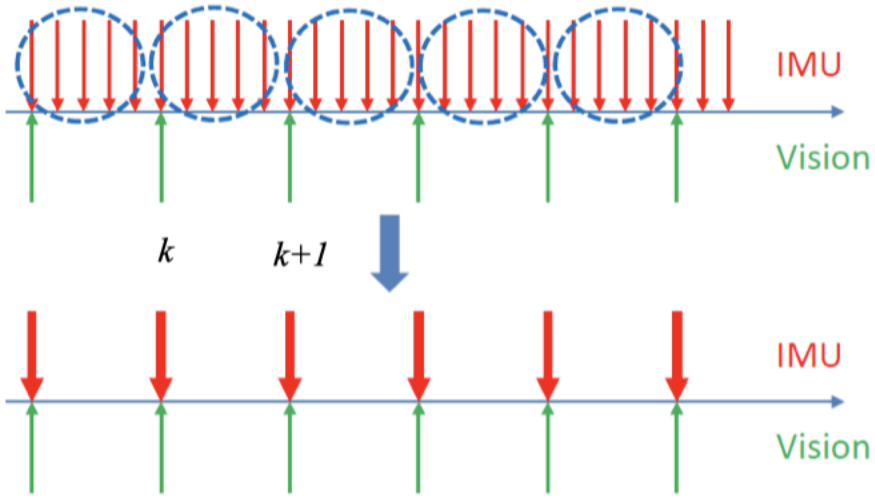
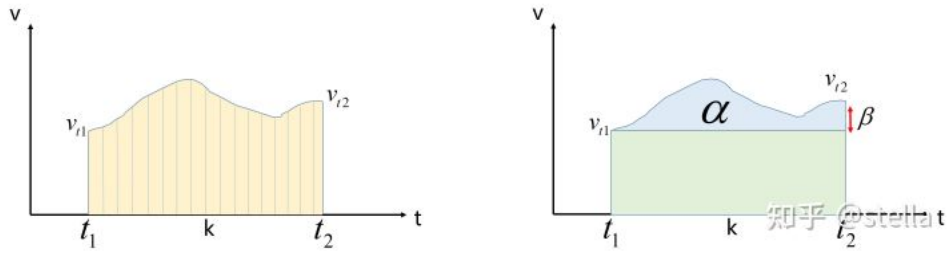
LVI-SAM 学习小组分享



为什么要做IMU预积分？

在构建优化问题时，会将相机或激光关键帧的pvq（位置、速度、旋转）以及imu bias作为状态量进行优化。在求解优化问题时，会不断迭代更新这些状态量。所以，我们在求解优化问题的过程中**每迭代一次，更新了一下关键帧的位姿、速度和IMU bias，就需要重复一次积分操作**，要知道我们在优化的时候不止迭代一次的，这样就会花费大量的时间重新积分，显然是不太合适的。

IMU预积分的思路就是先把每次优化迭代时不变的项提出来，减小每次重新积分的工作量



IMU预积分公式

基础知识：

- BCH 近似

$$
\begin{aligned}
\ln(\mathbf{C}_1\mathbf{C}_2)^\vee &= \ln(\exp(\phi_1^\wedge)\exp(\phi_2^\wedge))^\vee \\
&\approx \begin{cases} \mathbf{J}_l(\phi_2)^{-1}\phi_1 + \phi_2 & \text{若}\phi_1\text{很小} \\ \phi_1 + \mathbf{J}_r(\phi_1)^{-1}\phi_2 & \text{若}\phi_2\text{很小} \end{cases}
\end{aligned}
$$

- 伴随性质

$$\mathbf{R}^T \exp\left(\boldsymbol{\phi}^\wedge\right)\mathbf{R} = \exp\left(\left(\mathbf{R}^T\boldsymbol{\phi}\right)^\wedge\right)$$

1. 预积分的定义

一个IMU系统考虑五个变量：旋转R, 平移 p , 角速度ω , 线速度 v 与 加速度 a

由旋转运动学可得：

$$\dot{\boldsymbol{R}} = \boldsymbol{R}\boldsymbol{\omega}^\wedge$$
$$\dot{\boldsymbol{p}} = \boldsymbol{v}$$
$$\dot{\boldsymbol{v}} = \boldsymbol{a}$$

从 t 到 $t+\Delta t$ 的时间里，对上式进行欧拉积分可以得到：

$$\boldsymbol{R}(t+\Delta t) = \boldsymbol{R}(t)\,\mathrm{Exp}(\boldsymbol{\omega}(t)\Delta t)$$
$$\boldsymbol{v}(t+\Delta t) = \boldsymbol{v}(t) + \boldsymbol{a}(t)\Delta t$$
$$\boldsymbol{p}(t+\Delta t) = \boldsymbol{p}(t) + \boldsymbol{v}(t)\Delta t + \frac{1}{2}\boldsymbol{a}(t)\Delta t^2$$

其中角速度和加速度是IMU可以测量的量，但受到噪声与重力影响。令测量值为 $\tilde{\boldsymbol{\omega}}, \tilde{\boldsymbol{a}}$

$$\tilde{\boldsymbol{\omega}}(t) = \boldsymbol{\omega}(t) + \boldsymbol{b}_g(t) + \boldsymbol{\eta}_g(t)$$
$$\tilde{\boldsymbol{a}}(t) = \boldsymbol{R}^{\mathrm{T}}(\boldsymbol{a}(t) - \boldsymbol{g}) + \boldsymbol{b}_a(t) + \boldsymbol{\eta}_a(t)$$

代入上式可以得到**测量值和状态量**之间的关系：

$$\boldsymbol{R}(t+\Delta t) = \boldsymbol{R}(t)\,\mathrm{Exp}\left(\left(\tilde{\boldsymbol{\omega}} - \boldsymbol{b}_g(t) - \boldsymbol{\eta}_{gd}(t)\right)\Delta t\right)$$
$$\boldsymbol{v}(t+\Delta t) = \boldsymbol{v}(t) + \boldsymbol{g}\Delta t + \boldsymbol{R}(t)\left(\tilde{\boldsymbol{a}} - \boldsymbol{b}_a(t) - \boldsymbol{\eta}_{ad}(t)\right)\Delta t$$
$$\boldsymbol{p}(t+\Delta t) = \boldsymbol{p}(t) + \boldsymbol{v}(t)\Delta t + \frac{1}{2}\boldsymbol{g}\Delta t^2 + \frac{1}{2}\boldsymbol{R}(t)\left(\tilde{\boldsymbol{a}} - \boldsymbol{b}_a(t) - \boldsymbol{\eta}_{ad}(t)\right)\Delta t^2$$

