后端1

10.1 概述

10.1.1 状态估计的概率解释

10.1.2 线性系统和卡尔曼滤波

最简单的线性高斯系统：运动方程① 与观测方程②

$$\begin{cases} x_k = A_k x_{k-1} + u_k + w_k \\ z_k = C_k x_k + v_k \end{cases}$$

其中，xk表示k时刻的状态，xk-1表示k-1时刻的状态，uk为k时刻的输入，wk为k时刻的噪声；zk为传感器k时刻的读数，vk为测量噪声。Ak和Ck为参数矩阵。在这之中，假设所有状态和噪声均满足高斯分布：

$$w_k 服从 N(0, R)$$
$$v_k 服从 N(0, Q)$$

我们先根据运动方程确定xk的先验分布（根据运动方程以及上一时刻的xk-1，我们可以直接根据公式得到xk，这里xk只在其运动方程的条件下得到的）

$$P(x_k|x_{k-1}) = N(A_k \hat{x_{k-1}} + u_k, A_k \hat{P_{k-1}} A_k^T + R)$$

这一步称为**预测**，显示了如何根据上一个时刻的状态，根据输入信息(附带噪声)，推断当前时刻的状态分布，得到先验。

接下来，根据观测方程，在机器人位置为xk(真实)时，得到的观测数据zk，那么，在xk条件下，zk的分布为：

$$P(z_k|x_k) = N(C_k x_k, Q)$$

下一步，我们需要得到后验概率，即在zk的情况下，xk的分布，根据贝叶斯公式：

$$N(\hat{x}_k, \hat{P}_k) = N(C_k x_k, Q) \times N(\overline{x}_k, \overline{P}_k)$$

为了求后验的均值与协方差，我们直接考虑高斯分布的指数部分的计算：

$$(x_k - \hat{x}_k)^T \hat{P}_k^{-1} (x_k - \hat{x}_k) = (z_k - C_k x_k)^T Q^{-1} (z_k - C_k x_k) + (x_k - \overline{x}_k)^T \overline{P}_k^{-1} (x_k - \overline{x}_k)$$

展开，并比较xk的二次项系数和一次项系数：

①对于二次项系数：

$$\hat{P}_k^{-1} = C_k^T Q^{-1} C_k + \overline{P}_k^{-1}$$

上式左右乘Phatk，得到：

$$I = \hat{P}_k C_k^T Q^{-1} C_k + \hat{P}_k \overline{P}_k^{-1}$$

我们记：

$$K = \hat{P}_k C_k^T Q^{-1}$$

②对于一次项系数：

$$-2\hat{\boldsymbol{x}}_k^T\hat{\boldsymbol{P}}_k^{-1}\boldsymbol{x}_k=-2\boldsymbol{z}_k^T\boldsymbol{Q}^{-1}\boldsymbol{C}_k\boldsymbol{x}_k-2\bar{\boldsymbol{x}}_k^T\bar{\boldsymbol{P}}_k^{-1}\boldsymbol{x}_k.$$

取xk的系数并且取转置后，整理得到：

$$\hat{\boldsymbol{P}}_k^{-1}\hat{\boldsymbol{x}}_k=\boldsymbol{C}_k^T\boldsymbol{Q}^{-1}\boldsymbol{z}_k+\bar{\boldsymbol{P}}_k^{-1}\bar{\boldsymbol{x}}_k.$$

最终，代入K，得到：

$$\begin{aligned}\hat{\boldsymbol{x}}_k&=\hat{\boldsymbol{P}}_k\boldsymbol{C}_k^T\boldsymbol{Q}^{-1}\boldsymbol{z}_k+\hat{\boldsymbol{P}}_k\bar{\boldsymbol{P}}_k^{-1}\bar{\boldsymbol{x}}_k\\&=\boldsymbol{K}\boldsymbol{z}_k+(\boldsymbol{I}-\boldsymbol{K}\boldsymbol{C}_k)\,\bar{\boldsymbol{x}}_k=\bar{\boldsymbol{x}}_k+\boldsymbol{K}\left(\boldsymbol{z}_k-\boldsymbol{C}_k\bar{\boldsymbol{x}}_k\right).\end{aligned}$$

