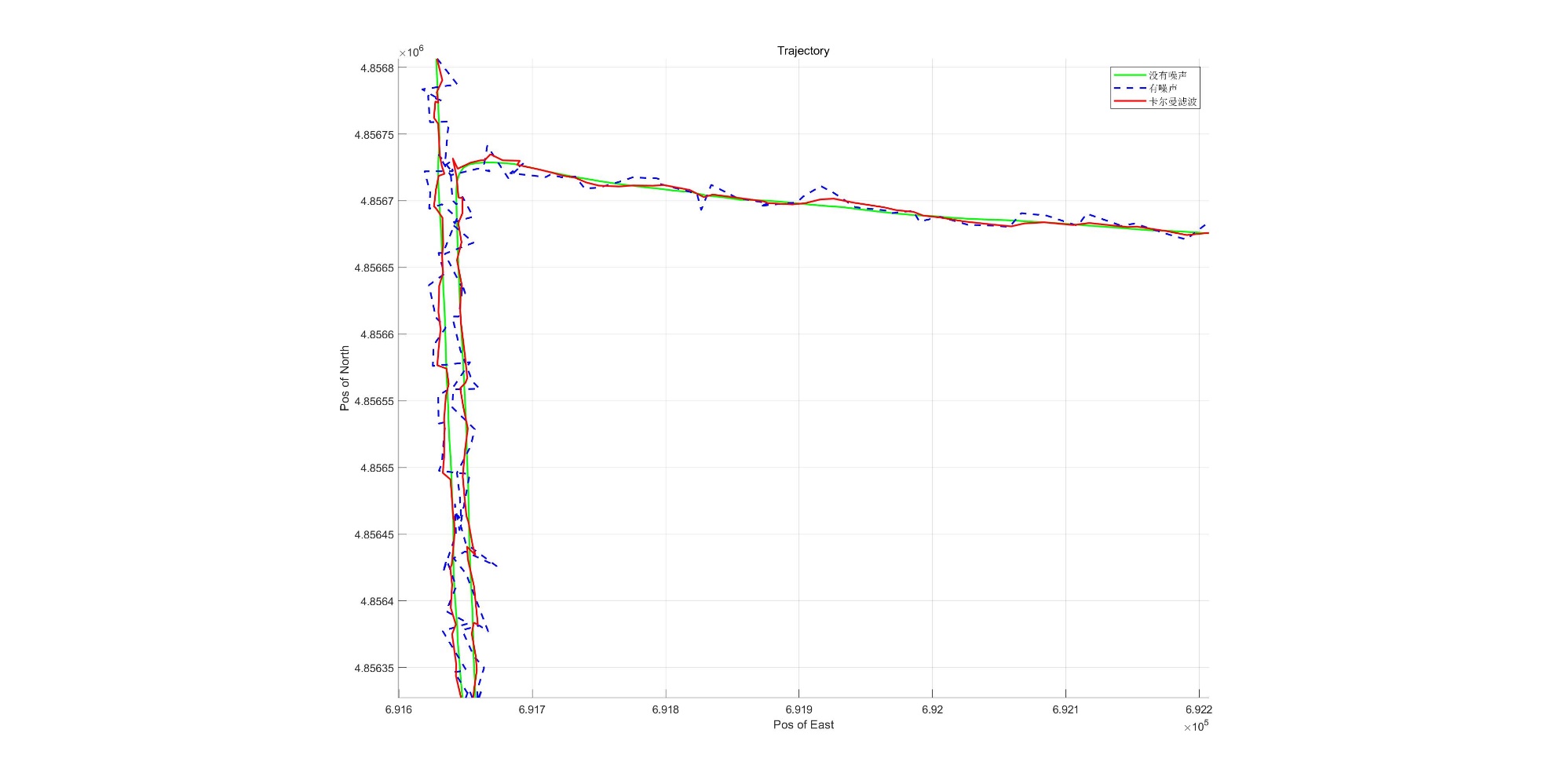


图2.3 组合导航运动轨迹局部放大

生成的速度Kalman估计结果为：

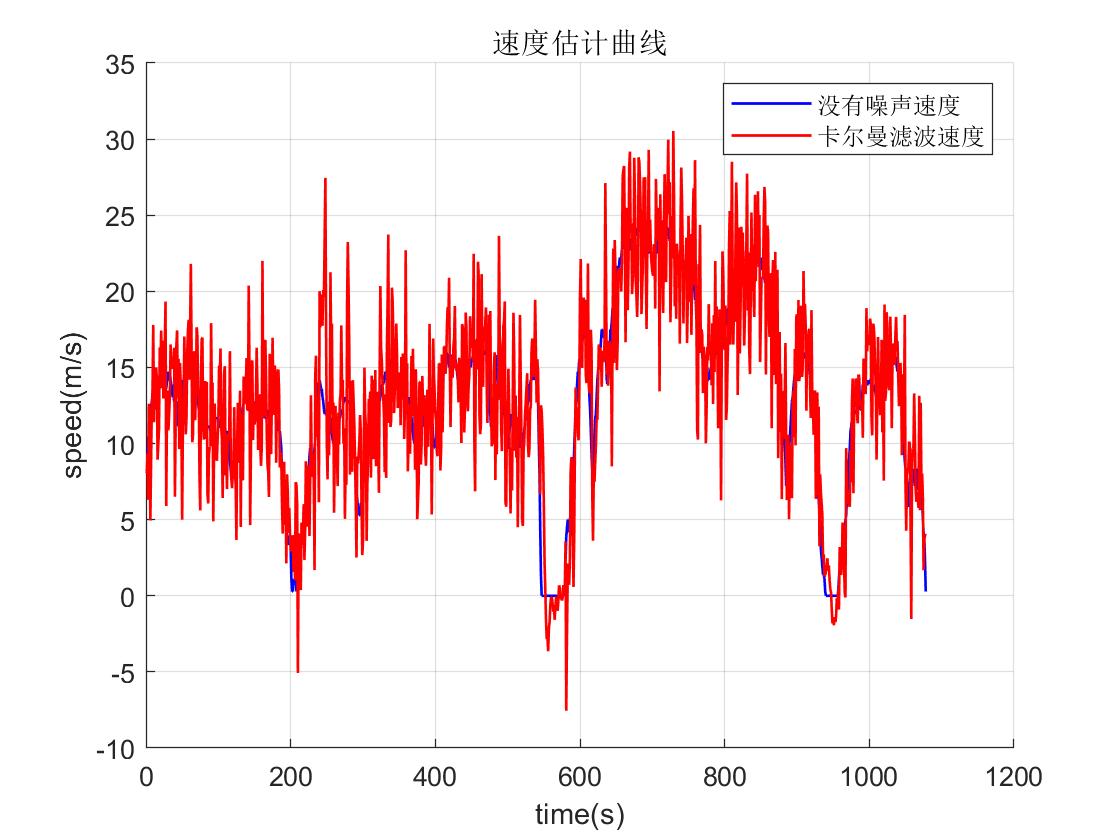


图2.4 速度观测数据与滤波估计结果

生成的航向角Kalman估计结果为：

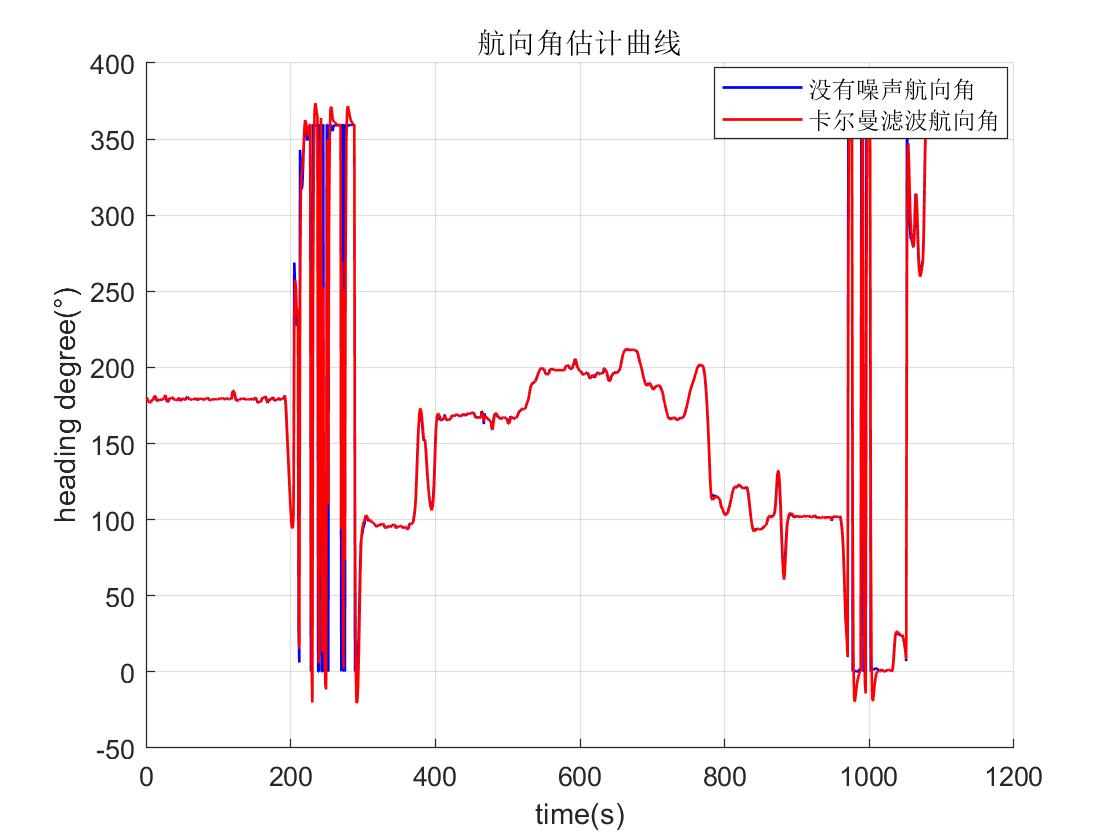


图2.5 航向角观测数据与滤波估计结果

生成的陀螺角速率Kalman估计结果为：

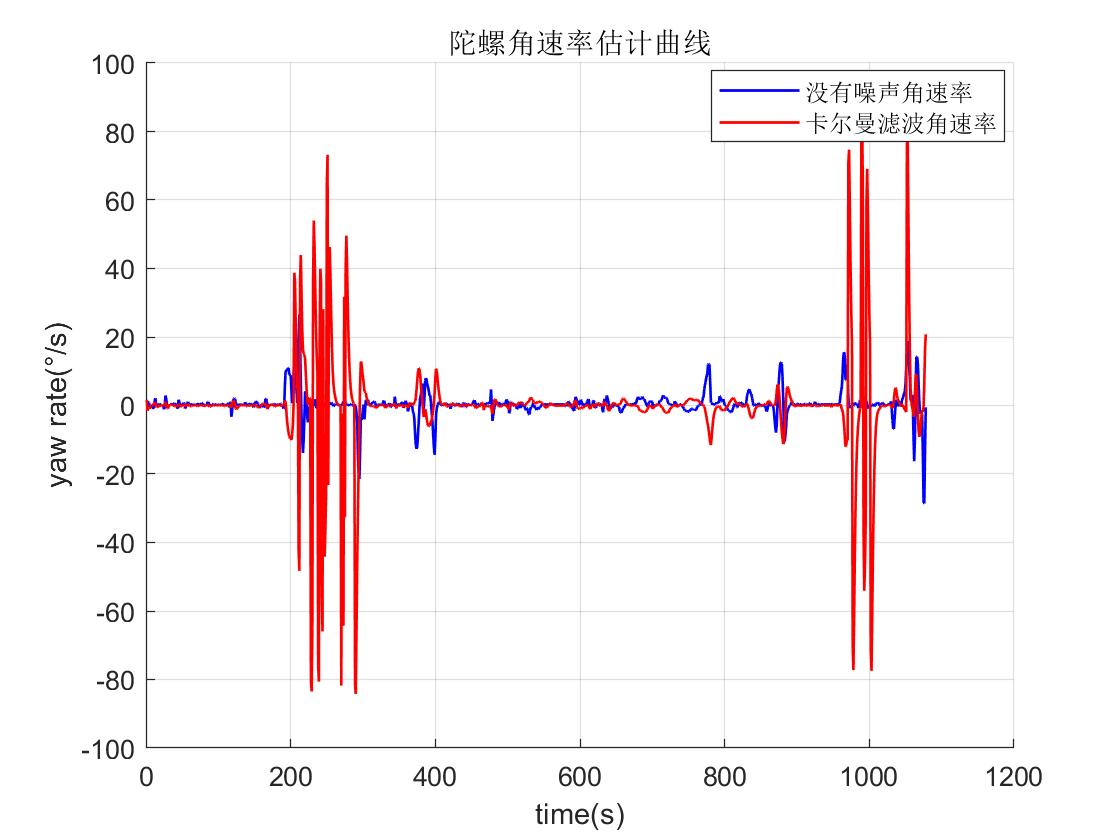


图2.6 陀螺角速率观测数据与滤波估计结果

生成的陀螺零偏Kalman估计结果为：

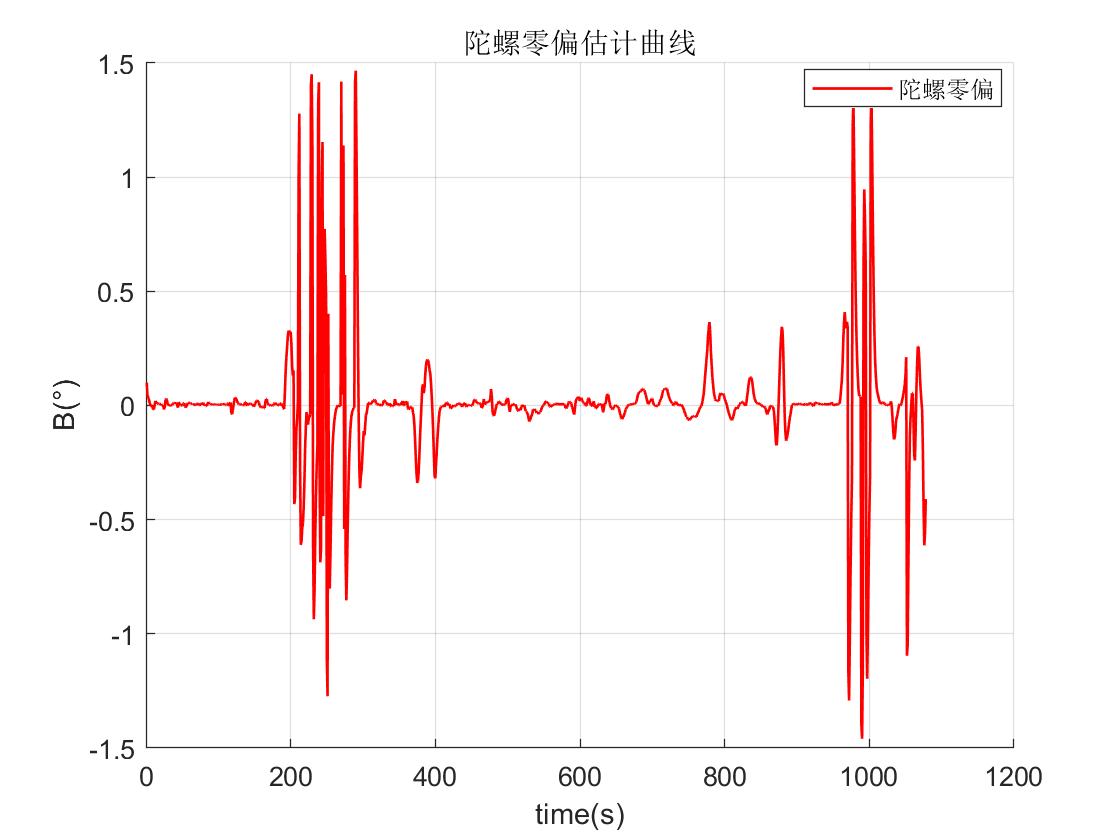


图2.7 陀螺零偏滤波估计结果

生成的里程计标度因数Kalman估计结果为：

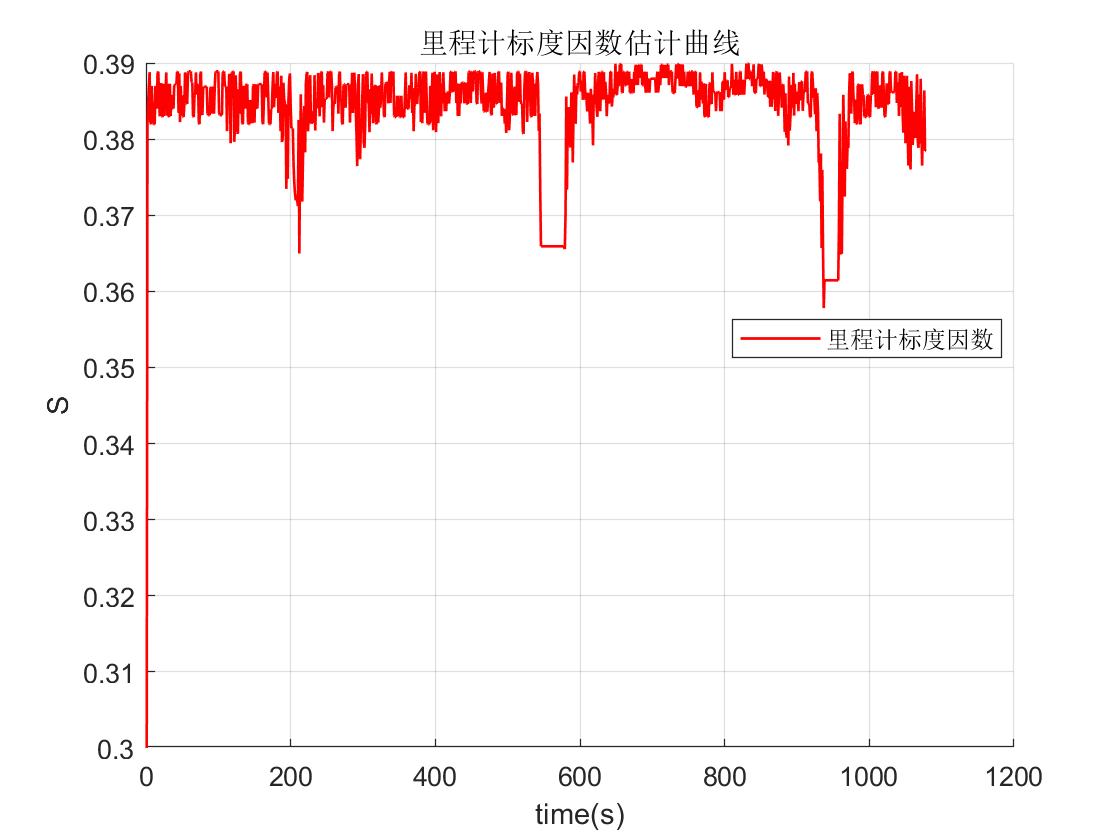


图2.8 里程计标度因数滤波估计结果

可以看出陀螺零偏在0值附近上下波动，里程计标度因数在0.385附近上下波动，是符合Kalman滤波的效果的。尽管状态的初值距离真实值是有偏差的，但若滤波合理，有效利用观测数据，则很快将状态估计到理想数值上，但估计出来的状态不是常值而是始终抖动的。

**三、陀螺仪零偏估计结果的利用**

问题4：分GPS输出有效、无效两种情况，说明陀螺仪的零偏估计结果如何利用，怎样才能最合理地利用；

**1、GPS有效**

观察上文列出的观测方程，由观测方程中航向角和陀螺角速率的部分：



可以看出，陀螺仪零偏估计B用来反馈修正陀螺仪测量出来的角速率。

**2、GPS无效**

当GPS受到遮挡，输出无效时，观测误差新息增大，增益K降低，状态估计更依赖于状态递推，减小观测误差对状态估计的修正作用，因此陀螺仪零偏估计不准确。

但由于零偏在短时间内变化不大，因此可以在GPS输出无效情况下使用GPS失效前一时刻的陀螺仪零偏值反馈修正陀螺仪角速率输出，直到GPS恢复正常，再用新的零偏估计值加以修正。而在GPS正常工作的情况下，组合导航系统滤波效果好，使用实时估计出的零偏值来反馈修正陀螺仪测量出来的角速率。

**四、Q****、R协方差阵的设置**

**1、Q、R矩阵的取值**

问题5：给出合理的Q、R协方差阵，并说明每项数据的设置理由；

Q和R的取值决定了观测信息来修正状态值的估计的利用率，Q中元素q增大，K增大，观测信息修正利用率增高；R中元素r增大，K减小，观测信息修正利用率降低。理论上，如果希望kalman滤波的效果最好，应该令Q阵和R阵接近理想的Q和R。

在该问题中，我们选取的Q和R矩阵数值为:





Q的选取反映系统噪声的影响程度，东向和北向位置由GPS测量，递推取决于车辆运动速度和航向角度，取位置噪声方差为(2)2。航向角方程采用的是陀螺仪角速率积分的方法，当采样周期较小，车辆转向可以认为是匀速转动，陀螺仪精度相对较高，考虑陀螺仪的精度，取航向噪声方差为(0.1)2。数据采样周期T=1s，采样周期相对来说比较小，在采样周期内将车辆视作匀速直线运动，根据GPS测速精度取值(4)2取为较为符合实际情况。同样根据陀螺仪精度，取陀螺仪角速率噪声方差为(0.05)2，陀螺零偏噪声方差为(0.1)2。

里程计误差应该是最小的，因此里程计标度因素噪声方差选择为(0.01)2。

R的选取反映观测噪声的影响程度，由于题目中位置量测噪声已知，因此位置量测噪声方差直接取为(8)2。而其它观测数据并没有添加噪声。但是为了保证计算过程的稳定性，不能把那些没加噪声的数据设置为0，我们选取与Q矩阵中与对应参数同样的数值作为R中的元素，即航向量测噪声方差取为(0.1)2，速度量测方差取为(4)2，陀螺仪角速率量测噪声方差取(0.05)2。

此外，为了能够定量地衡量kalman滤波的效果，我们可以计算滤波值和原始数据的误差标准差，通过误差标准差来衡量kalman滤波的结果与真实值的偏离程度。计算标准差的代码如下：

%将原始数据中东向和北向位置视为真实值,将其用来衡量滤波的好坏

E\_error\_array= X(:, 1)- X\_hat(2, :) ';

E\_error=std(E\_error\_array);

N\_error\_array= X(:, 2)- X\_hat(1, :) ';

N\_error=std(N\_error\_array);

figure(9);

subplot(211)

hold on;

plot(E\_error\_array);

title('东向位置误差波形图');

xlabel('time(s)');

ylabel('东向位置误差(m)');

grid on;

subplot(212)

hold on;

plot(N\_error\_array);

title('北向位置误差波形图');

xlabel('time(s)');

ylabel('北向位置误差(m)');

grid on;