我们完全可以用这种约束来构建图优化，对IMU相关的问题进行求解。但是这组方程刻画的时间太短，或者说，IMU的测量频率太高。我们并**不希望优化过程随着IMU数据进行调用，那样太浪费计算资源**。于是，预积分方法应运而生，它可以把一段时间的IMU数据累计起来统一处理。

假设关键帧 i 和 j 之间的IMU数据被累计起来，这种被累计起来的观测称为 **预积分**

$$\boldsymbol{R}_j = \boldsymbol{R}_i \prod_{k=i}^{j-1}\left(\mathrm{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_k - \boldsymbol{b}_{g,k} - \boldsymbol{\eta}_{gd,k}\right)\Delta t\right)\right.$$
$$\boldsymbol{v}_j = \boldsymbol{v}_i + \boldsymbol{g}\Delta t_{ij} + \sum_{k=i}^{j-1}\boldsymbol{R}_k\left(\tilde{\boldsymbol{a}_k} - \boldsymbol{b}_{a,k} - \boldsymbol{\eta}_{ad,k}\right)\Delta t$$
$$\boldsymbol{p}_j = \boldsymbol{p}_i + \sum_{k=i}^{j-1}\boldsymbol{v}_k\Delta t + \frac{1}{2}\boldsymbol{g}\Delta t_{ij}^2 + \frac{1}{2}\sum_{k=i}^{j-1}\boldsymbol{R}_k\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,k} - \boldsymbol{\eta}_{ad,k}\right)\Delta t^2$$

在已知 i 时刻状态和所有测量时，该式可以用于推断 j 时刻的状态，上式为传统意义的**直接积分**

直接积分的缺点：

如果我们对 i 时刻的状态进行优化，那么 i +1 , i + 2 … ，j -1 时刻的状态也会跟着发生改变，这个积分就必须重新计算。为此，我们对上式进行改变，定义相对运动量为：

$$\Delta \boldsymbol{R}_{ij} \doteq \boldsymbol{R}_i^{\mathrm{T}} \boldsymbol{R}_j = \prod_{k=i}^{j-1} \operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_k - \boldsymbol{b}_{g,k} - \boldsymbol{\eta}_{gd,k}\right) \Delta t\right)$$

$$\Delta \boldsymbol{v}_{ij} \doteq \boldsymbol{R}_i^{\mathrm{T}} \left(\boldsymbol{v}_j - \boldsymbol{v}_i - \boldsymbol{g}\Delta t_{ij}\right) = \sum_{k=i}^{j-1} \Delta \boldsymbol{R}_{ik} \left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,k} - \boldsymbol{\eta}_{ad,k}\right) \Delta t$$

$$\begin{aligned} \Delta \boldsymbol{p}_{ij} &\doteq \boldsymbol{R}_i^{\mathrm{T}} \left(\boldsymbol{p}_j - \boldsymbol{p}_i - \boldsymbol{v}_i \Delta t_{ij} - \frac{1}{2}\boldsymbol{g}\Delta t_{ij}^2\right) \\ &= \sum_{k=i}^{j-1} \left[\Delta \boldsymbol{v}_{ik} \Delta t + \frac{1}{2} \Delta \boldsymbol{R}_{ik} \left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,k} - \boldsymbol{\eta}_{ad,k}\right) \Delta t^2\right] \end{aligned}$$

这种改变实际上只是计算了某种从 i 到 j 的“差值”。这个定义在计算上有一些有趣的性质

1. 我们不妨考虑从 i 时刻出发，此时这三个量都为零。在 $i+1$ 时刻，我们计算出 $\Delta \boldsymbol{R}_{i,i+1}, \boldsymbol{v}_{i,i+1}$ 和 $\Delta \boldsymbol{p}_{i,i+1}$ 。而在 $i+2$ 时刻时，由于这三个式子都是累乘或累加的形式，只需在 $i, i+1$ 时刻的结果之上，加上第 $i+2$ 时刻的测量值即可。这在计算层面带来了很大的便利。进一步，我们还会发现这种性质对后续计算各种雅可比矩阵都非常方便。
2. 上述所有计算都和 $\boldsymbol{R}_i$ 的取值无关。即使 $\boldsymbol{R}_i$ 的估计值发生改变，这些量也无需重新计算。这又是非常方便的一个特性。
3. 不过，如果零偏 $\boldsymbol{b}_{a,k}$ 或 $\boldsymbol{b}_{g,k}$ 发生变化，那么上述式子理论上还是需要重新计算。然而，我们也可以通过“**修正**”而非“**重新计算**”的思路，来调整我们的预积分量。
4. 请注意，预积分量并没有直接的物理含义。尽管符号上用了 $\Delta \boldsymbol{v}, \Delta \boldsymbol{p}$ 之类的样子，但它并不表示某两个速度或位置上的偏差。它只是如此定义而已。当然，从量纲上来说，应该与角度、速度、位移对应。
5. 同样地，由于预积分量不是直接的物理量，这种“测量模型”的噪声也必须从原始的IMU噪声推导而来。

2. 预积分的测量模型

预积分内部带有IMU的零偏量，因此不可避免地会依赖此时的零偏量估计。为了处理这种依赖，我们对预积分定义作一些工程上的调整：

1. 我们首先认为 i 时刻的零偏是固定的，并且在整个预积分计算过程中也都是固定的。
2. 我们作出预积分对零偏量的一阶线性化模型，即，舍弃对零偏量的高阶项。
3. 当零偏估计发生改变时，用这个线性模型来修正预积分。