这样，我们最终得到线性卡尔曼滤波的步骤，分为**预测和更新两个部分**

1. 预测：

$$\bar{\boldsymbol{x}}_k=\boldsymbol{A}_k\hat{\boldsymbol{x}}_{k-1}+\boldsymbol{u}_k,\quad \bar{\boldsymbol{P}}_k=\boldsymbol{A}_k\hat{\boldsymbol{P}}_{k-1}\boldsymbol{A}_k^T+\boldsymbol{R}. \tag{10.24}$$

2. 更新：先计算 $\boldsymbol{K}$，它又称为卡尔曼增益：

$$\boldsymbol{K}=\bar{\boldsymbol{P}}_k\boldsymbol{C}_k^T\left(\boldsymbol{C}_k\bar{\boldsymbol{P}}_k\boldsymbol{C}_k^T+\boldsymbol{Q}\right)^{-1}. \tag{10.25}$$

然后计算后验概率的分布：

$$\begin{aligned}\hat{\boldsymbol{x}}_k&=\bar{\boldsymbol{x}}_k+\boldsymbol{K}\left(\boldsymbol{z}_k-\boldsymbol{C}_k\bar{\boldsymbol{x}}_k\right)\\\hat{\boldsymbol{P}}_k&=\left(\boldsymbol{I}-\boldsymbol{K}\boldsymbol{C}_k\right)\bar{\boldsymbol{P}}_k.\end{aligned} \tag{10.26}$$

至此，卡尔曼滤波结束。

matlab实践代码：

```
clear;
clc;
close;

%test kalmen filter
%linear system:
%xk=Ak xk-1+ uk +wk
%zk= Ck xk + vk

N=100
x0=0 %initial value
uk=sin(0.01:0.01:(N)*0.01)-sin(0:0.01:(N-1)*0.01)
uk=uk*1000

uk=zeros(1,N);
```

```
Ak=[1]
Ck=[1]

KalemFilter(Ak,Ck,x0,N,uk)


function KalemFilter(Ak,Ck,x0,N,uk)
    %x1=Ak*x0+uk(1)+randn(1)
    X_true=zeros(1,N);
    X_noise=zeros(1,N);
    X_KF=zeros(1,N);
    Z_Meas=zeros(1,N);
    wk=sqrt(0.01).*randn(1,N);
    vk=sqrt(0.25).*randn(1,N);

    xlk=x0;
    xlk_hat=x0;
    Plk_hat=0.01;
    for i=1:N
        %predict
        xk_line=Ak*xlk_hat+uk(i)+wk(i);
        Pk_line=Ak*Plk_hat*Ak'+0.01;

        xk=Ak*xlk+uk(i);
        zk=Ck*xk+vk(i);
        %clac
        K=Pk_line*Ck'/(Ck*Pk_line*Ck'+0.25);

        %update
        xk_hat=xk_line+K*(zk-Ck*xk_line);
        Pk_hat=(eye(1)-K*Ck)*Pk_line;

        X_true(i)=xk;
        X_noise(i)=xk_line;
        X_KF(i)=xk_hat;
        Z_Meas(i)=zk;

        Plk_hat=Pk_hat;
        xlk_hat=xk_hat;
    end
    plot(X_true)
    hold on;
     plot(X_noise)
     hold on;
     plot(X_KF)
     hold on;
    plot(Z_Meas)
    hold off;

    legend("true","noise","KF","meas")
end
```

10.1.3 非线性系统和扩展卡尔曼滤波(EKF)

在SLAM中的运动方程和观测方程通常是非线性函数，尤其是视觉SLAM中的相机模型，需要使用相机内参模型以及李代数表示的位姿。

10.2 BA与图优化

所谓的BA(Bundle Adjustment)，是指从视觉重建中提炼出的最优3D模型和相机参数(内参与外参)。

从每一个特征点反射出来的几束光线(bundles of light rays)，在我们把相机姿态和特征点空间位置做出最优的调整(adjustment)之后，最后收束到相机光心的这个过程，简称为BA