获得的估计误差波形图为：

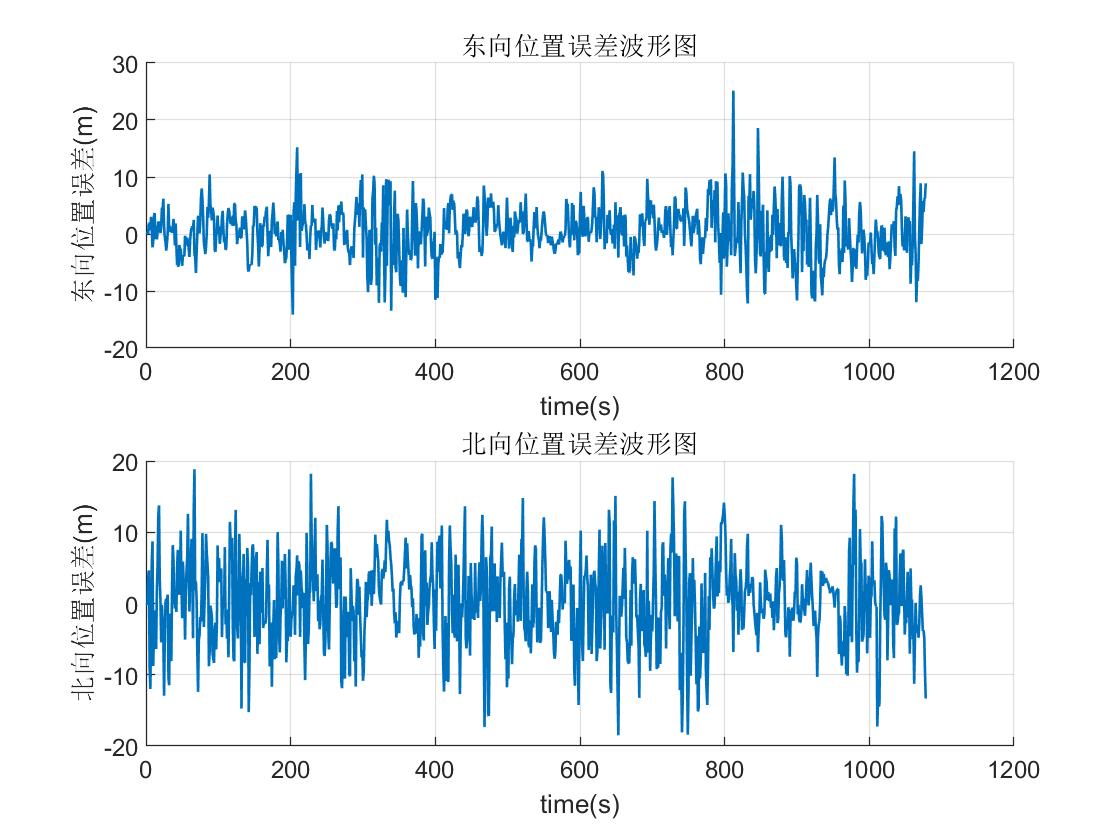


图4.1 估计误差波形图

**2、R理想，缩放Q**

问题6：使用理想R，对理想Q进行0.1~10倍的比例缩放，对不同情况下的轨迹滤波结果进行分析；

我们分别取0.1Q，Q，5Q以及10Q绘制轨迹曲线，从而实现对Q阵变化的分析。缩放的轨迹滤波结果为：

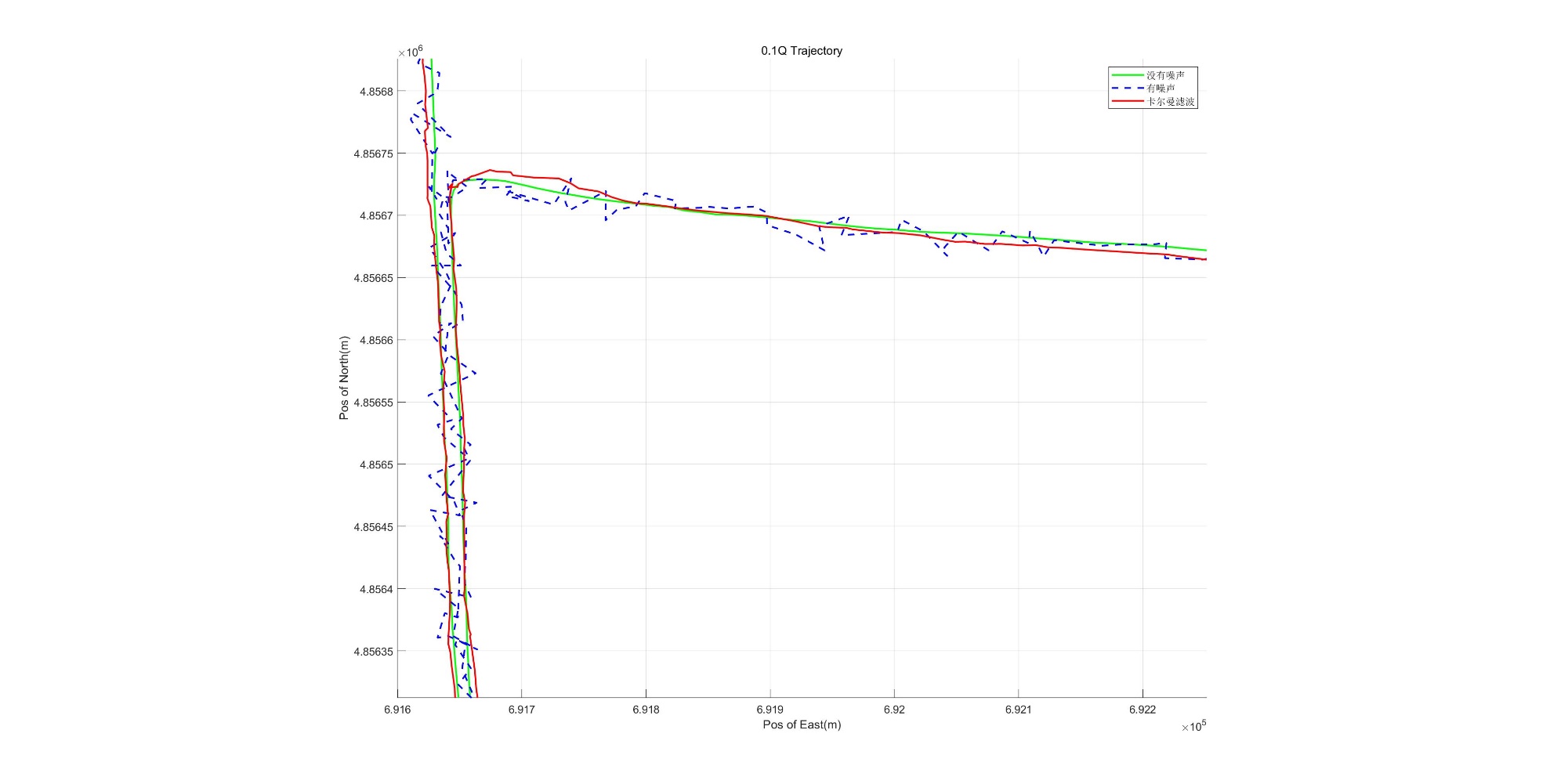


图4.2 0.1Q轨迹滤波结果

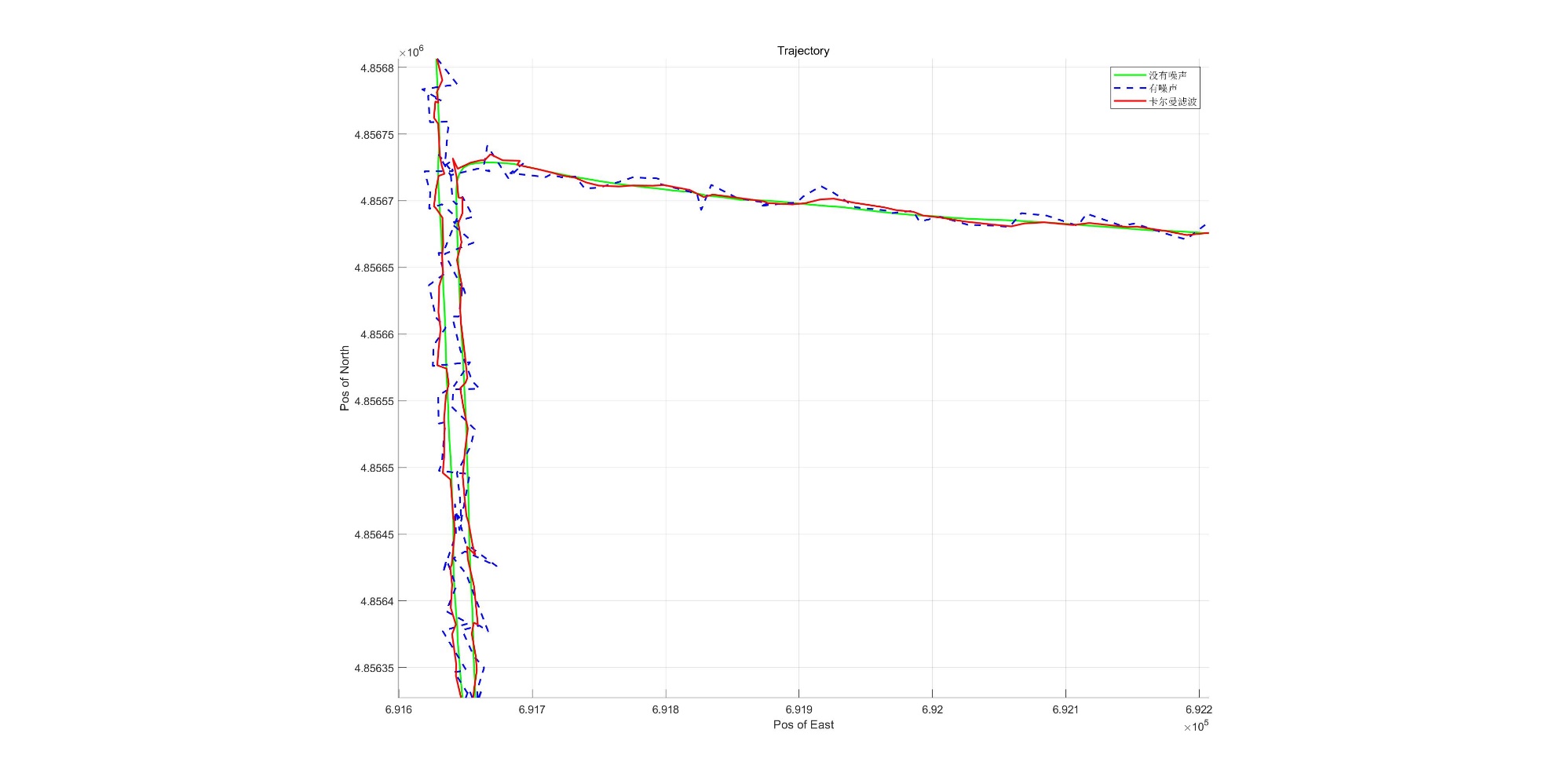


图4.3 Q轨迹滤波结果

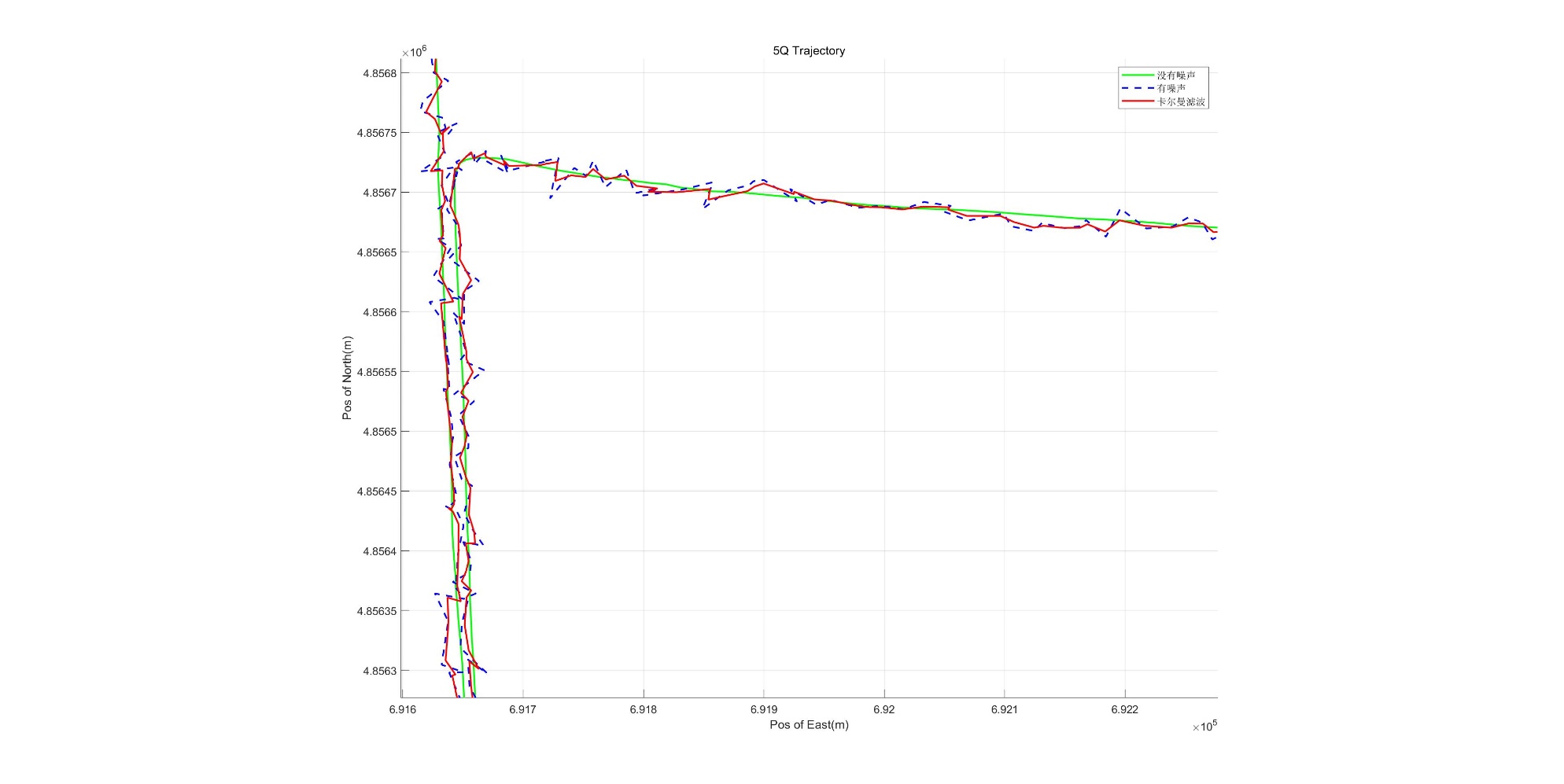


图4.4 5Q轨迹滤波结果

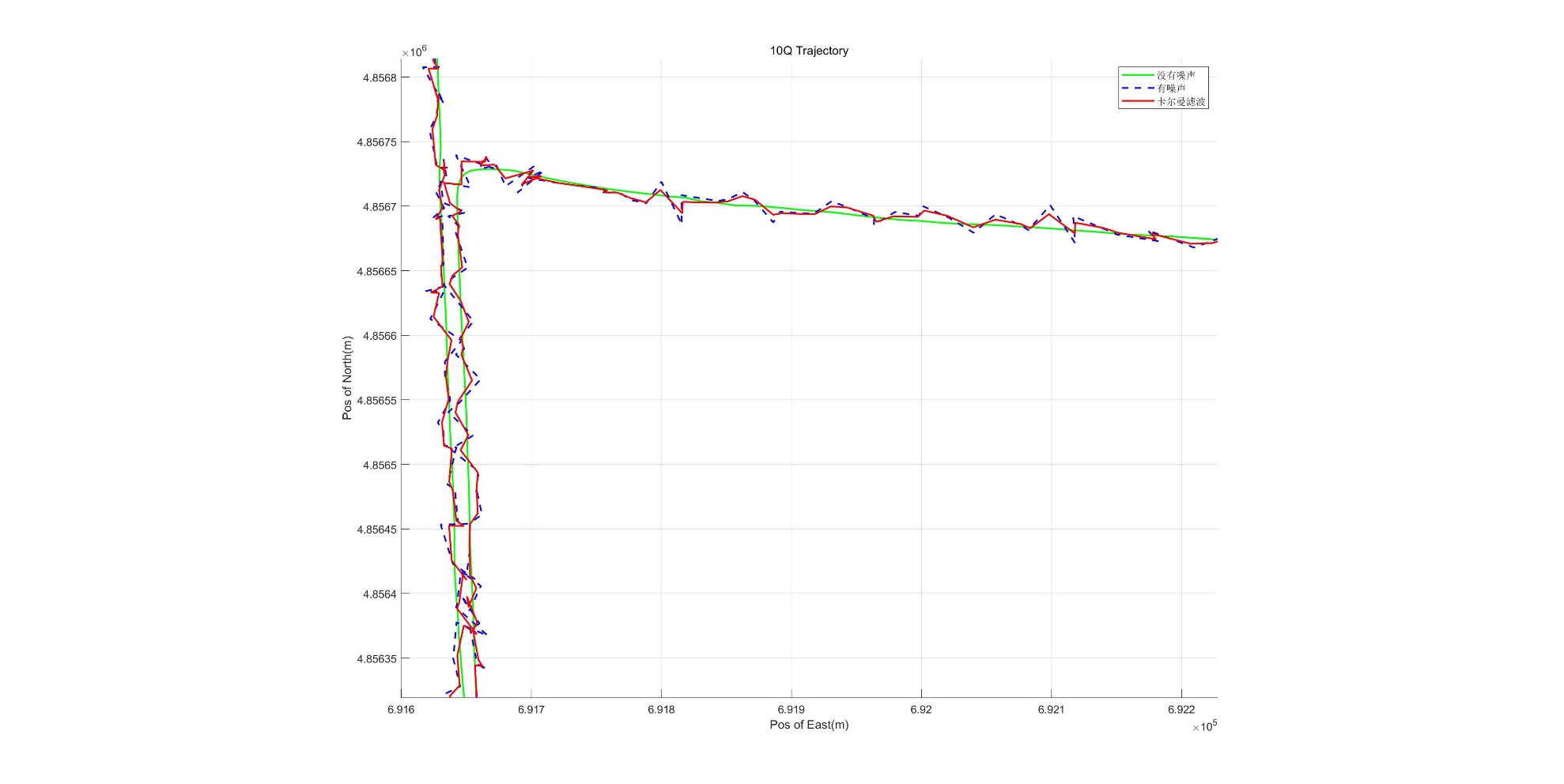


图4.5 10Q轨迹滤波结果

从结果分析来看，Q增大，K增大，观测信息修正利用率增高，状态估计更多依靠观测数据的修正，因此滤波曲线越来越靠近观测数据，但由于观测数据中噪声明显，故滤波曲线的平滑度变差。Q减少，状态估计更多依靠状态递推，平滑度变好，但由于几乎没有观测数据的修正，估计结果的准确性下降，并且会影响滤波的及时性，产生时滞。。