首先，我们固定 i 时刻的零偏估计，来分析预积分的噪声。无论是图优化还是滤波器技术，都需要知道某个测量量究竟含有多大的噪声

从旋转开始计算，因为旋转相对来说比较容易。利用BCH展开，可以作出下列的近似：

$$\begin{aligned} \Delta \boldsymbol{R}_{ij} &= \prod_{k=i}^{j-1} \underbrace{\operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_k - \boldsymbol{b}_{g,k} - \boldsymbol{\eta}_{gd,k}\right) \Delta t\right)}_{\text{利用 BCH:} \approx \operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_k - \boldsymbol{b}_{i,g} \Delta t\right)\right) \operatorname{Exp}\left(-\boldsymbol{J}_{r,k} \boldsymbol{\eta}_{gd,k} \Delta t\right)} \\ &\approx \prod_{k=i}^{j-1} \left[\operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_k - \boldsymbol{b}_{i,g}\right) \Delta t\right) \operatorname{Exp}\left(-\boldsymbol{J}_{r,k} \boldsymbol{\eta}_{gd,k} \Delta t\right)\right] \end{aligned}$$

在这个式子中，把噪声项分离出去，从而定义**预积分测量**$\Delta\tilde{\boldsymbol{R}}_{ij}$ （测量量会带有上标符号）

$\Delta\tilde{\boldsymbol{R}}_{ij} = \prod_{k=i}^{j-1} \operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_k - \boldsymbol{b}_{g,i}\right)\Delta t\right)$

基于这种巧妙的定义方式，可以把上式改写成：

$$
\begin{aligned}
\Delta\boldsymbol{R}_{ij} &= \underbrace{\operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_i - \boldsymbol{b}_{i,g}\right)\Delta t\right)}_{\Delta\tilde{\boldsymbol{R}}_{i,i+1}} \operatorname{Exp}\left(-\boldsymbol{J}_{r,i}\boldsymbol{\eta}_{gd,i}\Delta t\right) \underbrace{\operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_{i+1} - \boldsymbol{b}_{i,g}\right)\Delta t\right)}_{\Delta\tilde{\boldsymbol{R}}_{i+1,i+2}} \operatorname{Exp}\left(-\boldsymbol{J}_{r,i+1}\boldsymbol{\eta}_{gd,i}\Delta t\right)\cdots \\
&= \Delta\tilde{\boldsymbol{R}}_{i,i+1} \underbrace{\operatorname{Exp}\left(-\boldsymbol{J}_{r,i}\boldsymbol{\eta}_{gd,i}\Delta t\right)\Delta\tilde{\boldsymbol{R}}_{i+1,i+2}}_{=\Delta\tilde{\boldsymbol{R}}_{i+1,i+2}\operatorname{Exp}\left(-\Delta\tilde{\boldsymbol{R}}_{i+1,i+2}^{\mathrm{T}}\boldsymbol{J}_{r,i}\boldsymbol{\eta}_{gd,i}\Delta t\right)} \operatorname{Exp}\left(-\boldsymbol{J}_{r,i+1}\boldsymbol{\eta}_{gd,i}\Delta t\right)\cdots \\
&= \Delta\tilde{\boldsymbol{R}}_{i,i+2} \operatorname{Exp}\left(-\Delta\tilde{\boldsymbol{R}}_{i+1,i+2}^{\mathrm{T}}\boldsymbol{J}_{r,i}\boldsymbol{\eta}_{gd,i}\Delta t\right) \operatorname{Exp}\left(-\boldsymbol{J}_{r,i+1}\boldsymbol{\eta}_{gd,i}\Delta t\right)\Delta\tilde{\boldsymbol{R}}_{i+2,i+3}\cdots
\end{aligned}
$$

不断地把观测置换到左侧，并把噪声置换到右侧，并且把噪声项内部的$\Delta\tilde{\boldsymbol{R}}$ 项合并，可以得到：

$$
\begin{aligned}
\Delta\boldsymbol{R}_{ij} &= \Delta\tilde{\boldsymbol{R}}_{ij} \prod_{k=i}^{j-1} \operatorname{Exp}\left(-\Delta\tilde{\boldsymbol{R}}_{k+1,j}^{\mathrm{T}}\boldsymbol{J}_{r,k}\boldsymbol{\eta}_{gd,k}\Delta t\right) \\
&\doteq \Delta\tilde{\boldsymbol{R}}_{ij} \operatorname{Exp}\left(-\delta\boldsymbol{\phi}_{ij}\right)
\end{aligned}
$$

速度部分:

$$
\begin{aligned}
&\Delta\boldsymbol{v}_{ij} = \textstyle\sum_{k=i}^{j-1} \Delta\boldsymbol{R}_{ik}\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,k} - \boldsymbol{\eta}_{ad,k}\right)\Delta t \\
&= \textstyle\sum_{k=i}^{j-1} \Delta\tilde{\boldsymbol{R}}_{ik} \underbrace{\operatorname{Exp}\left(-\delta\phi_{ik}\right)}_{\approx\boldsymbol{I}-\delta\phi_{ik}} \left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,k} - \boldsymbol{\eta}_{ad,k}\right)\Delta t \\
&= \textstyle\sum_{k=i}^{j-1} \Delta\tilde{\boldsymbol{R}}_{ik}\left(\boldsymbol{I} - \delta\boldsymbol{\phi}_{ik}^{\wedge}\right)\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,k} - \boldsymbol{\eta}_{ad,k}\right)\Delta t
\end{aligned}
$$

舍掉上式中的噪声二阶小量，并且定义**预积分速度观测量**为:

$\Delta\tilde{\boldsymbol{v}}_{ij} = \sum_{k=i}^{j-1} \Delta\tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,i}\right)\Delta t$

那么上式化简为：

$$
\begin{aligned}
\Delta\boldsymbol{v}_{ij} &= \sum_{k=i}^{j-1} \underbrace{\Delta\tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,i}\right)\Delta t}_{\text{累加此项}} + \Delta\tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,i}\right)^{\wedge}\delta\boldsymbol{\phi}_{ik}\Delta t - \Delta\tilde{\boldsymbol{R}}_{ik}\boldsymbol{\eta}_{ad,k}\Delta t \\
&= \Delta\tilde{\boldsymbol{v}}_{ij} + \sum_{k=i}^{j-1} \Delta\tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,i}\right)^{\wedge}\delta\boldsymbol{\phi}_{ik}\Delta t - \Delta\tilde{\boldsymbol{R}}_{ik}\boldsymbol{\eta}_{ad,k}\Delta t \\
&= \Delta\tilde{\boldsymbol{v}}_{ij} - \delta\boldsymbol{v}_{ij}
\end{aligned}
$$