10.2.1 投影模型和BA代价函数

在此，复习投影模型和畸变，从一个世界坐标系中的点p出发，考虑内外参数和畸变，最后投影成像素坐标：

①把世界坐标转换到相机坐标，使用相机外参数(R,t)：

$$P' = Rp + t = [X', Y', Z']^T$$

②将P'投至归一化平面，得到归一化坐标：

$$Pc = [u_c, v_c, 1]^T = [X'/Z', Y'/Z', 1]^T$$

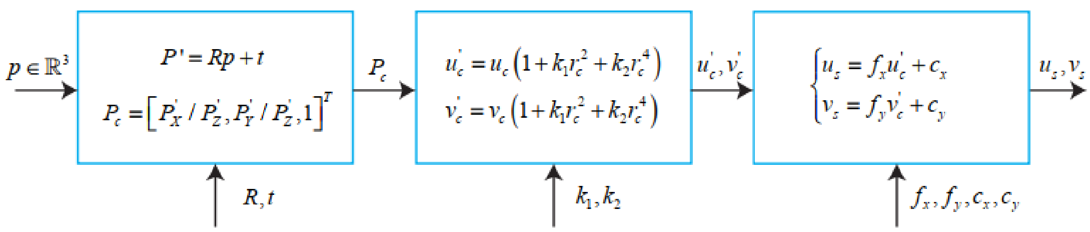
③对归一化坐标去畸变，得到去畸变后的坐标（只考虑径向畸变）：

$$\begin{cases} u_c' = u_c(1 + k_1 r_c^2 + k_2 r_c^4) \\ v_c' = v_c(1 + k_1 r_c^2 + k_2 r_c^4) \end{cases}$$

④最后，根据内参模型，计算像素坐标：

$$\begin{cases} u_s = f_x u_c' + c_x \\ v_s = f_y v_c' + c_y \end{cases}$$

整体流程如下图：



这个过程，就是前面提到的**观测方程**，在之前我们抽象地将其记为：z=h(x,y)。现在我们给出详细的参数化过程：**这里的x指代此时相机的位姿(R,t)，对应李代数为ξ。路标y即这里的三维点p，而观测数据则是像素坐标z=[us,vs]^T。**

从最小二乘角度：观测误差：

$$e = z - h(\xi, p)$$

我们设zij是在位姿ξi处观察路标pj产生的数据，那么整体的**代价函数(Cost Function)**为：

$$\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}||e_{ij}||^2 = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}||z_{ij} - h(\xi_i, p_j)||^2$$

对这个最小二乘进行求解，相当于对位姿和路标同时作了调整，也就是所谓的BA。

10.2.2 BA的求解

根据非线性优化思想，我们应该从某个初始值开始，不断地寻找下降方向Δx来找到目标函数的最优解。

我们给自变量一个小增量，可以得到：

$$\frac{1}{2}\left\|f(\boldsymbol{x}+\Delta\boldsymbol{x})\right\|^2 \approx \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\left\|\boldsymbol{e}_{ij}+\boldsymbol{F}_{ij}\Delta\boldsymbol{\xi}_i+\boldsymbol{E}_{ij}\Delta\boldsymbol{p}_j\right\|^2.$$

其中，Fij表示代价函数当前状态下对相机位姿的偏导数，Eij表示代价函数当前状态下对路标点位置的偏导数。

我们把相机位姿变量和空间点变量分别放在一起：

$$x_c=[\xi_1,\xi_2,\ldots,\xi_m]^T\in R^{6m}$$
$$x_p=[p_1,p_2,\ldots,p_n]^T\in R^{3n}$$