**3、Q理想，缩放R**

问题7：使用理想Q，对理想R进行0.1~10倍的比例缩放，对不同情况下的轨迹滤波结果进行分析；

类似的，我们分别取0.1R，R， 5R以及10R绘制轨迹曲线，从而实现对R阵变化的分析。缩放的轨迹滤波结果为：

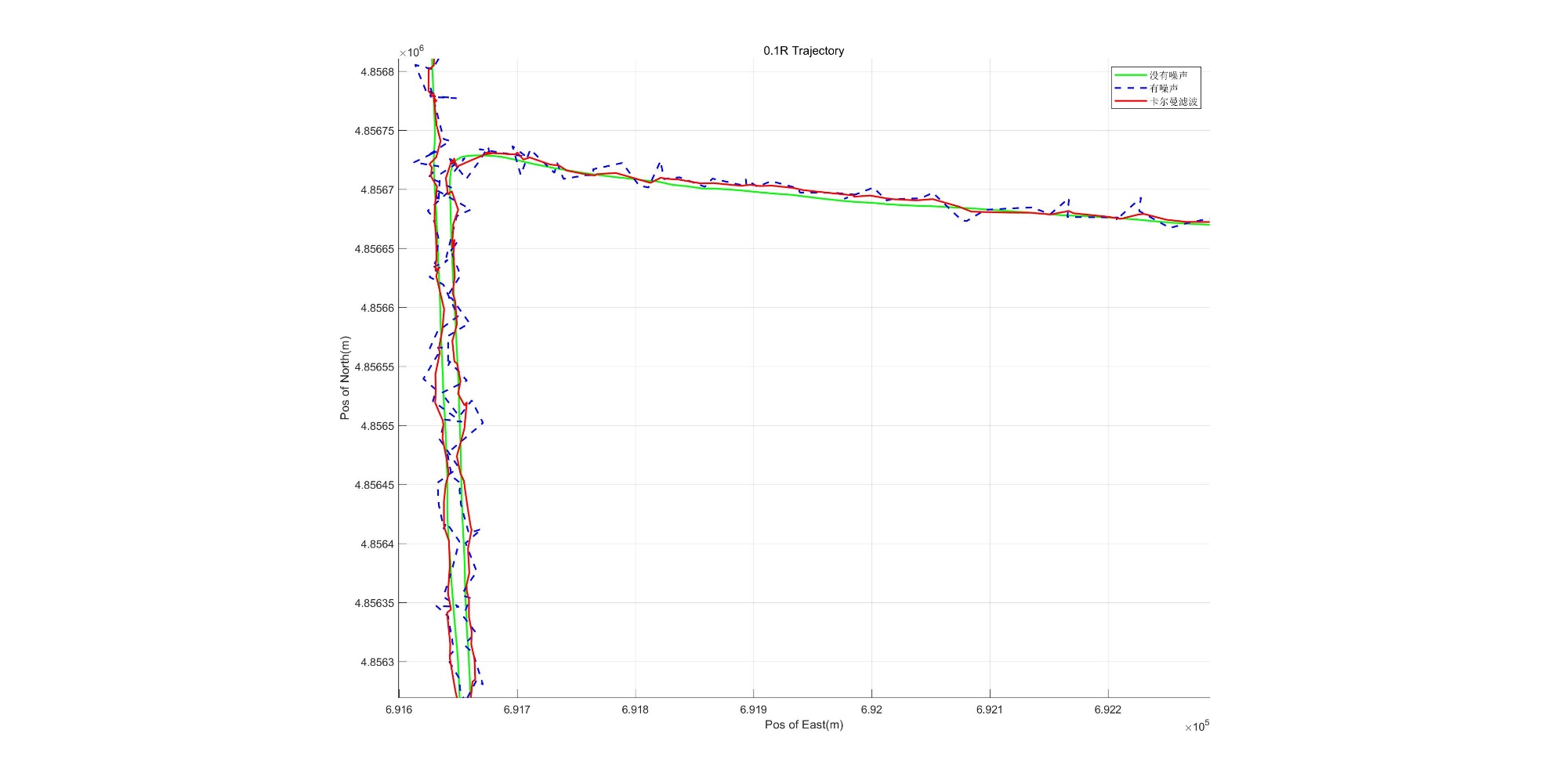


图4.6 0.1R轨迹滤波结果

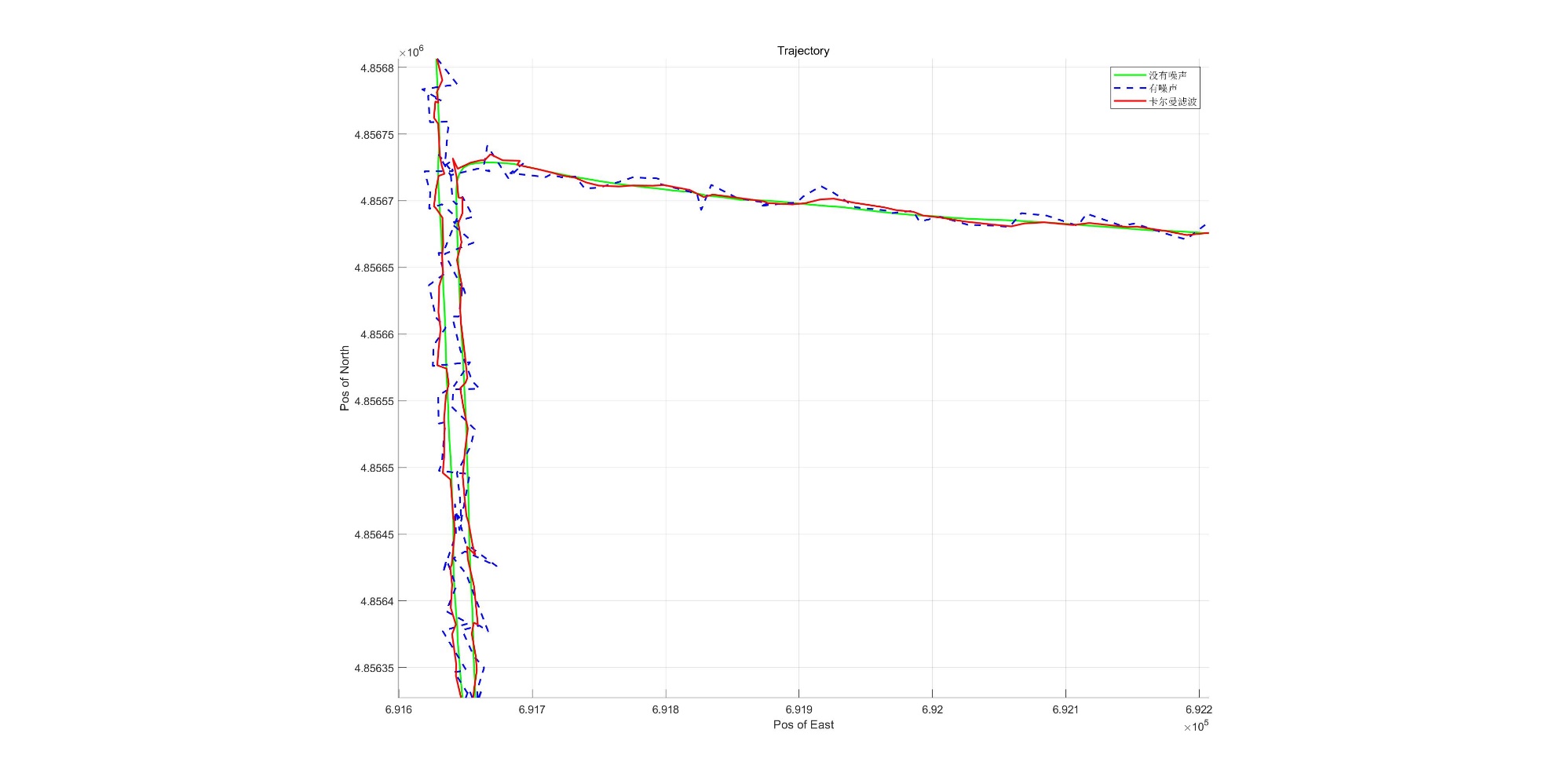


图4.7 R轨迹滤波结果

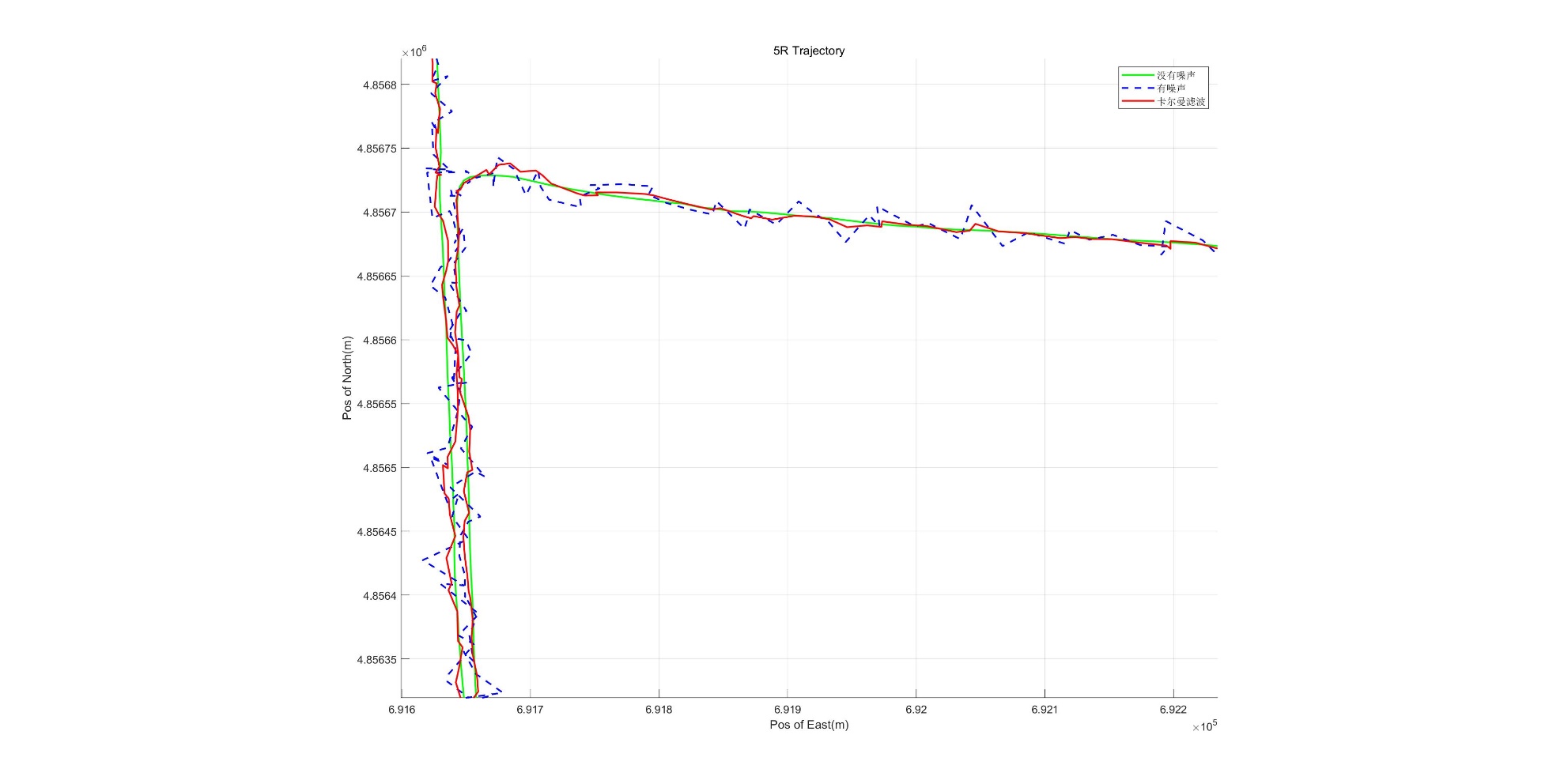


图4.8 5R轨迹滤波结果

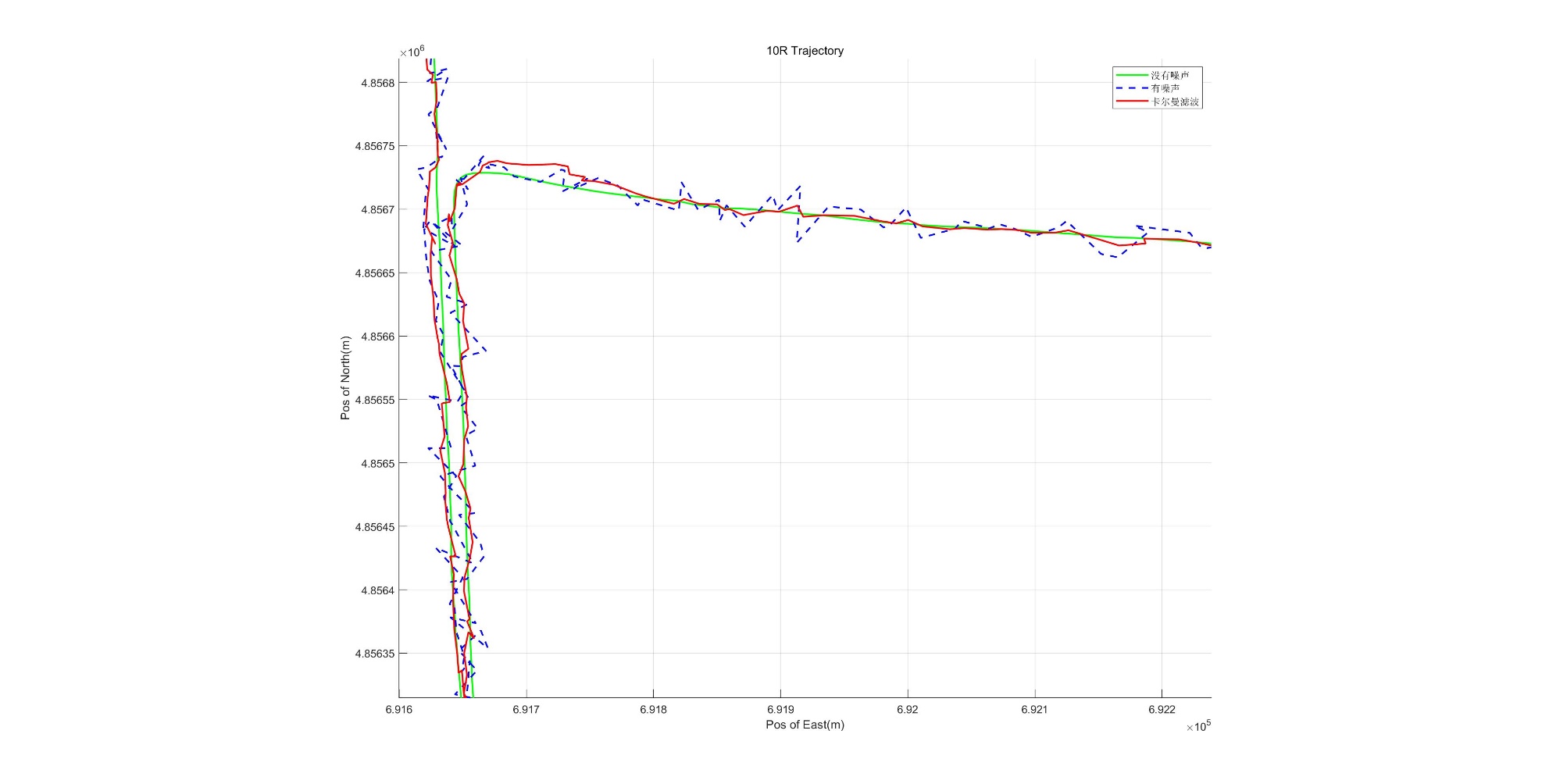


图4.9 10R轨迹滤波结果

从结果分析来看，R增大，K减小，观测信息修正利用率降低，状态估计更多依靠状态递推，因此滤波曲线越来越平滑，但由于缺少有效的修正，滤波曲线准确性性下降，并且会影响滤波的及时性，产生时滞。R减少，状态估计更多依靠观测数据修正，平滑度变差，但估计结果的准确性上升。

