SLAM14讲—10后端1

10.1概述

10.1.1状态估计的概率解释

在后端优化中，我们通常考虑一个更长的时间（或所有时间内）的状态估计问题，而且不仅使用过去的信息更新自己的状态，也会用未来的信息来更新自己，这种处理方式称为**"批量的"（Batch）。如果当前的状态只由过去的时刻决定，甚至只由前一个时刻决定，称为"渐进的"（Incremental）**。

我们已经知道 SLAM 过程可以由运动方程和观测方程来描述。那么，假设在 $t=0$ 到 $t=N$ 的时间内，我们有 $\boldsymbol{x}_0$ 到 $\boldsymbol{x}_N$ 那么多个位姿，并且有 $\boldsymbol{y}_1,\ldots,\boldsymbol{y}_M$ 那么多个路标。按照之前的写法，运动和观测方程为：

$$\begin{cases} \boldsymbol{x}_k = f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k) + \boldsymbol{w}_k \\ \boldsymbol{z}_{k,j} = h(\boldsymbol{y}_j, \boldsymbol{x}_k) + \boldsymbol{v}_{k,j} \end{cases} \quad k=1,\ldots,N,\ j=1,\ldots,M. \tag{10.1}$$

注意：

1. 观测方程中，只由当$\mathbf{x}_k$看到了$\mathbf{y}_j$时，才会产生观测数据。一个位置一般只能看到小部分路标；且SLAM特征点数量众多，故实际中观测方程数量会远远大于运动方程的数量。
2. 运动方程可以没有。如此，有若干种处理方式：①认为确实没有②假设相机不动③假设相机匀速运动。在没有运动方程的情况下，整个优化问题就只有许多个观测方程组成。类似于SfM(Structure from Motion)问题，相当于通过一组图像来恢复运动和结构。SLAM中的图像有时间上的先后顺序，而SfM中允许使用完全无关的图像。

因为每个方程都受噪声影响，所以这里的位姿**x**和路标**y**看成服从某种概率分布的随机变量。假设状态量和噪声项服从高斯分布，即在程序中只需要储存它们的均值和协方差矩阵即可。

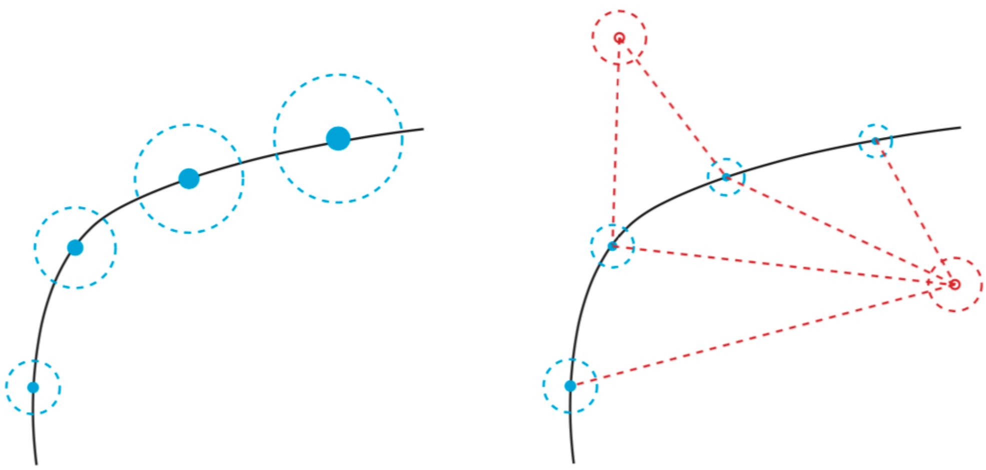


图 10-1　不确定性的直观描述。左侧：只有运动方程时，由于下一个时刻的位姿是在上一个时刻基础上添加了噪声，所以不确定性越来越大。右侧：存在路标点（红色）时，不确定性会明显减小。不过请注意这只是一个直观的示意图，并非实际数据。

- 令$\mathbf{x}_k$为k时刻的所有未知量。它包含了当前时刻的相机位姿与m个路标点。在这种记号的意义下，写成：

$$\boldsymbol{x}_k \stackrel{\Delta}{=} \{\boldsymbol{x}_k, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_m\}. \tag{10.2}$$

同时，把k时刻的所有观测记作$\mathbf{z}_k$。于是运动方程和观测方程可以写为：

$$\begin{cases} \boldsymbol{x}_k = f\left(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k\right) + \boldsymbol{w}_k \\ \boldsymbol{z}_k = h\left(\boldsymbol{x}_k\right) + \boldsymbol{v}_k \end{cases} \quad k = 1, \ldots, N. \tag{10.3}$$

- 现在考虑第k时刻的情况。我们希望用0到k中的数据，来估计现在的状态分布：

$$P(\boldsymbol{x}_k | \boldsymbol{x}_0, \boldsymbol{u}_{1:k}, \boldsymbol{z}_{1:k}). \tag{10.4}$$

按照Bayes法则有：

$$P\left(\boldsymbol{x}_k | \boldsymbol{x}_0, \boldsymbol{u}_{1:k}, \boldsymbol{z}_{1:k}\right) \propto P\left(\boldsymbol{z}_k | \boldsymbol{x}_k\right) P\left(\boldsymbol{x}_k | \boldsymbol{x}_0, \boldsymbol{u}_{1:k}, \boldsymbol{z}_{1:k-1}\right). \tag{10.5}$$