那么，上式可以简化为：

$$\frac{1}{2}||f(x+\Delta x)||^2=\frac{1}{2}||e+F\Delta x_c+E\Delta x_p||^2$$

这里，使用G-N或L-M法都包括增量方程：

$$H\Delta x=g$$

我们把雅可比矩阵分块为：

$$J=[F\ \ E]$$

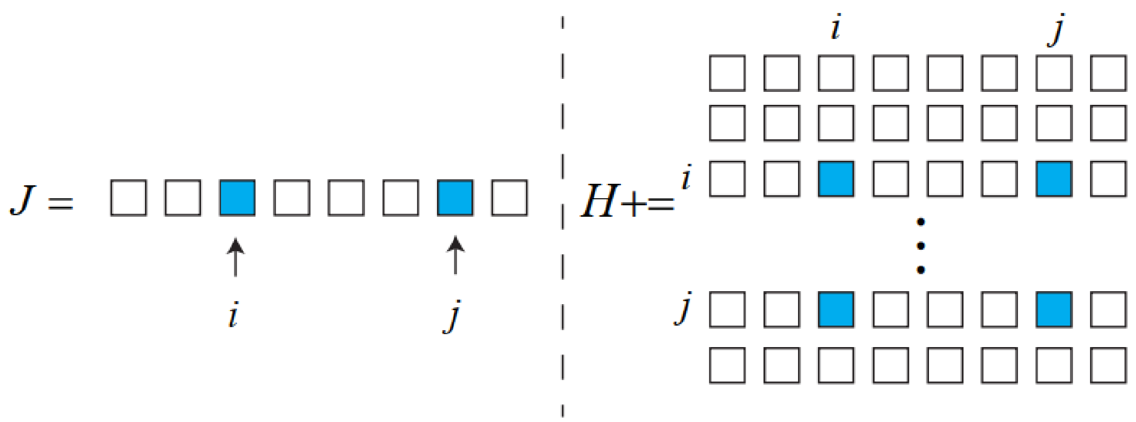
那么这个H矩阵规模会很大，但其拥有特殊结构，利用其特殊结构，我们可以加速求解过程。

10.2.3 稀疏性和边缘化

我们单独考虑Jij(x)，代表只涉及第i个相机位姿和第j个路标点，因此有

$$\boldsymbol{J}_{ij}(\boldsymbol{x})=\left(\boldsymbol{0}_{2\times6},\ldots\boldsymbol{0}_{2\times6},\frac{\partial\boldsymbol{e}_{ij}}{\partial\boldsymbol{\xi}_i},\boldsymbol{0}_{2\times6},\ldots\boldsymbol{0}_{2\times3},\ldots\boldsymbol{0}_{2\times3},\frac{\partial\boldsymbol{e}_{ij}}{\partial\boldsymbol{p}_j},\boldsymbol{0}_{2\times3},\ldots\boldsymbol{0}_{2\times3}\right).$$