**五****、里程计标度因数估计值变化的原因**

问题8：针对滤波模型分析里程计标度因数估计值变化的可能原因；

由观测方程里程计标度因数部分可知：



GPS速度与里程计计数值和里程计标度因数存在耦合关系，因此里程计标度因数变化的原因可能是GPS速度量测值发生较大的变化或者里程计计数值发生突变。

以上两点原因可以得到验证。观察里程计标度因数滤波估计结果，在200s、600s、 950s左右标度因数发生了剧烈变化。对GPS速度观测数据图像查看，速度也在200s、600s、 950s左右发生了较大的变化，真实的观测速度数值趋近于0，估计速度仍在抖动。同时，查看原始数据txt文件中的里程计脉冲数值，也在上述三个时间点，脉冲数字出现了近似为0的情况。观察车辆的轨迹曲线会发现在这三处时间点发生了明显的转弯。

基于上述讨论，我们对标度因数估计值变化的原因进一步分析：

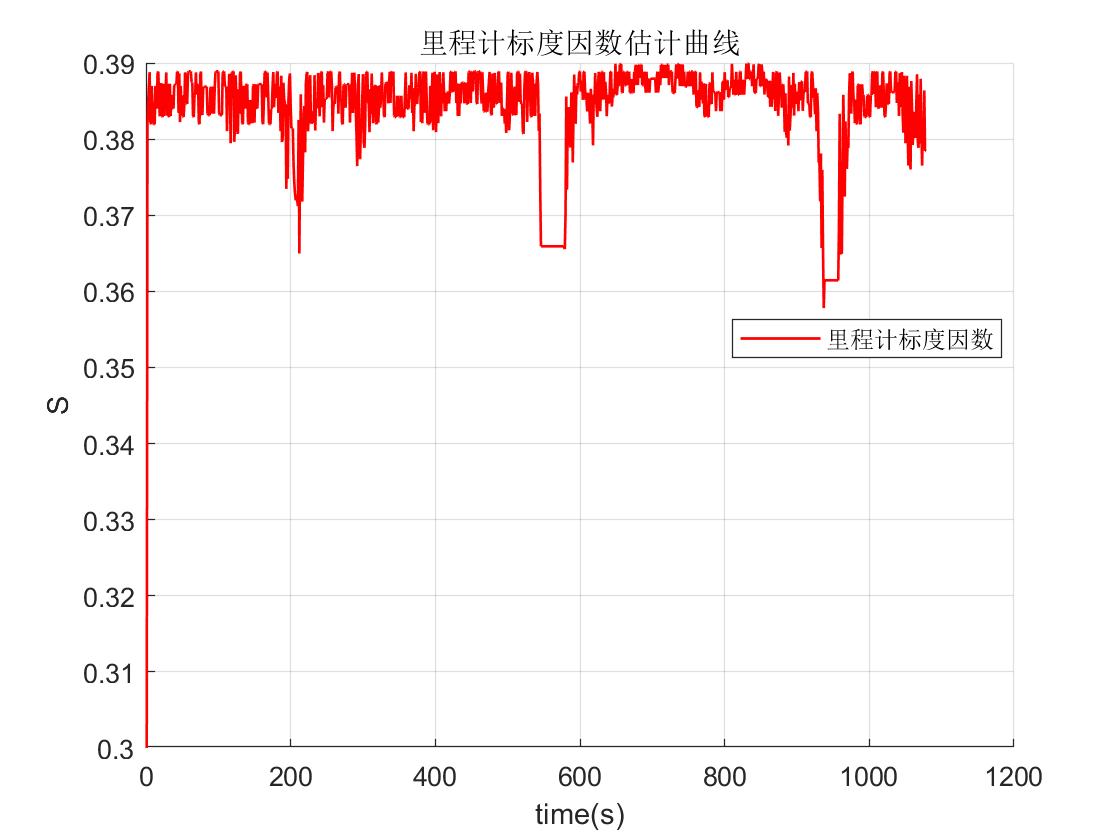


图5.1 里程计标度因数滤波估计结果

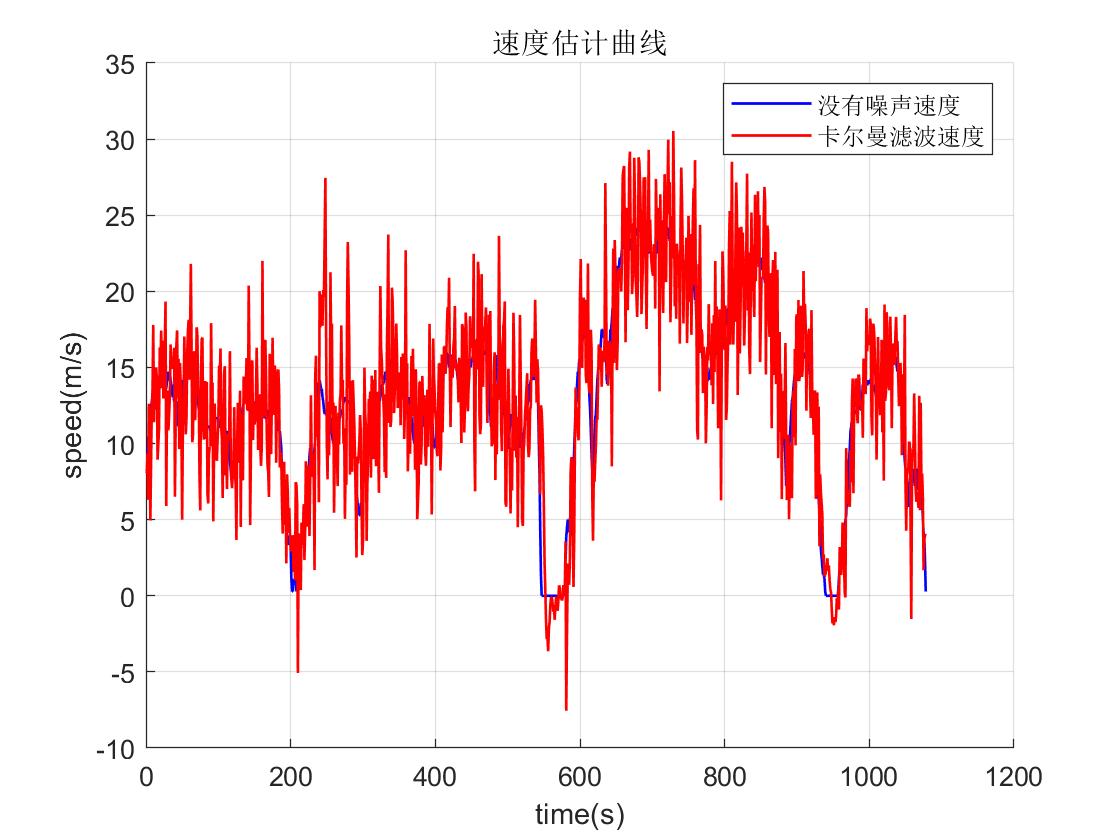


图5.2 速度观测数据与滤波估计结果

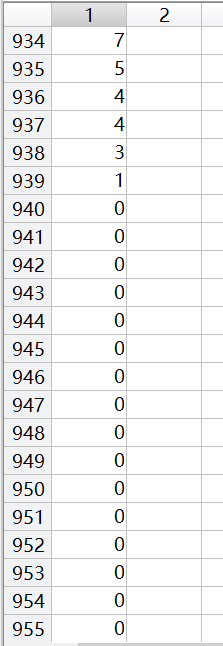
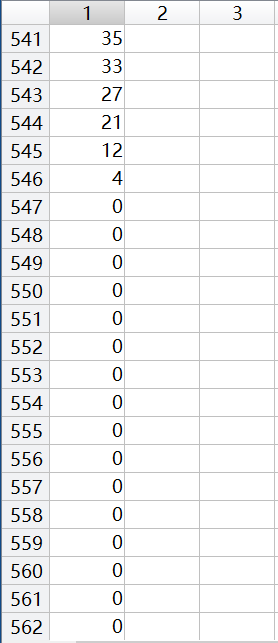
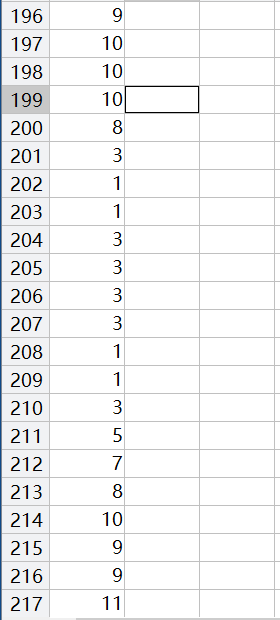


图5.3 里程计脉冲数值

车辆在缓慢移动时，在200s、600s、 950s左右要进行转弯，速度降低至0，速度有较大的突变，同时程计记录了车轮转动走过的圈数，GPS测速时由于往返的原因会被低估，因此估计出来的标度因数数值会比其他时刻要小很多，标度因数估计值发生较大的变化。

**六、消除航向数值抖动对位置精度的影响**

问题9：设计合理的处理方法，消除车辆沿正南、正北行进时航向数值抖动对位置精度的影响，并仿真验证；

航向角抖动原因是在0°附近摆动时，由于航向角的范围是0°~360°，因此会在0°~360°之间抖动。为了消除这个抖动，我们需要想办法将航向角度进行转换。转换的方法有两种：

方法一：修改航向角的范围到-180°~+180°。

该方法可以解决车辆向北行驶时航向角度抖动问题，但修改汽车运动方向，车辆向南行驶时仍会发生同样的严重抖动问题。

方法二：不再限制航向角范围。

在循环部分附加一段处理航向角度观测数据和航向角度估计的程序，其中state初始赋值为state=0，ang取值为ang =6\*PI/180。

if X\_hat(3,idx-1) >2\*PI-ang && Z(3,idx)<ang

Z(3,idx)=2\*PI+ Z(3,idx);

state=1;

else

if state==1 && Z(3,idx) <2\*PI-ang

X\_hat(3,idx-1)=X\_hat(3,idx-1)-2\*PI;

end

state=0;

end

运行后得到的航向角度观测数据和航向角度估计结果如图所示，可以看出对航向角度的估计不会再有突变出现，且曲线较为平滑。

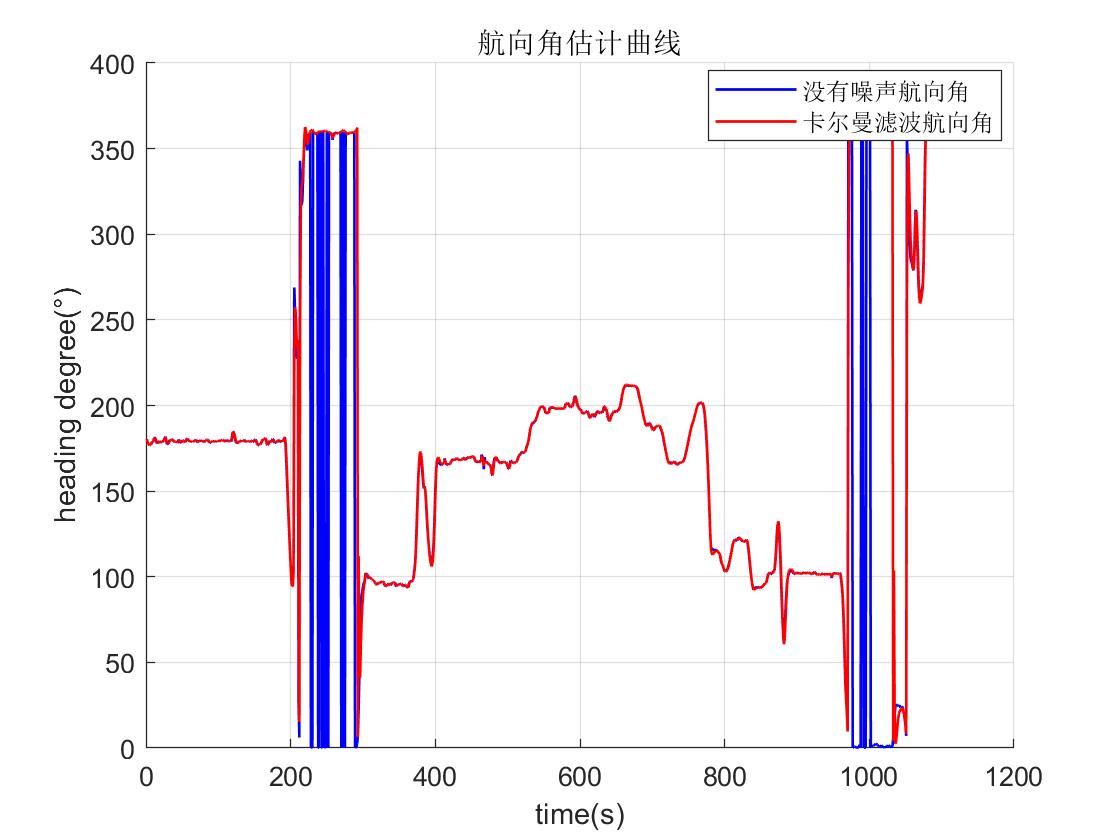


图6.1 处理后的航向角观测数据与滤波估计结果

**七、滤波计算对速度估计的作用**

问题10：分析速度估计结果与速度真值之间的关系，并从数学模型角度出发分析滤波计算的作用的；

速度观测数据与速度滤波的结果如图所示，可以看出速度滤波估计值围绕真值抖动。因此估计值与速度真值（即速度观测数据）的关系为估计值逼近真值。

从数学模型角度分析，观测方程中GPS观测速度表达式为：



速度的观测值和里程计计数值与标度因数之间具有耦合关系，用里程计间接量测速度，在单位时间内用里程计表示的走过的距离和用GPS测量出来的速度相等。对于速度估计来说，滤波计算的作用是利用GPS速度量测值来估计里程计的标度因数S，进而间接获得速度。由于滤波过程存在噪声，因此这种间接求得的速度存在抖动。