平移部分：

$$
\begin{aligned}
\Delta \boldsymbol{p}_{ij} &= \sum_{k=i}^{j-1}\left[\Delta \boldsymbol{v}_{ik}\Delta t+\frac{1}{2}\Delta \boldsymbol{R}_{ik}\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,k}-\boldsymbol{\eta}_{ad,k}\right)\Delta t^2\right] \\
&= \sum_{k=i}^{j-1}[\left(\Delta \tilde{\boldsymbol{v}}_{ik}-\delta \boldsymbol{v}_{ik}\right)\Delta t+\frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}\underbrace{\operatorname{Exp}\left(-\delta \boldsymbol{\phi}_{ik}\right)}_{\boldsymbol{I}-\delta \boldsymbol{\phi}_{ij}^{\wedge}}\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,k}-\boldsymbol{\eta}_{ad,k}\right)\Delta t^2] \\
&\approx \sum_{k=i}^{j-1}\left[\left(\Delta \tilde{\boldsymbol{v}}_{ik}-\delta \boldsymbol{v}_{ik}\right)\Delta t+\frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}\left(\boldsymbol{I}-\delta \boldsymbol{\phi}_{ik}^{\wedge}\right)\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}\right)\Delta t^2-\frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}\boldsymbol{\eta}_{ad,k}\Delta t^2\right]
\end{aligned}
$$

我们定义**预积分位移观测量**

$$
\begin{aligned}
&\approx \sum_{k=i}^{j-1}\left[\Delta \tilde{\boldsymbol{v}}_{ik}\Delta t+\frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}\right)\Delta t^2-\delta \boldsymbol{v}_{ik}\Delta t+\frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}\right)^{\wedge}\delta \boldsymbol{\phi}_{ik}\Delta t^2-\right. \\
&\left.\frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}\boldsymbol{\eta}_{ad,k}\Delta t^2\right]
\end{aligned}
$$

为：

$$
\Delta \tilde{\boldsymbol{p}}_{ij}=\sum_{k=i}^{j-1}\left[\left(\Delta \tilde{\boldsymbol{v}}_{ik}\Delta t\right)+\frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}\right)\Delta t^2\right]
$$

前面式子可以写成：

$$
\begin{aligned}
\Delta \boldsymbol{p}_{ij} &= \Delta \tilde{\boldsymbol{p}}_{ij}+\sum_{k=i}^{j-1}\left[-\delta \boldsymbol{v}_{ik}\Delta t+\frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}\right)^{\wedge}\delta \boldsymbol{\phi}_{ik}\Delta t^2-\frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}\boldsymbol{\eta}_{ad,k}\Delta t^2\right] \\
&\doteq \Delta \tilde{\boldsymbol{p}}_{ij}-\delta \boldsymbol{p}_{ij}
\end{aligned}
$$

于是，此三式共同定义了预积分的三个观测量和它们的噪声。将它们代回最初的定义式，可以简单写为：

$$
\begin{aligned}
\Delta \tilde{\boldsymbol{R}}_{ij} &= \boldsymbol{R}_i^{\mathrm{T}}\boldsymbol{R}_j\operatorname{Exp}\left(\delta \boldsymbol{\phi}_{ij}\right) \\
\Delta \tilde{\boldsymbol{v}}_{ij} &= \boldsymbol{R}_i^{\mathrm{T}}\left(\boldsymbol{v}_j-\boldsymbol{v}_i-\boldsymbol{g}\Delta t_{ij}\right)+\delta \boldsymbol{v}_{ij} \\
\Delta \tilde{\boldsymbol{p}}_{ij} &= \boldsymbol{R}_i^{\mathrm{T}}\left(\boldsymbol{p}_j-\boldsymbol{p}_i-\boldsymbol{v}_i\Delta t_{ij}-\frac{1}{2}\boldsymbol{g}\Delta t_{ij}^2\right)+\delta \boldsymbol{p}_{ij}
\end{aligned}
$$

这个式子归纳了前面我们讨论的内容，显示了预积分的几大优点：

1. 它的左侧是可以通过传感器数据积分得到的观测量，右侧是根据状态变量推断出来的预测值，再加上一个随机噪声
2. 左侧变量的定义方式非常适合程序实现

   $$
   \begin{aligned}
   \Delta \tilde{\boldsymbol{R}}_{ij} &= \prod_{k=i}^{j-1}\operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_k-\boldsymbol{b}_{g,i}\right)\Delta t\right) \\
   \Delta \tilde{\boldsymbol{v}}_{ij} &= \sum_{k=i}^{j-1}\Delta \tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}\right)\Delta t \\
   \Delta \tilde{\boldsymbol{p}}_{ij} &= \sum_{k=i}^{j-1}\left[\left(\Delta \tilde{\boldsymbol{v}}_{ik}\Delta t\right)+\frac{1}{2}\Delta \tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}\right)\Delta t^2\right]
   \end{aligned}
   $$

   并且如果知道k时刻的预积分观测，很容易根据 k+1 时刻的传感器读数，计算出k +1 时刻的预积分观测量
3. 从右侧来看，也很容易根据 i 和 j 时刻的状态变量来推测预积分观测量的大小，从而写出误差公式，构建最小二乘问题

现在的问题是：预积分的噪声是否符合零均值的高斯分布？如果是，它的协方差有多大？和IMU本身的噪声之间是什么关系？

3. 预积分噪声模型

将复杂的噪声项线性化，保留一阶项系数，然后推导线性模型下的协方差矩阵变化。这是一种非常常见的处理思路，对许多复杂模型都很有效。

先从旋转的噪声开始看：

$$\operatorname{Exp}\left(-\delta\boldsymbol{\phi}_{ij}\right)=\prod_{k=i}^{j-1}\operatorname{Exp}\left(-\Delta\tilde{\boldsymbol{R}}_{k+1}^{\mathrm{T}}\boldsymbol{J}_{r,k}\boldsymbol{\eta}_{gd,k}\Delta t\right)$$