然后按照$\mathbf{x}_{k-1}$时刻为条件概率展开：

$$P\left(\boldsymbol{x}_k | \boldsymbol{x}_0, \boldsymbol{u}_{1:k}, \boldsymbol{z}_{1:k-1}\right) = \int P\left(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}, \boldsymbol{x}_0, \boldsymbol{u}_{1:k}, \boldsymbol{z}_{1:k-1}\right) P\left(\boldsymbol{x}_{k-1} | \boldsymbol{x}_0, \boldsymbol{u}_{1:k}, \boldsymbol{z}_{1:k-1}\right) \mathrm{d}\boldsymbol{x}_{k-1}. \tag{10.6}$$

如何处理上述有以下两种方法：

1. 假设马尔科夫性，据此可以得到以**扩展卡尔曼滤波（EKF）**为代表的滤波器方法。
2. 考虑k时刻状态与之前所有状态的关系，此时讲得到**非线性优化**为主体的优化框架。

10.1.2线性系统和KF

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：

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

10.1.3非线性系统和EKF

我们希望把卡尔曼滤波器的结果拓展到非线性系统中来，称为扩展卡尔曼滤波器（Extended Kalman Filter，EKF）。通常的做法是，在某个点附近考虑运动方程以及观测方程的一阶泰勒展开，只保留一阶项，即线性的部分，然后按照线性系统进行推导。令 $k-1$ 时刻的均值与协方差矩阵为 $\hat{\boldsymbol{x}}_{k-1}, \hat{\boldsymbol{P}}_{k-1}$。在 k 时刻，我们把运动方程和观测方程，在 $\hat{\boldsymbol{x}}_{k-1}, \hat{\boldsymbol{P}}_{k-1}$ 处进行**线性化**（相当于一阶泰勒展开），有：

$$\boldsymbol{x}_k \approx f\left(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k\right) + \left.\frac{\partial f}{\partial \boldsymbol{x}_{k-1}}\right|_{\hat{\boldsymbol{x}}_{k-1}} \left(\boldsymbol{x}_{k-1} - \hat{\boldsymbol{x}}_{k-1}\right) + \boldsymbol{w}_k. \tag{10.27}$$

记这里的偏导数为：

$$\boldsymbol{F} = \left.\frac{\partial f}{\partial \boldsymbol{x}_{k-1}}\right|_{\hat{\boldsymbol{x}}_{k-1}}. \tag{10.28}$$

同样的，对于观测方程，亦有：

$$\boldsymbol{z}_k \approx h\left(\bar{\boldsymbol{x}}_k\right) + \left.\frac{\partial h}{\partial \boldsymbol{x}_k}\right|_{\bar{\boldsymbol{x}}_k} \left(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k\right) + \boldsymbol{n}_k. \tag{10.29}$$

记这里的偏导数为：

$$\boldsymbol{H} = \left.\frac{\partial h}{\partial \boldsymbol{x}_k}\right|_{\bar{\boldsymbol{x}}_k}. \tag{10.30}$$

那么，在**预测**步骤中，根据运动方程有：

$$P\left(\boldsymbol{x}_k | \boldsymbol{x}_0, \boldsymbol{u}_{1:k}, \boldsymbol{z}_{0:k-1}\right) = N(f\left(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k\right), \boldsymbol{F}\hat{\boldsymbol{P}}_{k-1}\boldsymbol{F}^{\mathrm{T}} + \boldsymbol{R}_k). \tag{10.31}$$

这些推导和卡尔曼滤波是十分相似的。为方便表述，记这里先验和协方差的均值为

$$\bar{\boldsymbol{x}}_k = f\left(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k\right), \quad \bar{\boldsymbol{P}}_k = \boldsymbol{F}\hat{\boldsymbol{P}}_k\boldsymbol{F}^T + \boldsymbol{R}_k. \tag{10.32}$$

然后，考虑在观测中，我们有：

$$P\left(\boldsymbol{z}_k|\boldsymbol{x}_k\right)=N(h\left(\bar{\boldsymbol{x}}_k\right)+\boldsymbol{H}\left(\boldsymbol{x}_k-\bar{\boldsymbol{x}}_k\right),\boldsymbol{Q}_k). \tag{10.33}$$

最后，根据最开始的 Bayes 展开式，可以推导出 $\boldsymbol{x}_k$ 的后验概率形式。我们略去中间的推导过程，只介绍其结果。读者可以仿照着卡尔曼滤波器的方式，推导 EKF 的预测与更新方程。简而言之，我们会先定义一个**卡尔曼增益** $\boldsymbol{K}_k$：

$$\boldsymbol{K}_k=\bar{\boldsymbol{P}}_k\boldsymbol{H}^{\mathrm{T}}\left(\boldsymbol{H}\bar{\boldsymbol{P}}_k\boldsymbol{H}^{\mathrm{T}}+\boldsymbol{Q}_k\right)^{-1}. \tag{10.34}$$