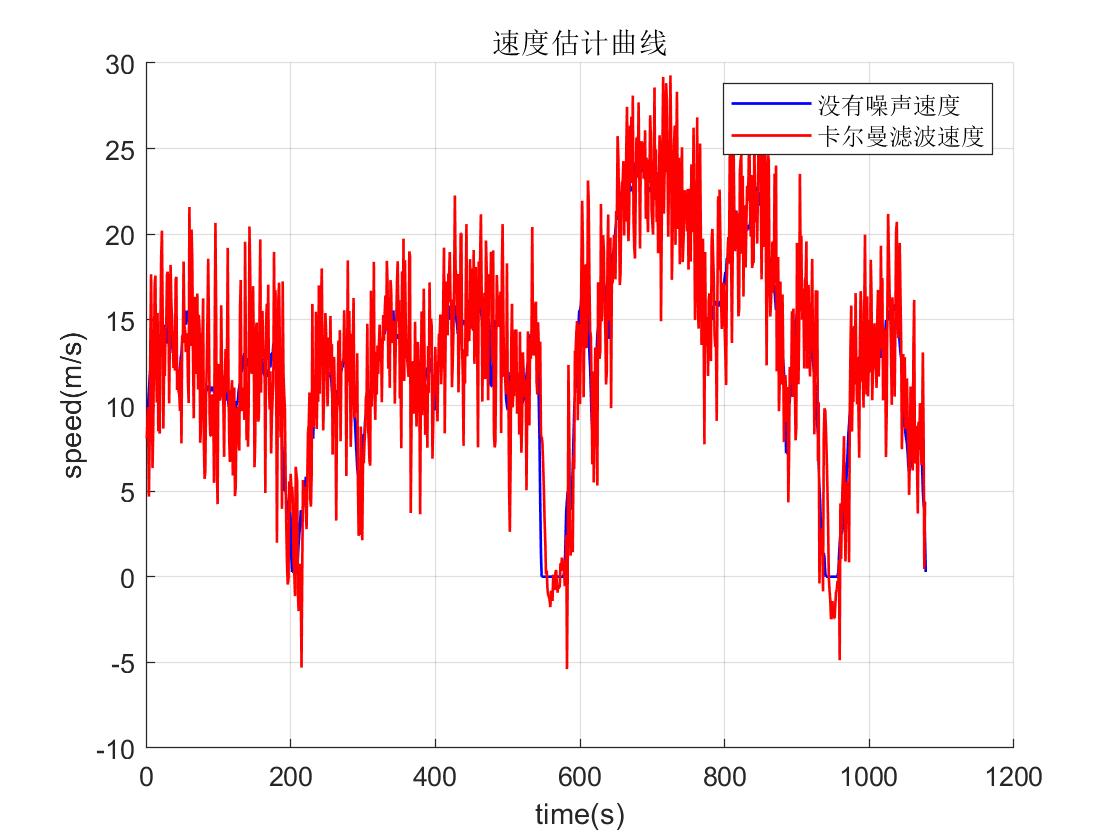


图7.1 速度观测数据与滤波估计结果

**八、里程计构成速度观测的模型改进方案**

问题11：请思考利用里程计构成速度观测的模型改进方案（观测方程、R），并对比速度估计结果；

建立新的Kalman滤波模型，根据上面的分析可知，滤波计算的作用是利用GPS速度量测值来估计里程计的标度因数S，进而间接获得速度，改变为直接用GPS测量的速度对速度状态进行观测。并且根据在单位时间内用里程计表示的走过的距离和用GPS测量出来的速度相等，改变为增加里程计脉冲数作为新的观测量在观测方程中表达。系统状态方程保持不变，改变后的观测方程为：





其中r66取为(0.01)2。对应的代码改变为：

Q = diag([2^2 2^2 (0.1)^2 4^2 (0.05)^2 (0.1)^2 (0.01)^2]);

R = diag([8^2 8^2 (0.1)^2 4^2 (0.05)^2 (0.01)^2]);

Kk1 = Pk1k \* h(Xk1k1\_hat(7))' \* ((h(Xk1k1\_hat(7)) \* Pk1k \* h(Xk1k1\_hat(7))' + R) ^ -1);

Xk1k\_hat = phi(Xk1k1\_hat(3)) \* Xk1k1\_hat;

Xk1k1\_hat = Xk1k\_hat + Kk1 \* (Z(:,idx) - h(Xk1k1\_hat(7)) \* Xk1k\_hat);

X\_hat(:,idx) = Xk1k1\_hat;

Pk1k1 = Pk1k - Kk1 \* h(Xk1k1\_hat(7)) \* Pk1k;

function [H] = h(number)

H = zeros(6, 7);

H(1, 1) = 1;

H(2, 2) = 1;

H(3, 3) = 1;

H(4,4)=1;

H(5, 5) = 1;

H(5, 6) = 1;

H(6,4)=1/number;

end

改进后的速度估计的结果如图：

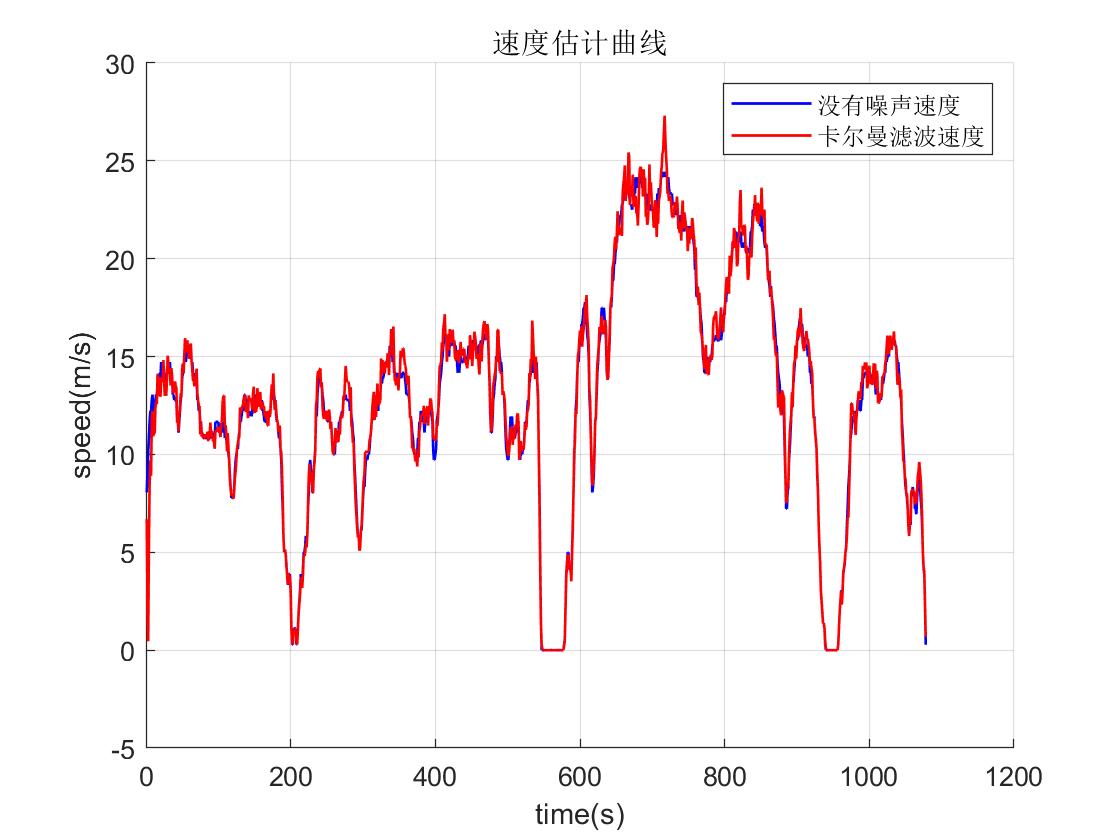


图8.1 改进后的速度观测数据与滤波估计结果

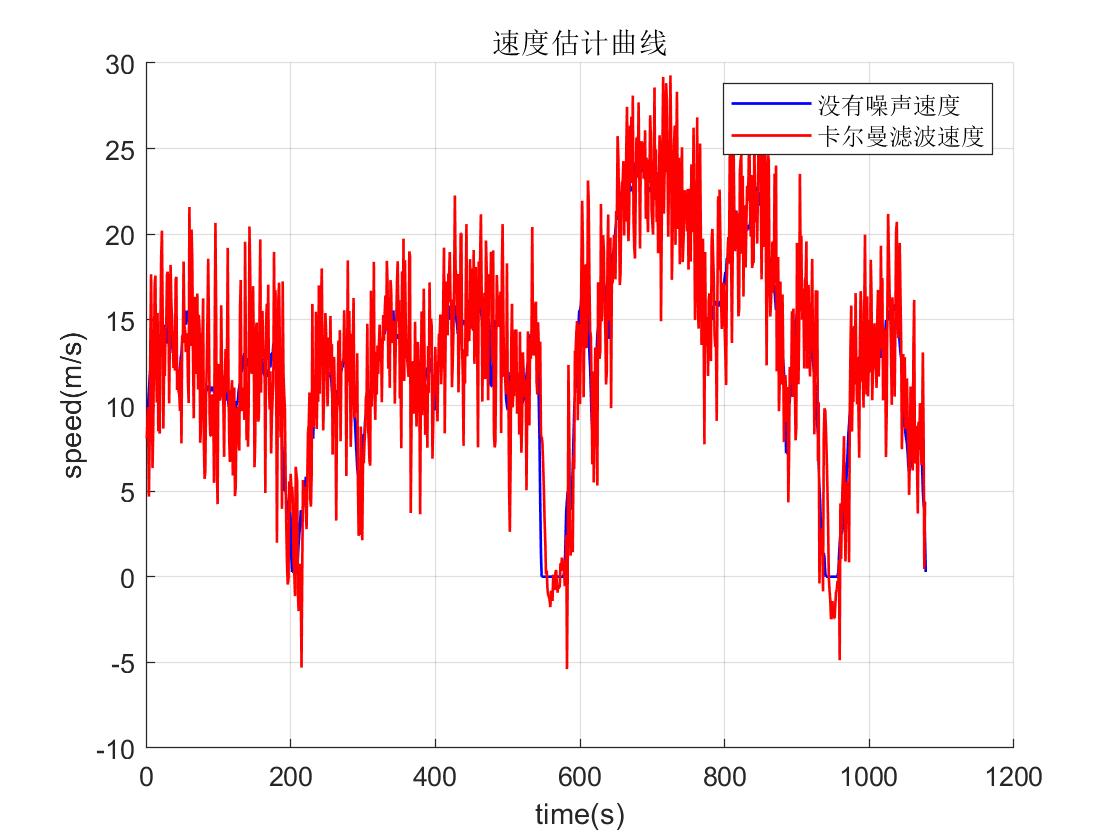


图8.2 未改进的速度观测数据与滤波估计结果

对比两种方案的结果，可以看出改进后的速度估计曲线较之前相比，曲线更为平滑且更加贴近速度的观测值，原因为:之前的速度通过里程计间接测量得到，而改进的观测方程用GPS对速度直接观测，所以速度滤波结果与速度测量输出重合度很高，改进后的速度滤波估计效果得到了提升。

**九、序贯处理方法求解状态滤波估计**

问题12：使用序贯处理方法进行求解，与常规计算方法结果进行比较。

序贯处理的思想是将观测更新中对Z(k)的集中处理分散为Z(k)各分量组的顺序处理，是对高阶矩阵的求逆转变为对低阶矩阵的求逆，以便有效降低计算量。该算法的数学表达式为：



在本问题中，即共进行五次计算。其中，





对应的序贯处理求解滤波问题的matlab代码为：

%构造Q和R阵

Q = diag([2^2 2^2 (0.1)^2 4^2 (0.05)^2 (0.1)^2 (0.01)^2]);

R = diag([8^2 8^2 (0.1)^2 4^2 (0.05)^2]);

%卡尔曼滤波初值设置

Xk1k1\_hat = [posOfNorth(1);posOfEast(1);GPSHeading(1);GPSSpeed(1);GyroAngularRate(1);0.1;0.3];

Pk1k1 = 10\*eye(7);

X\_hat = zeros(7, length(posOfEast));

X\_hat(:,1) = Xk1k1\_hat\_ka;

for idx = 2:length(posOfEast)

if Z(4,idx)<1

R(1,1)=2000;

R(2,2)=2000;

R(4,4)=4;

else

R(1,1)=36;

R(2,2)=36;