这种形式的雅可比矩阵，计算J^T J时候会出现什么样的结果？如下图所示：



那么我们计算整体的H，使用如下公式：

$$H=\sum_{i,j} J_{ij}^T J_{ij}$$

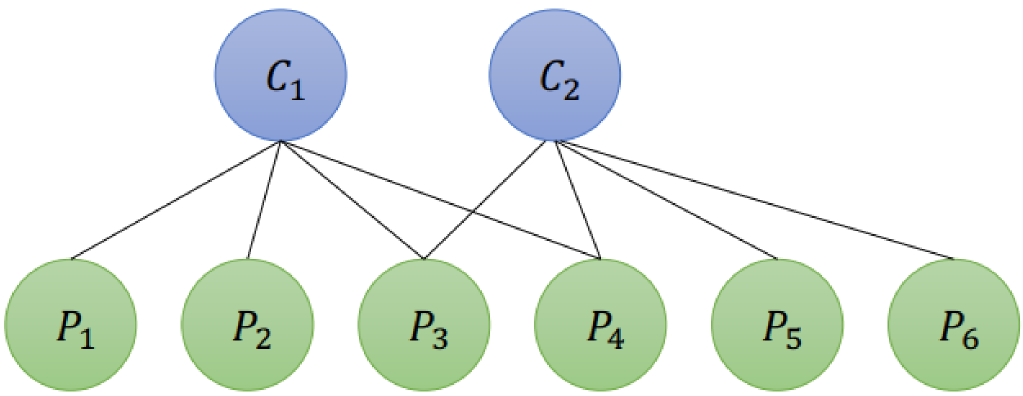
i在所有相机位姿中取值，j在所有路标点中取值，并且把H进行分块

$$H=\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}$$

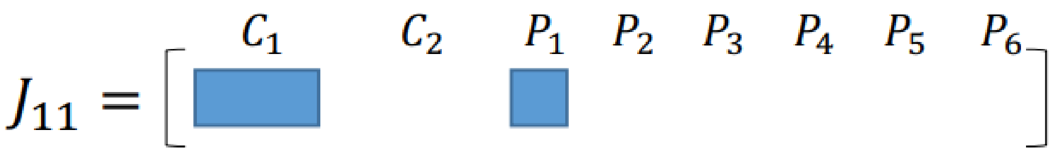
这里，H11和H22都是分块对角矩阵，分别只与相机位姿和路标点有关，这个H有三大特点：

1. 不管i，j如何变化，H11都是分块对角矩阵，只在Hii处有非零块
2. 同理，H22也是对角阵，只在Hjj处有非零块
3. 对于H12和H21，它们可能是稀疏的，也可能是稠密的，根据具体的观测数据而定。

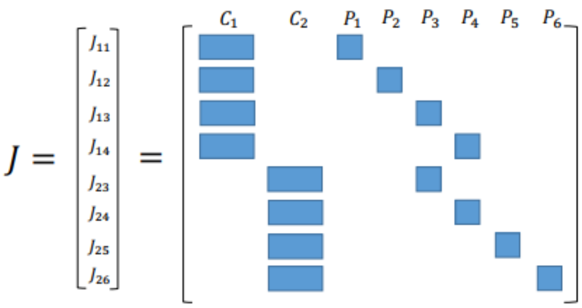
H的稀疏结构，对之后的线性方程求解中很重要。现在举一个例子，如下图：C1，C2为相机位姿，P1~P6路标。