在卡尔曼增益的基础上，后验概率的形式为：

$$\hat{\boldsymbol{x}}_k=\bar{\boldsymbol{x}}_k+\boldsymbol{K}_k\left(\boldsymbol{z}_k-h\left(\bar{\boldsymbol{x}}_k\right)\right),\hat{\boldsymbol{P}}_k=\left(\boldsymbol{I}-\boldsymbol{K}_k\boldsymbol{H}\right)\bar{\boldsymbol{P}}_k. \tag{10.35}$$

卡尔曼滤波器给出了在线性化之后，状态变量分布的变化过程。在线性系统和高斯噪声下，卡尔曼滤波器给出了无偏最优估计。而在 SLAM 这种非线性的情况下，它给出了单次线性近似下最大后验估计（MAP）。

10.1.4EKF的讨论

EKF在计算资源受限，或待估计量比较简单的场合比较有效，但局限有：

1. 一定程度上假设了马尔科夫性。如果当前帧确实与很久之前的数据有关（如回环），那么滤波器很难处理这种情况。而非线性优化则倾向于使用所有的历史数据，当然也有更多的计算。
2. 存在非线性误差。假设某点处的线性化近似，在后验概率处仍然是有效的，但是实际上离开工作点较远的时候，一节泰勒展开并不一定能近似整个函数，这取决于运动模型和观测模型的非线性情况。
3. EKF的存储量很大，且与状态量呈平方增长，故不适用于大兴场景。

10.2BA与图优化

10.2.1投影模型和BA代价函数

- 从一个世界坐标系中的点**p**出发，把相机的内外参数和畸变都考虑进来，最后投影成像素坐标，一共需要以下几个步骤：

1. 首先，把世界坐标转换到相机坐标，这里将用到相机外参数 $(\boldsymbol{R}, \boldsymbol{t})$：

$$\boldsymbol{P}' = \boldsymbol{R}\boldsymbol{p} + \boldsymbol{t} = [X', Y', Z']^T. \tag{10.36}$$

2. 然后，将 $\boldsymbol{P}'$ 投至归一化平面，得到归一化坐标：

$$\boldsymbol{P}_c = [u_c, v_c, 1]^T = [X'/Z', Y'/Z', 1]^T. \tag{10.37}$$

3. 对归一化坐标去畸变，得到去畸变后的坐标。这里暂时只考虑径向畸变：

$$\begin{cases} u_c' = u_c\left(1 + k_1 r_c^2 + k_2 r_c^4\right) \\ v_c' = v_c\left(1 + k_1 r_c^2 + k_2 r_c^4\right) \end{cases}. \tag{10.38}$$

4. 最后，根据内参模型，计算像素坐标：

$$\begin{cases} u_s = f_x u_c' + c_x \\ v_s = f_y v_c' + c_y \end{cases}. \tag{10.39}$$

流程图示意如下：

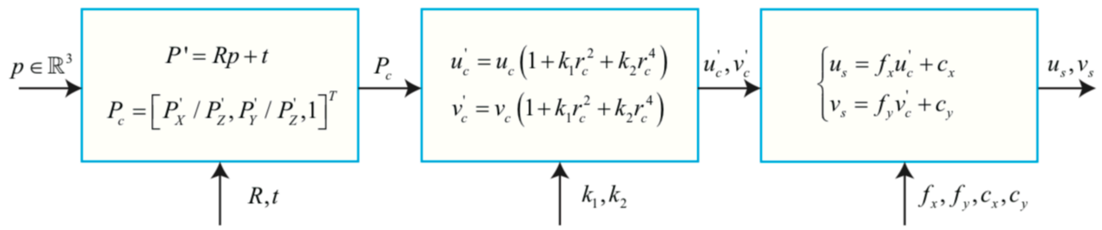


图 10-2　计算流程示意图。左侧的 $\boldsymbol{p}$ 是全局坐标系下的三维坐标点，右侧的 u_s, v_s 是该点在图像平面上的最终像素坐标。中间畸变模块中的 $r_c^2 = u_c^2 + v_c^2$

之前我们抽象地记成**z**=h(**x**,**y**)。现在我们给出详细的参数化过程，**x**指此时相机的位姿，即外参**R**，**t**，它对应的李代数为**ξ**。路标**y**即这里的三维点**p**，而观测数据则是像素坐标**z**=$[u_S, v_S]^T$。以最小二乘的角度考虑，可以写出关于此次观测误差：

$$\boldsymbol{e} = \boldsymbol{z} - h(\boldsymbol{\xi}, \boldsymbol{p}). \tag{10.41}$$

那么整体的代价函数为：

$$\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\|\boldsymbol{e}_{ij}\|^2 = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\|\boldsymbol{z}_{ij} - h(\boldsymbol{\xi}_i, \boldsymbol{p}_j)\|^2. \tag{10.42}$$

我们求解这个最小二乘，相当于对位姿和路标做了调整，就是所谓的BA。

10.2.2BA的求解

在整体BA目标函数上，我们把自变量定义成所有待优化的变量：

$$\boldsymbol{x} = [\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_m, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_n]^T. \tag{10.43}$$

当我们给自变量一个增量时，目标函数变为：

$$\frac{1}{2}\left\|f(\boldsymbol{x}+\Delta\boldsymbol{x})\right\|^2 \approx \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\left\|\boldsymbol{e}_{ij}+\boldsymbol{F}_{ij}\Delta\boldsymbol{\xi}_i+\boldsymbol{E}_{ij}\Delta\boldsymbol{p}_j\right\|^2. \qquad (10.44)$$