R(4,4)=0.005;

end

if X\_hat\_ka(3,idx-1) >2\*PI-ang\_swap && Z(3,idx)<ang\_swap

Z(3,idx)=2\*PI+ Z(3,idx);

state1=1;

else

if state1==1 && Z(3,idx) <2\*PI-ang\_swap

X\_hat\_ka(3,idx-1)=X\_hat\_ka(3,idx-1)-2\*PI;

end

state=0;

end

%序贯处理的卡尔曼滤波

Pk1k = phi(Xk1k1\_hat(3)) \* Pk1k1 \* phi(Xk1k1\_hat(3))' +gamma() \* Q \* gamma()';

Xk1k\_hat = phi(Xk1k1\_hat(3)) \* Xk1k1\_hat;

% X(k+1|k+1)和P(k+1|k+1)

Pk\_0= Pk1k;

Xk\_0\_hat=Xk1k\_hat;

Hk=h(Odometer(idx));

%r=1

Kk\_1= Pk\_0 \* Hk(1,:)'\* inv(Hk(1,:) \* Pk\_0 \* Hk(1,:)'+ R(1,1)) ;

Xk\_1\_hat=Xk\_0\_hat+Kk\_1\*( Z(1,idx)-Hk(1,:)\*Xk\_0\_hat );

Pk\_1= Pk\_0- Kk\_1\*Hk(1,:)\*Pk\_0;

%r=2

Kk\_2= Pk\_1 \* Hk(2,:)'\* inv(Hk(2,:) \* Pk\_1 \* Hk(2,:)'+ R(2,2)) ;

Xk\_2\_hat=Xk\_1\_hat+Kk\_2\*( Z(2,idx)-Hk(2,:)\*Xk\_1\_hat );

Pk\_2= Pk\_1- Kk\_2\*Hk(2,:)\*Pk\_1;

%r=3

Kk\_3= Pk\_2 \* Hk(3,:)'\* inv(Hk(3,:) \* Pk\_2 \* Hk(3,:)'+ R(3,3)) ;

Xk\_3\_hat=Xk\_2\_hat+Kk\_3\*( Z(3,idx)-Hk(3,:)\*Xk\_2\_hat );

Pk\_3= Pk\_2- Kk\_3\*Hk(3,:)\*Pk\_2;

%r=4

Kk\_4= Pk\_3 \* Hk(4,:)'\* inv(Hk(4,:) \* Pk\_3 \* Hk(4,:)'+ R(4,4)) ;

Xk\_4\_hat=Xk\_3\_hat+Kk\_4\*( Z(4,idx)-Hk(4,:)\*Xk\_3\_hat );

Pk\_4= Pk\_3- Kk\_4\*Hk(4,:)\*Pk\_3;

%r=5

Kk\_5= Pk\_4 \* Hk(5,:)'\* inv(Hk(5,:) \* Pk\_4 \* Hk(5,:)'+ R(5,5)) ;

Xk\_5\_hat=Xk\_4\_hat+Kk\_5\*( Z(5,idx)-Hk(5,:)\*Xk\_4\_hat );

Pk\_5= Pk\_4- Kk\_5\*Hk(5,:)\*Pk\_4;

Xk1k1\_hat=Xk\_5\_hat;

X\_hat(:,idx) = Xk1k1\_hat;

Pk1k1=Pk\_5;