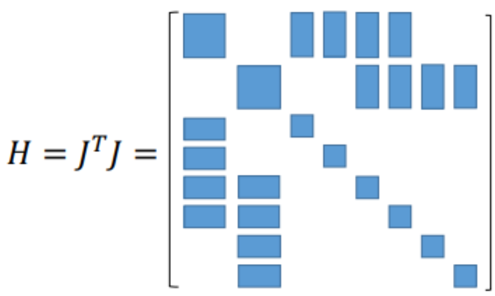


在位姿C1处观测到了P1~P4路标，在位姿C2处观测到了P3~P6路标。由上述的稀疏性，易得J11的形式：



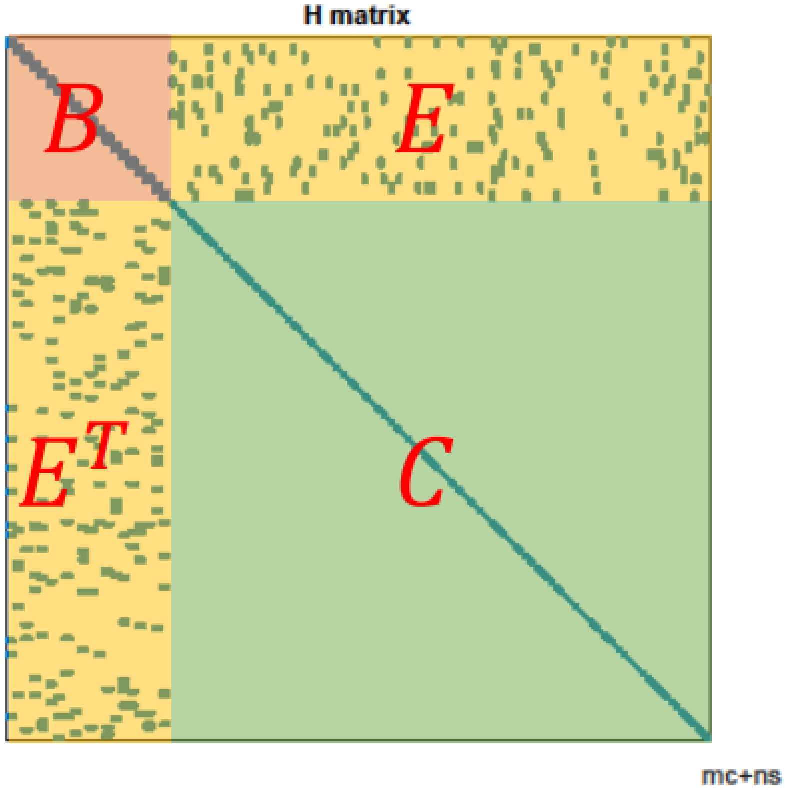
类推，J的形式和H的形式：





求解方法

对于这种稀疏结构的H，线性方程HΔx=g的求解会变得简单，我们将其分块：如下图所示



于是对应的线性方程组变为：

$$\begin{bmatrix} B & E \\ E^T & C \end{bmatrix} \begin{bmatrix} \Delta x_c \\ \Delta x_p \end{bmatrix} = \begin{bmatrix} v \\ w \end{bmatrix}$$

其中B和C为分块对角矩阵。考虑到，求解分块对角矩阵的逆就是求解对角线上每个矩阵的逆。因此，我们目标是消去非对角部分E：

$$\begin{bmatrix} I & -EC^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} B & E \\ E^T & C \end{bmatrix} \begin{bmatrix} \Delta x_c \\ \Delta x_p \end{bmatrix} = \begin{bmatrix} I & -EC^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix}$$

整理得到：

$$\begin{bmatrix} B - EC^{-1}E^T & 0 \\ E^T & C \end{bmatrix} \begin{bmatrix} \Delta x_c \\ \Delta x_p \end{bmatrix} = \begin{bmatrix} v - EC^{-1}w \\ w \end{bmatrix}$$

第一行方程组变为和Δxp无关的项，把它解出来，再代入原方程解出Δxp即可。

10.2.4 鲁棒核函数

当出现误匹配情况时，调整这个情况是没有任何意义的，并且会严重影响正确的优化，因此我们提出鲁棒核函数：**当误差很大时，其增长速度没有那么快**。例如常用的Huber核：

$$H(e) = \begin{cases} \frac{1}{2}e^2 & , \ if \ |e| \leq \delta \\ \delta(|e| - \frac{1}{2}\delta) & , \ otherwise \end{cases}$$

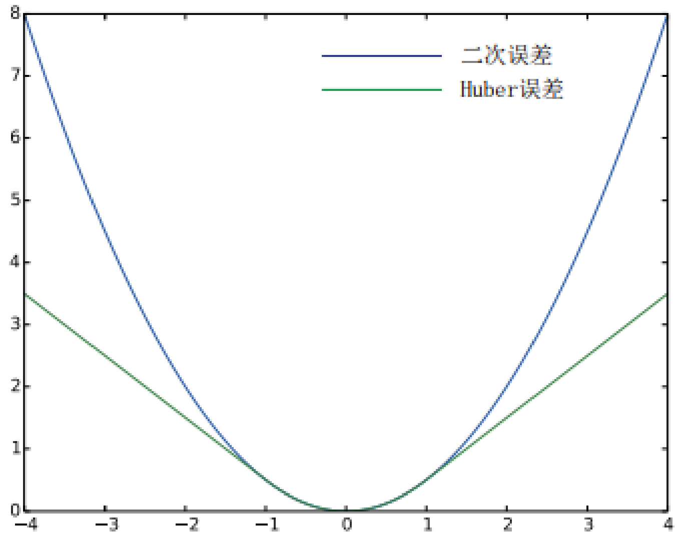


图 10-12　Huber 核函数。

除了Huber核之外，还有Cauchy核、Tukey核等等。