其中$\mathbf{F}_{i,j}$表示整个代价函数在当前状态下对相机姿态的偏导数，$\mathbf{E}_{i,j}$表示该函数对路标点位置的偏导。

它们的推导了。现在，把相机位姿变量放在一起：

$$\boldsymbol{x}_c=[\boldsymbol{\xi}_1,\boldsymbol{\xi}_2,\ldots,\boldsymbol{\xi}_m]^T\in\mathbb{R}^{6m}, \qquad (10.45)$$

并把空间点的变量也放在一起：

$$\boldsymbol{x}_p=[\boldsymbol{p}_1,\boldsymbol{p}_2,\ldots,\boldsymbol{p}_n]^T\in\mathbb{R}^{3n}, \qquad (10.46)$$

那么，式（10.44）可以简化表达为如下：

$$\frac{1}{2}\left\|f(\boldsymbol{x}+\Delta\boldsymbol{x})\right\|^2=\frac{1}{2}\left\|\boldsymbol{e}+\boldsymbol{F}\Delta\boldsymbol{x}_c+\boldsymbol{E}\Delta\boldsymbol{x}_p\right\|^2. \qquad (10.47)$$

我们可以利用G-N或L-M方法进行求解，差别在于**H**的形式是$\mathbf{J}^\mathbf{T}\mathbf{J}$还是$\mathbf{J}^\mathbf{T}\mathbf{J}+\lambda\mathbf{I}$。因为把变量归类成了位姿和空间点两种，故雅可比矩阵可以分块为：

$$\boldsymbol{J}=[\boldsymbol{F}\ \ \boldsymbol{E}]. \qquad (10.49)$$

那么，以 G-N 为例，则 $\boldsymbol{H}$ 矩阵为：

$$\boldsymbol{H}=\boldsymbol{J}^T\boldsymbol{J}=\begin{bmatrix}\boldsymbol{F}^T\boldsymbol{F} & \boldsymbol{F}^T\boldsymbol{E}\\ \boldsymbol{E}^T\boldsymbol{F} & \boldsymbol{E}^T\boldsymbol{E}\end{bmatrix}. \qquad (10.50)$$

10.2.3稀疏性和边缘化

H矩阵的稀疏性是由雅可比**J(x)**引起的。考虑这些代价函数当中的其中一个$\mathbf{e}_{\mathbf{ij}}$。这个误差项只描述了在$\boldsymbol{\xi}_i$看到$\mathbf{p}_j$这件事，只涉及到第i个相机位姿和第j个路标点，对其余部分的变量的导数都为0，故该误差项对应的雅可比矩阵有下面的形式：

$$\boldsymbol{J}_{ij}(\boldsymbol{x})=\left(\mathbf{0}_{2\times6},...\mathbf{0}_{2\times6},\frac{\partial\boldsymbol{e}_{ij}}{\partial\boldsymbol{\xi}_i},\mathbf{0}_{2\times6},...\mathbf{0}_{2\times3},...\mathbf{0}_{2\times3},\frac{\partial\boldsymbol{e}_{ij}}{\partial\boldsymbol{p}_j},\mathbf{0}_{2\times3},...\mathbf{0}_{2\times3}\right). \qquad (10.51)$$

其中$\mathbf{0}_{2x6}$表示维度为2x6的**0**矩阵，该误差项对相机姿态的偏导维度为2x6，对路标点的偏导为2x3。

- 某个误差项**J**有稀疏性时，对**H**的贡献也有稀疏形式

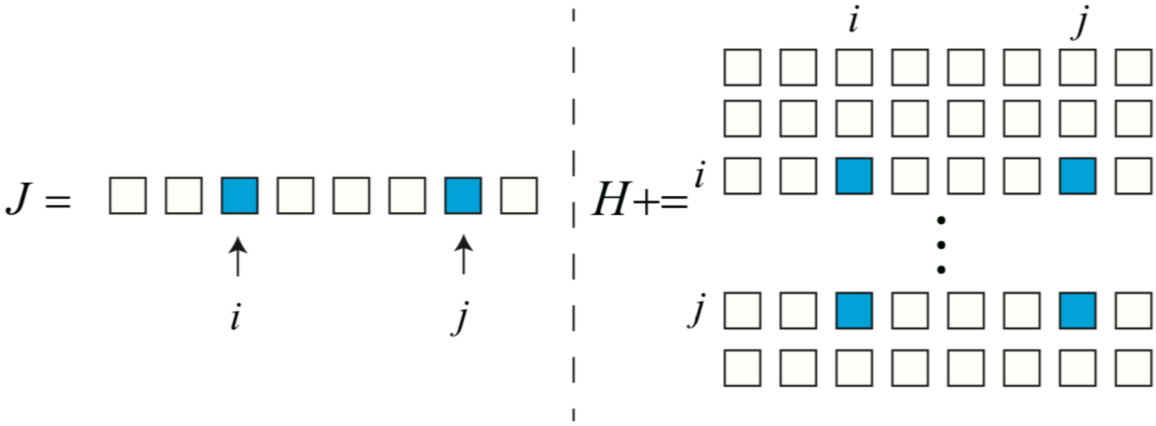


图 10-3 当某个误差项 $\boldsymbol{J}$ 具有稀疏性时，它对 $\boldsymbol{H}$ 的贡献也具有稀疏形式。

设$\mathbf{J}_{\mathrm{ij}}$只在i,j处有非零块，那么它对$\mathbf{H}$的贡献为$\mathbf{J}^{\mathrm{T}}{}_{\mathrm{ij}}\mathbf{J}_{\mathrm{ij}}$，具有示意图上所画的稀疏形式，即位于(i,i),(i,j),(j,i),(j,j)有非零块。对于整体的$\mathbf{H}$，由于：

$$\boldsymbol{H} = \sum_{i,j} \boldsymbol{J}_{ij}^T \boldsymbol{J}_{ij}, \tag{10.52}$$