end

生成的运动轨迹为：

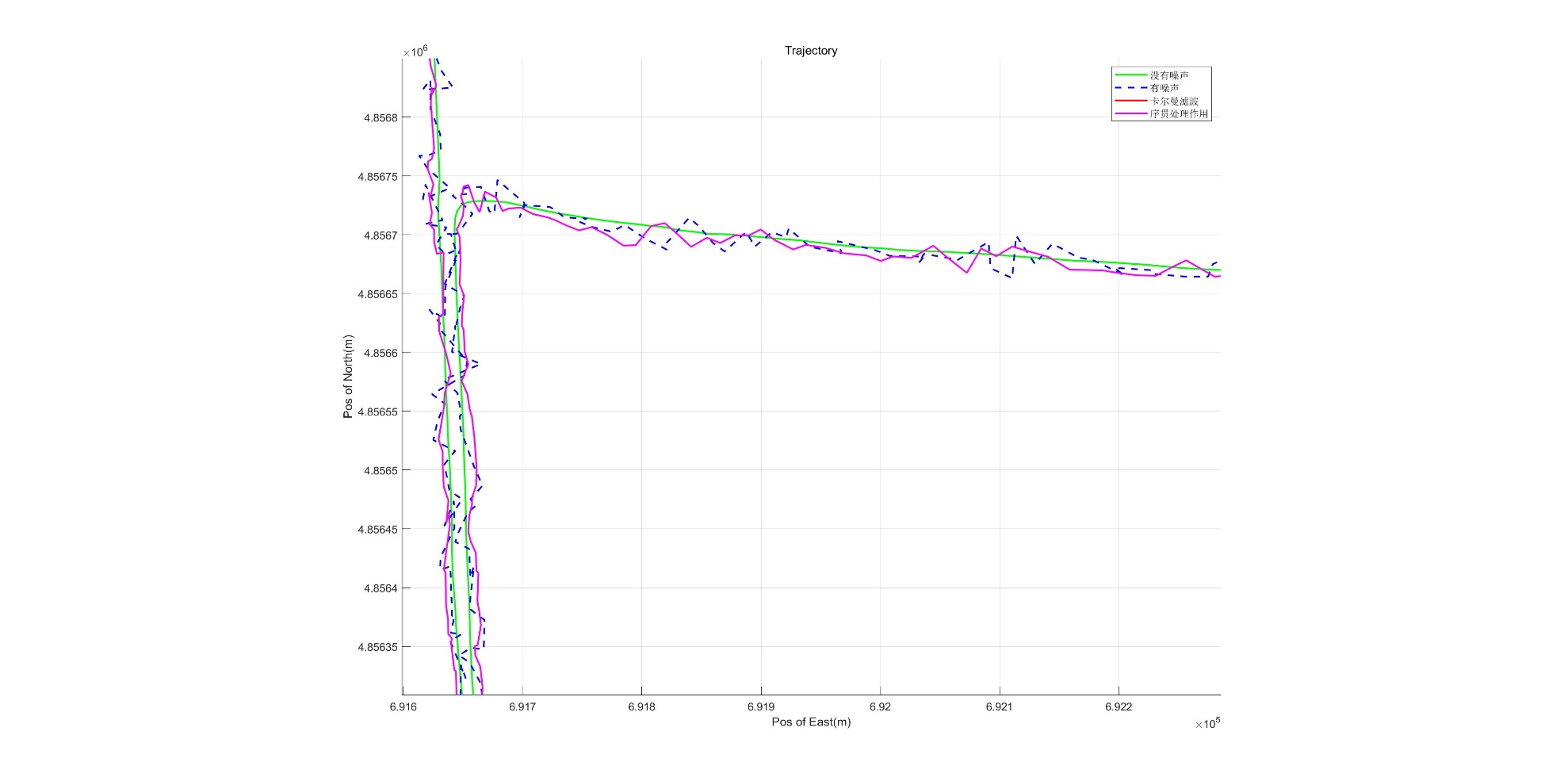


图9.1 组合导航运动轨迹局部放大

可以看到序贯处理后的滤波结果与常规Kalman滤波结果几乎一样。同时我们对两种估计方法作差分析，得到的不同状态估计方法结果的差值图像如下图所示：

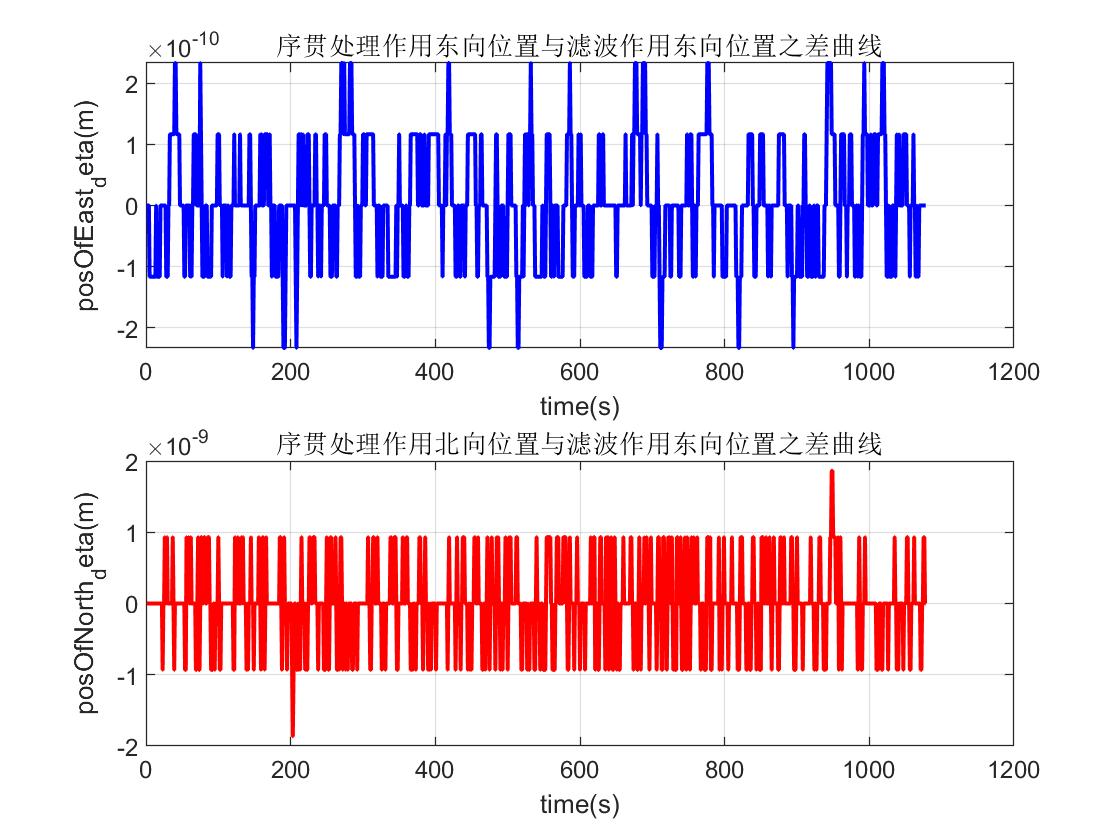


图9.2 两种方法位置估计结果之差

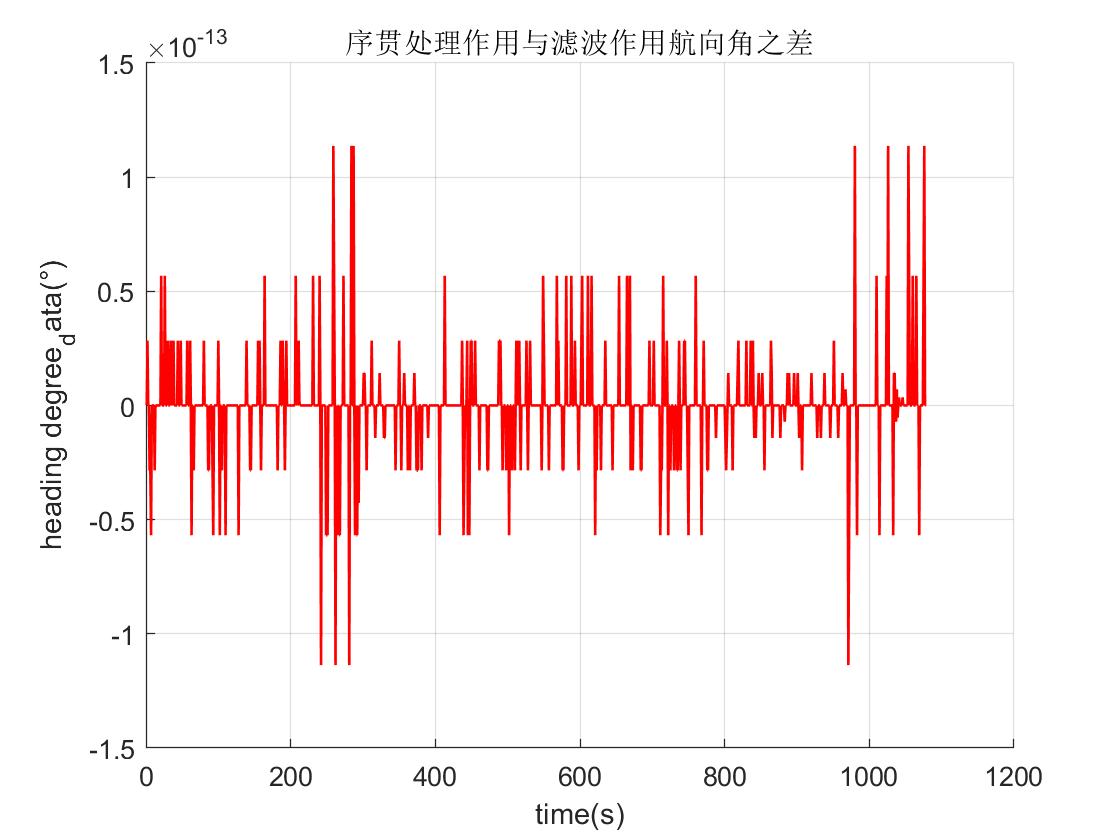


图9.3 两种方法航向角估计结果之差

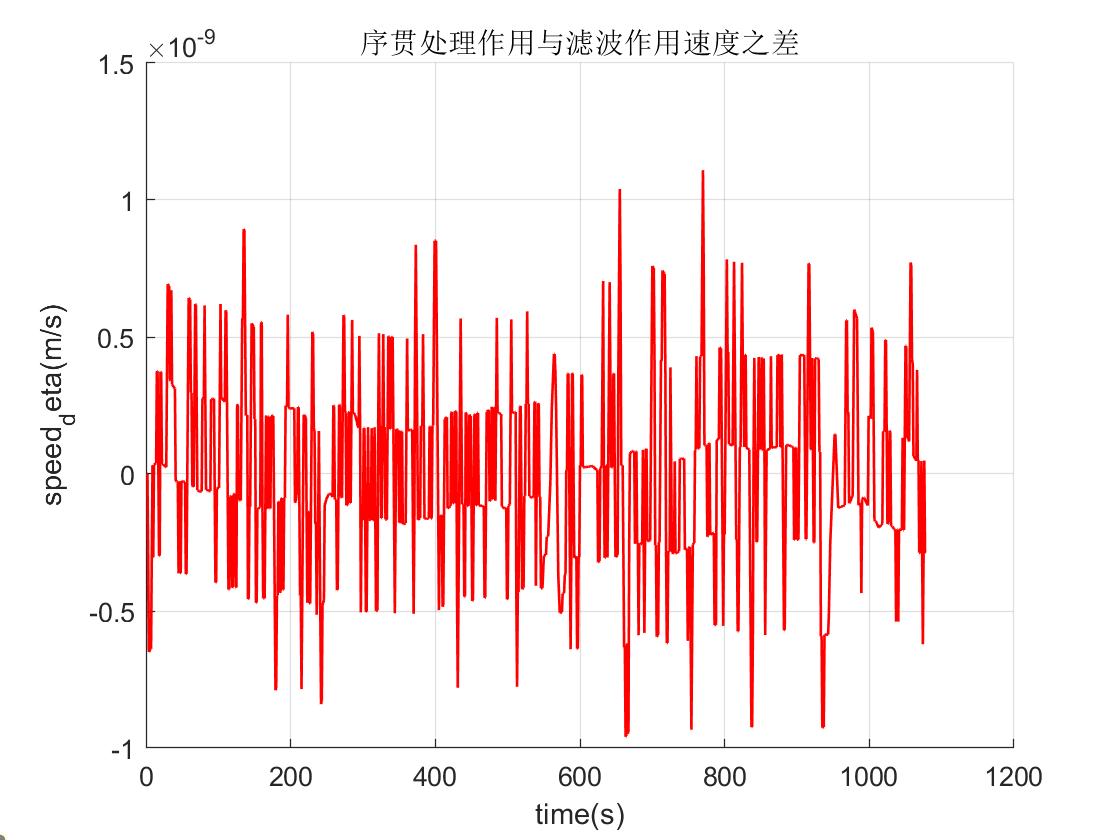


图9.4 两种方法速度估计结果之差

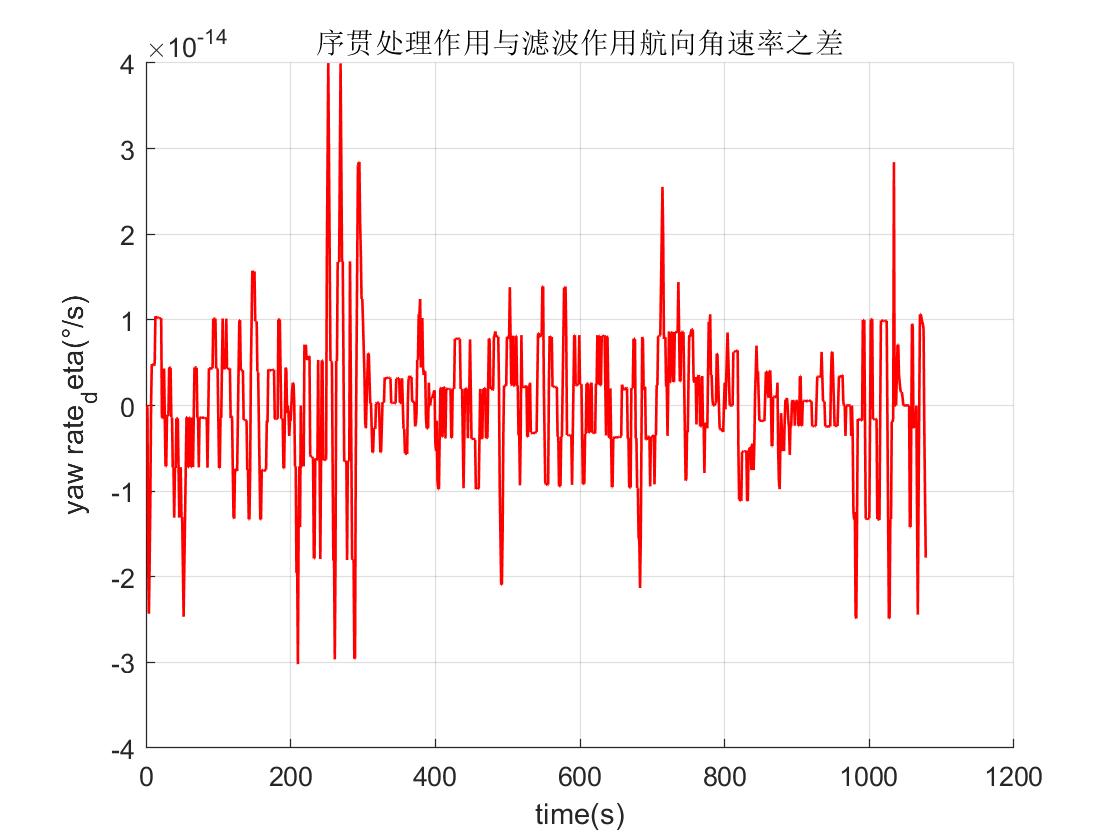


图9.5 两种方法陀螺仪角速率估计结果之差

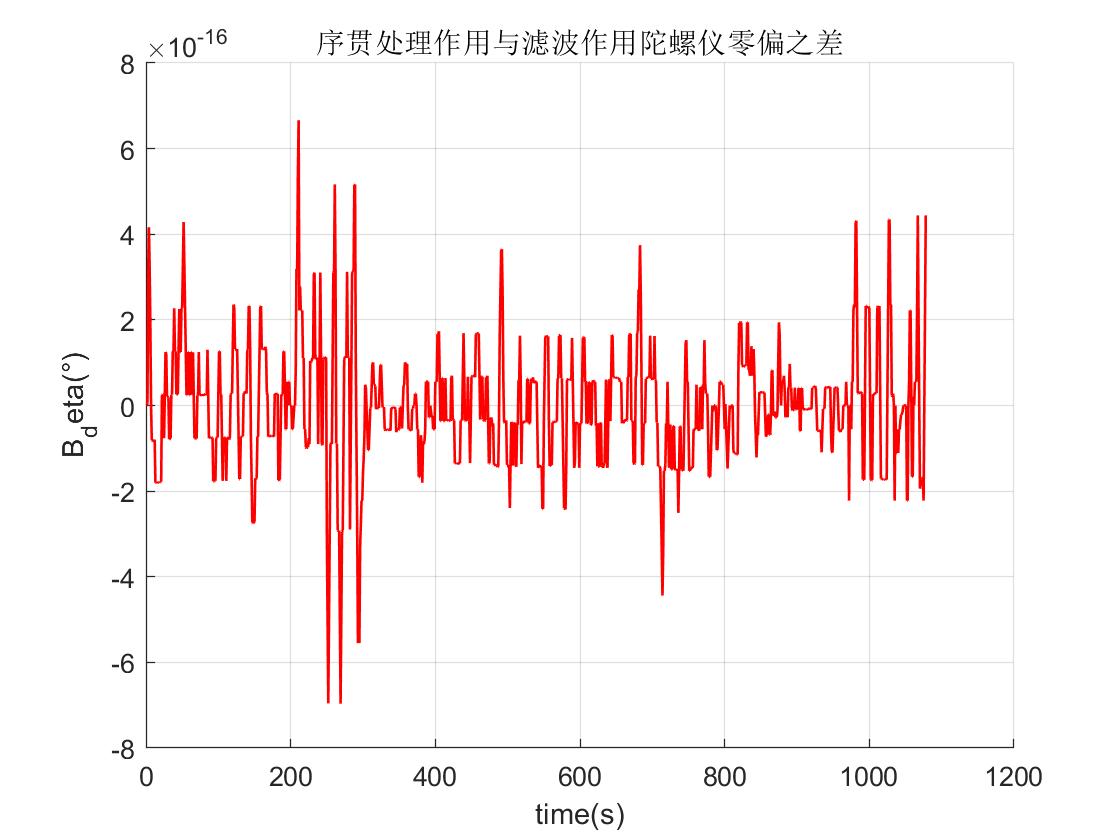


图9.6 两种方法陀螺仪零偏估计结果之差

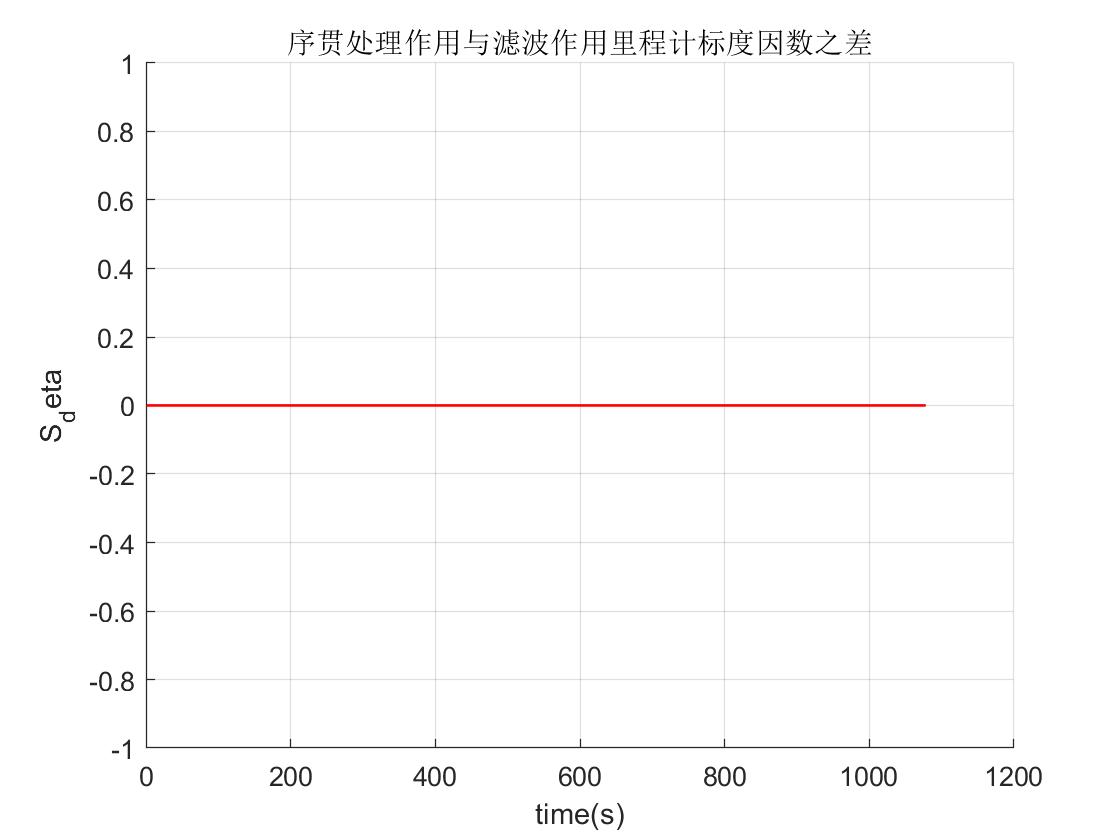


图9.6 两种方法里程计标度因数估计结果之差

从图中可以看出，序贯处理的状态估计结果与常规Kalman滤波结果差值为0，但序贯处理的状态估计稳定性较差，抖动比较大。

**十、在理想Q、R未知情况下有效估计Q、R**

问题13：如何在理想Q、R未知情况下有效估计Q、R？

Q、R的大小是对实际的模型误差和观测误差的估计，在Kalman滤波求解中对P、K的解算是不可或缺的部分，影响着状态估计的结果，因此Q、R选取的准确性对系统滤波的精度起着重要的作用，单纯的试凑法不能跟踪噪声的变化，会影响滤波的精度。

当Q和R的理想数值未知，可以采用以下几种方法进行对Q和R矩阵的有效估计：

方法一：采用遗传算法等智能算法求解合理的Q和R矩阵。评价函数取多次均方误差的均值，这样可以比较真实地反映实际情况。

方法二：采用自适应滤波方法，给定Q和R的初始值，利用观测数据进行修正和滤波的同时，对未知的或变化的系统模型或者噪声统计参数进行估计，以便动态调整Q、R到合适的R和Q取值使得滤波保持稳定。常见的方法在第五章节的内容里都可以找到。