作为随机变量的 $\delta\phi_{ij}$ 只和随机变量 $\boldsymbol{\eta}_{gd}$ 有关，而其他的都是确定的观测量。当我们线性化后取期望时，由于 $\boldsymbol{\eta}_{gd}$ 为白噪声，因此 $\delta\phi_{ij}$ 均值也为零

为了分析它的协方差，我们需要对上式进行线性化。对两侧取 Log， 可得：

$$\delta\boldsymbol{\phi}_{ij}=-\log\left(\prod_{k=i}^{j-1}\operatorname{Exp}\left(-\Delta\tilde{\boldsymbol{R}}_{k+1}^{\mathrm{T}}\boldsymbol{J}_{r,k}\boldsymbol{\eta}_{gd,k}\Delta t\right)\right)$$

该式又可以通过BCH进行线性近似。同时，由于内部的系数项 $-\Delta\tilde{\boldsymbol{R}}_{k+1}^{\mathrm{T}}\boldsymbol{J}_{r,k}\boldsymbol{\eta}_{gd,k}\Delta t$ 接近于零，且为噪声，我们可将BCH近似的右雅可比取为单位阵，可以得到：

$$\delta\boldsymbol{\phi}_{ij}\approx\sum_{k=i}^{j-1}\Delta\boldsymbol{R}_{k+1,j}^{\mathrm{T}}\boldsymbol{J}_{r,k}\boldsymbol{\eta}_{gd,k}\Delta t$$

此式是高斯随机变量的线性组合，它的结果依然是高斯的。同时，由于预积分的累加特性，预测分观测量的噪声也会随着时间不断累加

上式是累加形式的，很容易将其写成递推的形式：

$$\begin{aligned}\delta\boldsymbol{\phi}_{ij}&\approx\sum_{k=i}^{j-1}\Delta\tilde{\boldsymbol{R}}_{k+1,j}^{\mathrm{T}}\boldsymbol{J}_{r,k}\boldsymbol{\eta}_{gd,k}\Delta t\\&=\sum_{k=i}^{j-2}\Delta\tilde{\boldsymbol{R}}_{k+1,j}^{\mathrm{T}}\boldsymbol{J}_{r,k}\boldsymbol{\eta}_{gd,k}\Delta t+\underbrace{\Delta\boldsymbol{R}_{j,j}^{\mathrm{T}}}_{=\boldsymbol{I}}\boldsymbol{J}_{r,j-1}\boldsymbol{\eta}_{gd,j-1}\Delta t\\&=\sum_{k=i}^{j-2}\underbrace{\Delta\tilde{\boldsymbol{R}}_{k+1,j}^{\mathrm{T}}}\boldsymbol{J}_{r,k}\boldsymbol{\eta}_{gd,k}\Delta t+\boldsymbol{J}_{r,j-1}\boldsymbol{\eta}_{gd,j-1}\Delta t\\&=\Delta\tilde{\boldsymbol{R}}_{j-1,j}^{\mathrm{T}}\sum_{k=i}^{j-2}\Delta\tilde{\boldsymbol{R}}_{k+1,j-1}^{\mathrm{T}}\boldsymbol{J}_{r,k}\boldsymbol{\eta}_{gd,k}\Delta t+\boldsymbol{J}_{r,j-1}\boldsymbol{\eta}_{gd,j-1}\Delta t\\&=\Delta\tilde{\boldsymbol{R}}_{j-1,j}^{\mathrm{T}}\delta\boldsymbol{\phi}_{i,j-1}+\boldsymbol{J}_{r,j-1}\boldsymbol{\eta}_{gd,j-1}\Delta t\end{aligned}$$

该式描述了如何从 j - 1 时刻的噪声推断到 j 时刻。显然这是一个线性系统，设 j -1 时刻的 $\delta\boldsymbol{\phi}_{i,j-1}$ 的协方差为 $\boldsymbol{\Sigma}_{j-1}$， $\boldsymbol{\eta}_{gd}$ 的协方差为 $\boldsymbol{\Sigma}_{\eta_{gd}}$， 则有：

$$\boldsymbol{\Sigma}_{j}=\Delta\tilde{\boldsymbol{R}}_{j-1,j}^{\mathrm{T}}\boldsymbol{\Sigma}_{j-1}\Delta\tilde{\boldsymbol{R}}_{j-1,j}+\boldsymbol{J}_{r,j-1}\boldsymbol{\Sigma}_{\eta_{gd}}\boldsymbol{J}_{r,j-1}^{\mathrm{T}}\Delta t^{2}$$

表明预积分误差会随着累计变大，预积分观测量也会变得越来越不确定

速度部分的噪声：

$$\delta\boldsymbol{v}_{ij}\approx\sum_{k=i}^{j-1}\left[-\Delta\tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_{k}-\boldsymbol{b}_{a,i}\right)^{\wedge}\delta\boldsymbol{\phi}_{ik}\Delta t+\Delta\tilde{\boldsymbol{R}}_{ik}\boldsymbol{\eta}_{ad,k}\Delta t\right]$$

写成累加的形式：

$$\begin{aligned}\delta\boldsymbol{v}_{ij}=&\sum_{k=i}^{j-1}\left[-\Delta\tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_{k}-\boldsymbol{b}_{a,i}\right)^{\wedge}\delta\boldsymbol{\phi}_{ik}\Delta t+\Delta\tilde{\boldsymbol{R}}_{ik}\boldsymbol{\eta}_{ad,k}\Delta t\right]\\=&\sum_{k=i}^{j-2}\left[-\Delta\tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_{k}-\boldsymbol{b}_{a,i}\right)^{\wedge}\delta\boldsymbol{\phi}_{ik}\Delta t+\Delta\tilde{\boldsymbol{R}}_{ik}\boldsymbol{\eta}_{ad,k}\Delta t\right]\\&-\Delta\tilde{\boldsymbol{R}}_{i,j-1}\left(\tilde{\boldsymbol{a}}_{j-1}-\boldsymbol{b}_{a,i}\right)^{\wedge}\delta\boldsymbol{\phi}_{i,j-1}\Delta t+\Delta\tilde{\boldsymbol{R}}_{i,j-1}\boldsymbol{\eta}_{ad,j-1}\Delta t\\=&\delta\boldsymbol{v}_{i,j-1}-\Delta\tilde{\boldsymbol{R}}_{i,j-1}\left(\tilde{\boldsymbol{a}}_{j-1}-\boldsymbol{b}_{a,i}\right)^{\wedge}\delta\boldsymbol{\phi}_{i,j-1}\Delta t+\Delta\tilde{\boldsymbol{R}}_{i,j-1}\boldsymbol{\eta}_{ad,j-1}\Delta t\end{aligned}$$