我们可以对$\mathbf{H}$进行分块：

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{H}_{11} & \boldsymbol{H}_{12} \\ \boldsymbol{H}_{21} & \boldsymbol{H}_{22} \end{bmatrix}. \tag{10.53}$$

这里 $\boldsymbol{H}_{11}$ 只和相机位姿有关，而 $\boldsymbol{H}_{22}$ 只和路标点有关。当我们遍历 i,j 时，以下事实总是成立的：

1. 不管 i,j 怎么变，$\boldsymbol{H}_{11}$ 都是对角阵，只在 $\boldsymbol{H}_{i,i}$ 处有非零块；
2. 同理，$\boldsymbol{H}_{22}$ 也是对角阵，只在 $\boldsymbol{H}_{j,j}$ 处有非零块；
3. 对于 $\boldsymbol{H}_{12}$ 和 $\boldsymbol{H}_{21}$，它们可能是稀疏的，也可能是稠密的，视具体的观测数据而定。

- 举例：假设一个场景内有2个相机位姿(C1,C2)和6个路标(P1,P2,P3,P4,P5,P6)，这些相机和点云所对应的变量为$\boldsymbol{\xi}$i，i=1,2以及$\mathbf{p}$j，j=1，2...6。相机C1观测到路标P1,P2,P3,P4,相机C2观测到路标P3,P4,P5,P6。相机和路标用圆形节点表示，若能观测到则连上一条边：

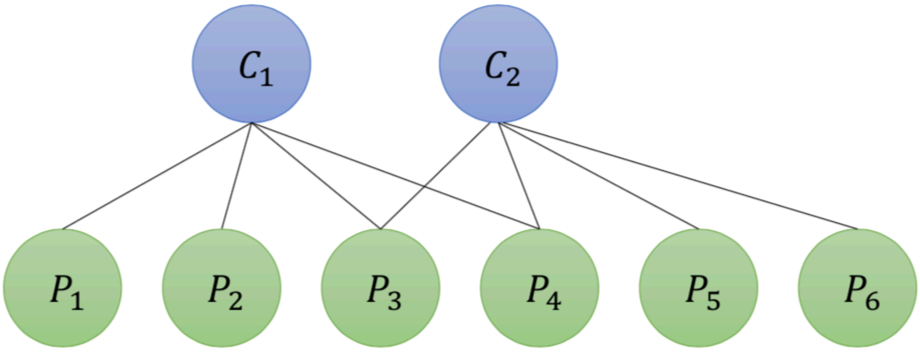


图 10-4　点和边组成的示意图。该图显示相机 C_1 观测到了路标点 P_1, P_2, P_3, P_4，相机 C_2 看到了 P_3 到 P_6。

可以推测出该场景下的BA目标函数应该是:

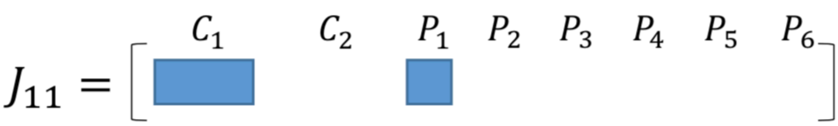
$$\frac{1}{2}\left(\|\boldsymbol{e}_{11}\|^2 + \|\boldsymbol{e}_{12}\|^2 + \|\boldsymbol{e}_{13}\|^2 + \|\boldsymbol{e}_{14}\|^2 + \|\boldsymbol{e}_{23}\|^2 + \|\boldsymbol{e}_{24}\|^2 + \|\boldsymbol{e}_{25}\|^2 + \|\boldsymbol{e}_{26}\|^2\right). \quad (10.54)$$

这里的 $\boldsymbol{e}_{ij}$ 使用之前定义过的代价函数，即式（10.42）。以 $\boldsymbol{e}_{11}$ 为例，它描述了在 C_1 看到了 P_1 这件事，与其他的相机位姿和路标无关。令 $\boldsymbol{J}_{11}$ 为 $\boldsymbol{e}_{11}$ 所对应的雅可比矩

阵，不难看出 $\boldsymbol{e}_{11}$ 对相机变量 $\boldsymbol{\xi}_2$ 和路标点 $\boldsymbol{p}_2, \ldots, \boldsymbol{p}_6$ 的偏导都为 0。我们把所有变量以 $\boldsymbol{x} = (\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_2)^T$ 的顺序摆放，则有：

$$\boldsymbol{J}_{11} = \frac{\partial \boldsymbol{e}_{11}}{\partial \boldsymbol{x}} = \left(\frac{\partial \boldsymbol{e}_{11}}{\partial \boldsymbol{\xi}_1}, \boldsymbol{0}_{2\times 6}, \frac{\partial \boldsymbol{e}_{11}}{\partial \boldsymbol{p}_1}, \boldsymbol{0}_{2\times 3}, \boldsymbol{0}_{2\times 3}, \boldsymbol{0}_{2\times 3}, \boldsymbol{0}_{2\times 3}, \boldsymbol{0}_{2\times 3}\right). \quad (10.55)$$