这里我们采用Sage-Husa自适应滤波方法可以对对Q、R动态地进行估计。该算法的表达式如下：



称为遗忘因子，通常在0.95-0.99之间选取。并且虽然公式给出了多参数调节的方法，但不能对Q、R同时调节，否则会滤波发散，因此我们只对R进行调节。



该部分的代码可以写为：

b=0.97;

for idx = 2:length(posOfEast)

d\_k=(1-b)/(1-b^(idx));

Xk1k\_hat = phi(Xk1k1\_hat(3)) \* Xk1k1\_hat;

Pk1k = phi(Xk1k1\_hat(3)) \* Pk1k1 \* phi(Xk1k1\_hat(3))' +gamma() \* Q \* gamma()';

v\_k1(:,idx)=Z(:,idx) - h(Odometer(idx)) \* Xk1k\_hat;

Kk1 = Pk1k \* h(Odometer(idx))' \* inv(h(Odometer(idx)) \* Pk1k \* h(Odometer(idx))' + R);

Xk1k1\_hat = Xk1k\_hat + Kk1 \* (Z(:,idx) - h(Odometer(idx)) \* Xk1k\_hat);

X\_hat(:,idx) = Xk1k1\_hat;

Pk1k1 = Pk1k - Kk1 \* h(Odometer(idx)) \* Pk1k;

R =(1-d\_k)\*R +d\_k\*(v\_k1\*v\_k1'-h(Odometer(idx))\*Pk1k\* h(Odometer(idx))');

end

但在程序实际运行中一直出现发散现象，原因可能为系统阶次较高，不能保证完全可靠，有可能随Q、R失去非负定性而发散，因此应针对该问题进行算法的改进或者采用其他的自适应滤波方法。

此外需要额外注意的是，该数学模型中认为Q和R矩阵均为对角阵，实际中是不准确的，因为从模型分析可见，各个状态量之间存在一定的耦合，因此若想使该问题更符合实际情况，应应用矩阵理论的知识对Q和R矩阵进行三角分解。