平移部分：

$$
\begin{aligned}
\delta \boldsymbol{p}_{ij} =& \sum_{k=i}^{j-1}\left[\delta \boldsymbol{v}_{ik} \Delta t-\frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_{k}-\boldsymbol{b}_{a, i}\right)^{\wedge} \delta \boldsymbol{\phi}_{ik} \Delta t^{2}+\frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{ik} \boldsymbol{\eta}_{ad, k} \Delta t^{2}\right] \\
=& \sum_{k=i}^{j-2}\left[\delta \boldsymbol{v}_{ik} \Delta t-\frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_{k}-\boldsymbol{b}_{a, i}\right)^{\wedge} \delta \boldsymbol{\phi}_{ik} \Delta t^{2}+\frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{ik} \boldsymbol{\eta}_{ad, k} \Delta t^{2}\right] \\
&+\delta \boldsymbol{v}_{i, j-1} \Delta t-\frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{i, j-1}\left(\tilde{\boldsymbol{a}}_{j-1}-\boldsymbol{b}_{a, i}\right)^{\wedge} \delta \boldsymbol{\phi}_{i, j-1} \Delta t^{2}+\frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{i, j-1} \boldsymbol{\eta}_{ad, j-1} \Delta t^{2} \\
=& \delta \boldsymbol{p}_{i, j-1}+\delta \boldsymbol{v}_{i, j-1} \Delta t-\frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{i, j-1}\left(\tilde{\boldsymbol{a}}_{j-1}-\boldsymbol{b}_{a, i}\right)^{\wedge} \delta \boldsymbol{\phi}_{i, j-1} \Delta t^{2}+\frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{i, j-1} \boldsymbol{\eta}_{ad, j-1} \Delta t^{2}
\end{aligned}
$$

至此，推导了如何从 j -1 时刻将噪声递推至 j 时刻，下面整理成矩阵形式：

$$
\boldsymbol{\eta}_{ik}=\left[\begin{array}{c}
\delta \boldsymbol{\phi}_{ik} \\
\delta \boldsymbol{v}_{ik} \\
\delta \boldsymbol{p}_{ik}
\end{array}\right]
$$

并且把IMU的零偏噪声定义为:

$$
\boldsymbol{\eta}_{d, j}=\left[\begin{array}{c}
\boldsymbol{\eta}_{gd, j} \\
\boldsymbol{\eta}_{ad, j}
\end{array}\right]
$$

那么从 $\eta_{i,j-1}$ 到 $\eta_{i,j}$ 的递推式可以写为：

$$
\boldsymbol{\eta}_{ij}=\boldsymbol{A}_{j} \boldsymbol{\eta}_{i, j-1}+\boldsymbol{B}_{j} \boldsymbol{\eta}_{d, j-1}
$$

其中的系数矩阵为：

$$
\boldsymbol{A}_{j}=\left[\begin{array}{ccc}
\Delta \tilde{\boldsymbol{R}}_{j-1, j}^{\mathrm{T}} & \mathbf{0} & \mathbf{0} \\
-\Delta \tilde{\boldsymbol{R}}_{i, j-1}\left(\tilde{\boldsymbol{a}}_{j-1}-\boldsymbol{b}_{a, i}\right)^{\wedge} \Delta t & \boldsymbol{I} & \mathbf{0} \\
-\frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{i, j-1}\left(\tilde{\boldsymbol{a}}_{j-1}-\boldsymbol{b}_{a, i}\right)^{\wedge} \Delta t^{2} & \Delta t & \boldsymbol{I}
\end{array}\right]
$$

$$
\boldsymbol{B}_{j}=\left[\begin{array}{cc}
\boldsymbol{J}_{r, j-1} \Delta t & \mathbf{0} \\
\mathbf{0} & \Delta \tilde{\boldsymbol{R}}_{i, j-1} \Delta t \\
\mathbf{0} & \frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{i, j-1} \Delta t^{2}
\end{array}\right]
$$

矩阵形式更清晰地显示了几个噪声项之间累计递推关系。

由于它们的累加关系，在程序实现中也十分便捷。

4. 零偏的更新

先前的讨论都假设了在 i 时刻的IMU零偏恒定不变，当然这都是为了方便后续的计算。然而**在实际的图优化中，我们经常会对状态变量（优化变量）进行更新**。那么，理论上来讲，如果IMU零偏发生了变化，预积分应该重新计算，因为预积分的每一步都用到了 i 时刻的IMU零偏。但是实际操作过程中，我们也可以选用一种讨巧的做法：**假定预积分观测是随零偏线性变化的**，虽然实际上并不是线性变化的，但我们总可以对一个复杂函数做线性化并保留一阶项,然后在原先的观测量上进行修正

把预积分观测量看成是 $b_{g,i}, b_{a,i}$的 函数，当其更新了 $\delta b_{g,i}, \delta b_{a,i}$ 之后，预积分作如下修正：

$$
\Delta \tilde{\boldsymbol{R}}_{ij}\left(\boldsymbol{b}_{g, i}+\delta \boldsymbol{b}_{g, i}\right)=\Delta \tilde{\boldsymbol{R}}_{ij}\left(\boldsymbol{b}_{g, i}\right) \operatorname{Exp}\left(\frac{\partial \Delta \tilde{\boldsymbol{R}}_{ij}}{\partial \boldsymbol{b}_{g}} \delta \boldsymbol{b}_{g, i}\right)
$$