为了方便表示稀疏性，我们用带有颜色的方块表示矩阵在该方块内有数值，其余没有颜色的区域表示矩阵在该处数值都为 0。那么上面的 $\boldsymbol{J}_{11}$ 则可以表示成图 10-5 的图案。同理，其他的雅可比矩阵也会有类似的稀疏图案。



为了得到该目标函数对应的雅可比矩阵，我们可以将这些 $\boldsymbol{J}_{ij}$ 按照一定顺序列为向量，那么整体雅可比矩阵以及相应的 $\boldsymbol{H}$ 矩阵的稀疏情况就是图 10-6 中所展示的那样。

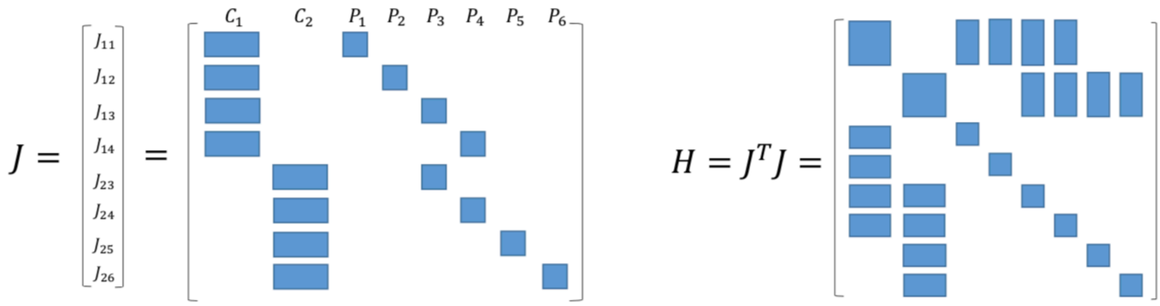


图 10-6　Jacobian 矩阵的稀疏性 (左) 和 $\boldsymbol{H}$ 矩阵的稀疏性 (右)，蓝色的方块表示矩阵在对应的矩阵块处有数值，其余没有颜色的部分表示矩阵在该处的数值始终为 0。

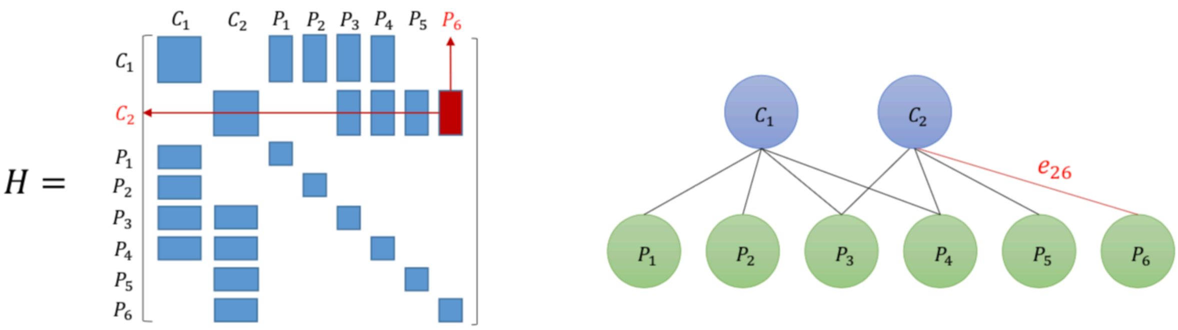


图 10-7　$\boldsymbol{H}$ 矩阵中非零矩阵块和图中边的对应关系。如左图当中的 $\boldsymbol{H}$ 矩阵当中红色的矩阵块，表示在右图中其对应的变量 C_2 和 P_6 之间存在一条边 $\boldsymbol{e}_{26}$ 。

- 一般情况下的**H**矩阵。此时假设m个相机位姿，n个路标点。n>>m。

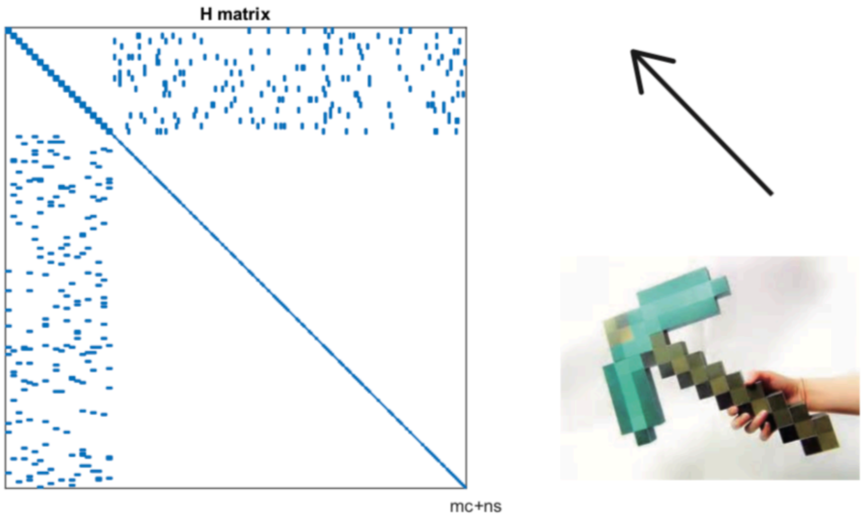


图 10-8　一般情况下的 $\boldsymbol{H}$ 矩阵。

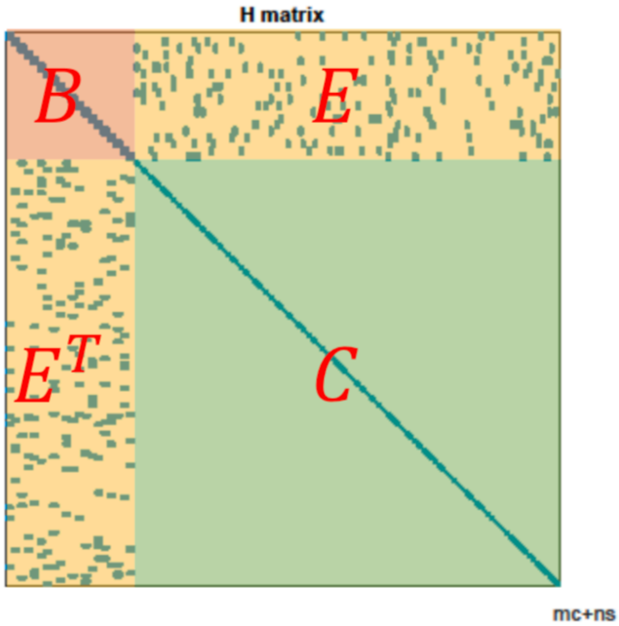


图 10-9　$\boldsymbol{H}$ 矩阵的区域划分。

- 利用**H**的稀疏性加速计算的方法：Schur消元/边缘化

  我们可以把**H**划分成四个矩阵块**B**，**E**，**C**，左上角为对角矩阵块矩阵，每个对角块元素的维度与相机位姿的维度相等，且是一个对角块矩阵。右下角也是对角块矩阵，每个对角块的维度是路标的维度。

于是，对应的线性方程组也可以由 $\boldsymbol{H}\boldsymbol{\Delta x} = \boldsymbol{g}$ 变为如下形式：

$$\begin{bmatrix} \boldsymbol{B} & \boldsymbol{E} \\ \boldsymbol{E}^T & \boldsymbol{C} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{x}_c \\ \Delta \boldsymbol{x}_p \end{bmatrix} = \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{w} \end{bmatrix}. \tag{10.56}$$

其中 $\boldsymbol{B}$ 是对角块矩阵，每个对角块的维度和相机参数的维度相同，对角块的个数是相机变量的个数。由于路标数量会远远大于相机变量个数，所以 $\boldsymbol{C}$ 往往也远大于 $\boldsymbol{B}$。三维空间中每个路标点为三维，于是 $\boldsymbol{C}$ 矩阵为对角块矩阵，每个块为 3×3 维矩阵。对角块

我们进行高斯消元，消去右上角的非对角部分**E**，整理可得到：

$$
\begin{bmatrix} \boldsymbol{B} - \boldsymbol{E}\boldsymbol{C}^{-1}\boldsymbol{E}^T & \boldsymbol{0} \\ \boldsymbol{E}^T & \boldsymbol{C} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{x}_c \\ \Delta \boldsymbol{x}_p \end{bmatrix} = \begin{bmatrix} \boldsymbol{v} - \boldsymbol{E}\boldsymbol{C}^{-1}\boldsymbol{w} \\ \boldsymbol{w} \end{bmatrix}. \tag{10.58}
$$

其中第一行方程组即:

$$
\left[\boldsymbol{B} - \boldsymbol{E}\boldsymbol{C}^{-1}\boldsymbol{E}^T\right] \Delta \boldsymbol{x}_c = \boldsymbol{v} - \boldsymbol{E}\boldsymbol{C}^{-1}\boldsymbol{w}. \tag{10.59}
$$

我们先求解这个方程，然后把解得的**$\triangle x_c$代入原方程，然后求解$\triangle x_p$**即可。

- 进行了Schur消元后**S**的稀疏性的物理意义
  S矩阵的非对角线上的非零矩阵块，表示了该处对应的两个相机变量之间存在着的共同观测的路标点，有时候称为共视。反之，若该块为0，则表示两个相机没有共同观测。