$$
\Delta \tilde{\boldsymbol{v}}_{ij}\left(\boldsymbol{b}_{g, i}+\delta \boldsymbol{b}_{g, i}, \boldsymbol{b}_{a, i}+\delta \boldsymbol{b}_{a, i}\right)=\Delta \tilde{\boldsymbol{v}}_{ij}\left(\boldsymbol{b}_{g, i}, \boldsymbol{b}_{a, i}\right)+\frac{\partial \Delta \tilde{\boldsymbol{v}}_{ij}}{\partial \boldsymbol{b}_{g, i}} \delta \boldsymbol{b}_{g, i}+\frac{\partial \Delta \tilde{\boldsymbol{v}}_{ij}}{\partial \boldsymbol{b}_{a, i}} \delta \boldsymbol{b}_{a, i}
$$

$$
\Delta \tilde{\boldsymbol{p}}_{ij}\left(\boldsymbol{b}_{g, i}+\delta \boldsymbol{b}_{g, i}, \boldsymbol{b}_{a, i}+\delta \boldsymbol{b}_{a, i}\right)=\Delta \tilde{\boldsymbol{p}}_{ij}\left(\boldsymbol{b}_{g, i}, \boldsymbol{b}_{a, i}\right)+\frac{\partial \Delta \tilde{\boldsymbol{p}}_{ij}}{\partial \boldsymbol{b}_{g, i}} \delta \boldsymbol{b}_{g, i}+\frac{\partial \Delta \tilde{\boldsymbol{p}}_{ij}}{\partial \boldsymbol{b}_{a, i}} \delta \boldsymbol{b}_{a, i}
$$

如何计算上面列写的几个偏导数（雅可比）呢?

首先来考虑旋转。预积分旋转观测量可以写为：

$$
\begin{aligned}
\Delta\tilde{\boldsymbol{R}}_{ij}\left(\boldsymbol{b}_{g,i}+\delta\boldsymbol{b}_{g,i}\right) &= \prod_{k=i}^{j-1}\operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_k-\left(\boldsymbol{b}_{g,i}+\delta\boldsymbol{b}_{g,i}\right)\right)\Delta t\right) \\
&= \prod_{k=i}^{j-1}\operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_k-\boldsymbol{b}_{g,i}\right)\Delta t\right)\operatorname{Exp}\left(-\boldsymbol{J}_{r,k}\delta\boldsymbol{b}_{g,i}\Delta t\right) \\
&= \underbrace{\operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_i-\boldsymbol{b}_{g,i}\right)\Delta t\right)}_{\Delta\tilde{\boldsymbol{R}}_{i,i+1}}\operatorname{Exp}\left(-\boldsymbol{J}_{r,i}\delta\boldsymbol{b}_{g,i}\Delta t\right)\underbrace{\operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_{i+1}-\boldsymbol{b}_{g,i}\right)\Delta t\right)}_{\Delta\tilde{\boldsymbol{R}}_{i+1,i+2}} \\
&= \Delta\tilde{\boldsymbol{R}}_{i,i+1}\Delta\tilde{\boldsymbol{R}}_{i+1,i+2}\operatorname{Exp}\left(-\Delta\tilde{\boldsymbol{R}}_{i+1,i+2}^{\mathrm{T}}\boldsymbol{J}_{r,i}\delta\boldsymbol{b}_{g,i}\Delta t\right)\cdots \\
&= \Delta\tilde{\boldsymbol{R}}_{ij}\prod_{k=i}^{j-1}\operatorname{Exp}\left(-\Delta\tilde{\boldsymbol{R}}_{k+1,j}^{\mathrm{T}}\boldsymbol{J}_{r,k}\delta\boldsymbol{b}_{g,i}\Delta t\right) \\
&\approx \Delta\tilde{\boldsymbol{R}}_{ij}\operatorname{Exp}\left(-\sum_{k=i}^{j-1}\Delta\tilde{\boldsymbol{R}}_{k+1,j}^{\mathrm{T}}\boldsymbol{J}_{r,k}\Delta t\delta\boldsymbol{b}_{g,i}\right)
\end{aligned}
$$

通过这种方式我们可以算出 $\Delta\tilde{\boldsymbol{R}}_{ij}$ 相对于 $b_{g,i}$ 的雅可比矩阵，记为 $\frac{\partial\Delta\tilde{\boldsymbol{R}}_{ij}}{\partial\boldsymbol{b}_g}$

速度测量：

$$
\begin{aligned}
\Delta\tilde{\boldsymbol{v}}\left(\boldsymbol{b}_i+\delta\boldsymbol{b}_i\right) &= \sum_{k=i}^{j-1}\Delta\tilde{\boldsymbol{R}}_{ik}\left(\boldsymbol{b}_{g,i}+\delta\boldsymbol{b}_{g,i}\right)\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}-\delta\boldsymbol{b}_{a,i}\right)\Delta t \\
&= \sum_{k=i}^{j-1}\Delta\tilde{\boldsymbol{R}}_{ik}\operatorname{Exp}\left(\frac{\partial\Delta\tilde{\boldsymbol{R}}_{ik}}{\partial\boldsymbol{b}_{g,i}}\delta\boldsymbol{b}_{g,i}\right)\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}-\delta\boldsymbol{b}_{a,i}\right)\Delta t \\
&= \sum_{k=i}^{j-1}\Delta\tilde{\boldsymbol{R}}_{ik}\left(\boldsymbol{I}+\left(\frac{\partial\Delta\tilde{\boldsymbol{R}}_{ik}}{\partial\boldsymbol{b}_{g,i}}\delta\boldsymbol{b}_{g,i}\right)^{\wedge}\right)\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}-\delta\boldsymbol{b}_{a,i}\right)\Delta t \\
&\approx \Delta\boldsymbol{v}_{ij}-\sum_{k=i}^{j-1}\Delta\tilde{\boldsymbol{R}}_{ik}\Delta t\delta\boldsymbol{b}_{a,i}-\sum_{k=i}^{j-1}\Delta\tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}\right)^{\wedge}\frac{\partial\Delta\tilde{\boldsymbol{R}}_{ik}}{\partial\boldsymbol{b}_{g,i}}\Delta t\delta\boldsymbol{b}_{g,i} \\
&= \Delta\boldsymbol{v}_{ij}+\frac{\partial\Delta\boldsymbol{v}_{ij}}{\partial\boldsymbol{b}_{a,i}}\delta\boldsymbol{b}_{a,i}+\frac{\partial\Delta\boldsymbol{v}_{ij}}{\partial\boldsymbol{b}_{g,i}}\delta\boldsymbol{b}_{g,i}
\end{aligned}
$$