图 10-10 对 $\boldsymbol{H}$ 矩阵进行 Schur 消元后 $\boldsymbol{S}$ 矩阵的稀疏状态。

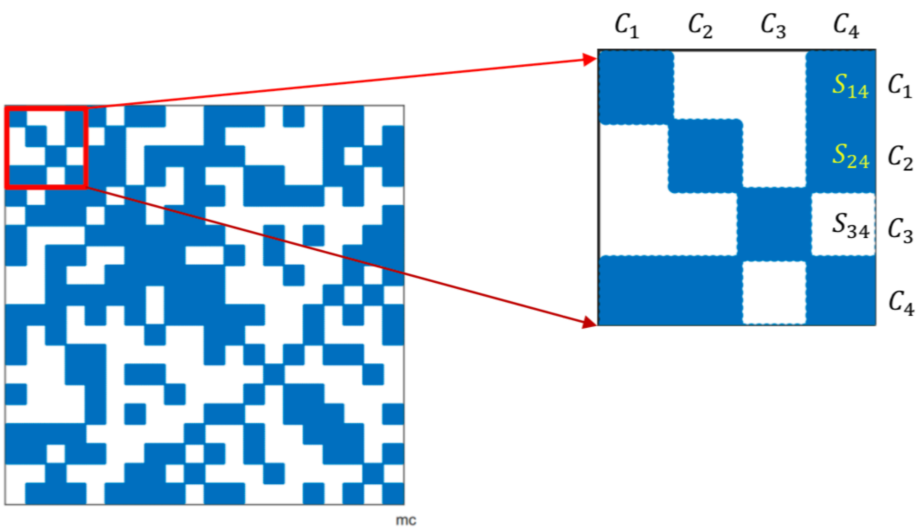


图 10-11 以 $\boldsymbol{S}$ 矩阵中前 4×4 个矩阵块为例，这区域当中的矩阵块 $\boldsymbol{S}_{14}, \boldsymbol{S}_{24}$ 不为零，表示相机 C_4 和相机 C_1 和 C_2 之间有共同观测的点。而 S_{34} 为零则表示 C_3 和 C_4 之间没有共同观测的路标。

10.2.4鲁棒核函数

前面的BA中，我们最小化误差项的二范数平方和作为目标函数。若出于误匹配等原因，某个误差项给的数据是错误的，把一条原本不应该加到图中的边给加进去了，然而优化算法不能辨别出这是一个错误数据。这时算法会看到一条误差很大的边，梯度也很大，意味着调整与它相关的变量会使目标函数下降很多。**出现这个问题的原因时，当误差很大时，二范数增长得太快了。**故需要核函数，来保证每条边的误差不会太大，掩盖掉其他的边。
具体做法为：把原先误差的二范数度量，替换成一个增长没有那么快的函数，同时保证光滑性质。

鲁棒核函数有许多种，例如最常用的 Huber 核：

$$H(e)=\begin{cases}\frac{1}{2}e^2 & \text{if } |e|\le\delta,\\ \delta\left(|e|-\frac{1}{2}\delta\right) & \text{otherwise.}\end{cases} \tag{10.61}$$

我们看到，当误差 e 大于某个阈值 δ 后，函数增长由二次形式变成了一次形式，相当于限制了梯度的最大值。同时，Huber 核函数又是光滑的，可以很方便地求导。图 10-12 显示了 Huber 核函数与二次函数的对比，可见在误差较大时 Huber 核函数增长明显低于二次函数。

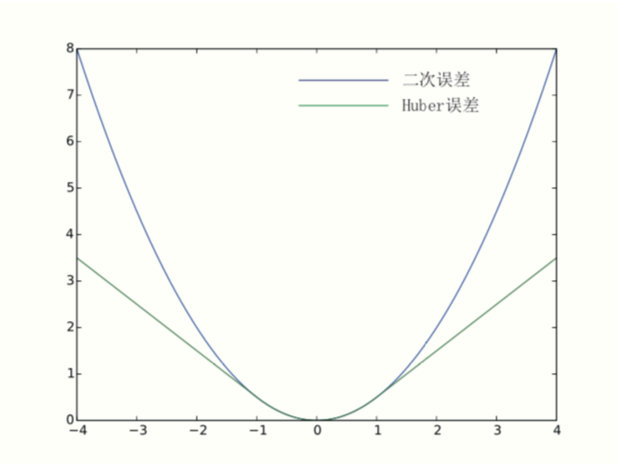


图 10-12　Huber 核函数。

除了 Huber 核之外，还有 Cauchy 核，Tukey 核等等，读者可以看看 g2o 和 Ceres 都提供了哪些核函数。

10.3实践：g2o

g2o_bundle.cpp代码修改
先注释掉标注没用的部分，然后对186行代码进行修改

```
//原代码
solver_ptr = new BalBlockSolver(linearSolver);
//修改为
solver_ptr = new BalBlockSolver(std::unique_ptr<BalBlockSolver>(linearSolver));
```

192及195行代码修改如下

```
//原代码
solver = new g2o::OptimizationAlgorithomLevenberg(solver_ptr);
//修改为
solver = new g2o::OptimizationAlgorithomLevenberg(std::unique_ptr<BalBlockSolver>
```

```
(solver_ptr));
```

然后编译后运行代码如下

```
./g2o_customBundle -input ../data/problem-16-22106-pre.txt
```

附：meshlab的安装方法

```
sudo add-apt-repository ppa:zarquon42/meshlab
sudo apt-get update
sudo apt-get install meshlab
```

10.4实践：Ceres

make时报错找不到Eigen/core，在CMakeLists文件中添加如下命令

```
include dictionaries("/usr/include/eigen3")
```

ceresBundle中17行代码需要修改：

```
//原代码
options->num_linear_solver_threads = params.num_threads;
//修改代码
options->num_threads = params.num_threads;
```

- Solver::Options::num_threads
  Default: 1
  Ceres求Jacobain的线程数目。
- Solver::Options::num_linear_solver_threads
  Default: 1
  线性解算使用的线程数。