最后是平移部分：

$$
\begin{aligned}
\Delta\tilde{\boldsymbol{p}}_{ij}\left(\boldsymbol{b}_i+\delta\boldsymbol{b}_i\right) =& \sum_{k=i}^{j-1}\left[\left(\Delta\tilde{\boldsymbol{v}}_{ik}+\frac{\partial\Delta\boldsymbol{v}_{ij}}{\partial\boldsymbol{b}_{a,i}}\delta\boldsymbol{b}_{a,i}+\frac{\partial\Delta\boldsymbol{v}_{ij}}{\partial\boldsymbol{b}_{g,i}}\delta\boldsymbol{b}_{g,i}\right)\Delta t+\right. \\
&\left.\frac{1}{2}\Delta\tilde{\boldsymbol{R}}_{ik}\left(\boldsymbol{I}+\left(\frac{\partial\Delta\tilde{\boldsymbol{R}}_{ik}}{\partial\boldsymbol{b}_{g,i}}\delta\boldsymbol{b}_{g,i}\right)^{\wedge}\right)\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}-\delta\boldsymbol{b}_{a,i}\right)\Delta t^2\right] \\
=& \delta\tilde{\boldsymbol{p}}_{ij}+\sum_{k=i}^{j-1}\left[\frac{\partial\Delta\boldsymbol{v}_{ij}}{\partial\boldsymbol{b}_{a,i}}\Delta t-\frac{1}{2}\Delta\tilde{\boldsymbol{R}}_{ik}\Delta t^2\right]\delta\boldsymbol{b}_{a,i}+ \\
&\sum_{k=i}^{j-1}\left[\frac{\partial\Delta\boldsymbol{v}_{ij}}{\partial\boldsymbol{b}_{g,i}}\Delta t-\frac{1}{2}\Delta\tilde{\boldsymbol{R}}_{ik}\left(\tilde{\boldsymbol{a}}_k-\boldsymbol{b}_{a,i}\right)^{\wedge}\frac{\partial\Delta\tilde{\boldsymbol{R}}_{ik}}{\partial\boldsymbol{b}_{g,i}}\Delta t^2\right]\delta\boldsymbol{b}_{g,i} \\
=& \delta\tilde{\boldsymbol{p}}_{ij}+\frac{\partial\Delta\tilde{\boldsymbol{p}}_{ij}}{\partial\boldsymbol{b}_{a,i}}\delta\boldsymbol{b}_{a,i}+\frac{\partial\Delta\tilde{\boldsymbol{p}}_{ij}}{\partial\boldsymbol{b}_{g,i}}\boldsymbol{b}_{g,i}
\end{aligned}
$$

请注意上述计算都是迭代形式的。也就是说，我们只需要在程序中保留一个变量，随着数据不断进行累加即可。

把这些雅可比矩阵整理一下，得到更整齐的结果：

$$\frac{\partial \Delta \tilde{\boldsymbol{R}}_{ij}}{\partial \boldsymbol{b}_{g,i}} = -\sum_{k=i}^{j-1} \left[\Delta \tilde{\boldsymbol{R}}_{k+1,j}^{\mathrm{T}} \boldsymbol{J}_{k,r} \Delta t \right]$$

$$\frac{\partial \Delta \tilde{\boldsymbol{v}}_{ij}}{\partial \boldsymbol{b}_{a,i}} = -\sum_{k=i}^{j-1} \Delta \tilde{\boldsymbol{R}}_{ik} \Delta t$$

$$\frac{\partial \Delta \tilde{\boldsymbol{v}}_{ij}}{\partial \boldsymbol{b}_{g,i}} = -\sum_{k=i}^{j-1} \Delta \tilde{\boldsymbol{R}}_{ik} \left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,i} \right)^{\wedge} \frac{\partial \Delta \tilde{\boldsymbol{R}}_{ij}}{\partial \boldsymbol{b}_{g,i}} \Delta t$$

$$\frac{\partial \Delta \tilde{\boldsymbol{p}}_{ij}}{\partial \boldsymbol{b}_{a,i}} = \sum_{k=i}^{j-1} \frac{\partial \Delta \boldsymbol{v}_{ij}}{\partial \boldsymbol{b}_{a,i}} \Delta t - \frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{ik} \Delta t^2$$

$$\frac{\partial \Delta \tilde{\boldsymbol{p}}_{ij}}{\partial \boldsymbol{b}_{g,i}} = \sum_{k=i}^{j-1} \frac{\partial \Delta \boldsymbol{v}_{ij}}{\partial \boldsymbol{b}_{g,i}} \Delta t - \frac{1}{2} \Delta \tilde{\boldsymbol{R}}_{ik} \left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_{a,i} \right)^{\wedge} \frac{\partial \Delta \tilde{\boldsymbol{R}}_{ik}}{\partial \boldsymbol{b}_{g,i}} \Delta t^2$$

5. 预积分模型归结到图优化

定义了预积分的测量模型，推导了它的噪声模型和协方差矩阵，并说明了随着零偏量更新，预积分应该怎么更新。事实上，我们已经可以把预积分观测作为图优化的因子（Factor）或者边（Edge）了。下面我们来说明如何使用这种边，以及它推导它相对于状态变量的雅可比矩阵。

在IMU相关的应用中，通常把每个时刻的状态建模为包含旋转、平移、线速度、IMU零偏的变量，构成状态变量集合 $\mathcal{X}$:

$$\boldsymbol{x}_k = \left[\boldsymbol{R}, \boldsymbol{p}, \boldsymbol{v}, \boldsymbol{b}_a, \boldsymbol{b}_g \right]_k \in \mathcal{X}$$

预积分模型构建了关键帧 i 与关键帧 j 之间的一种约束，它的残差可以写成：

$$\boldsymbol{r}_{\Delta \boldsymbol{R}_{ij}} = \log \left(\Delta \tilde{\boldsymbol{R}}_{ij}^{\mathrm{T}} \left(\boldsymbol{R}_i^{\mathrm{T}} \boldsymbol{R}_j \right) \right)$$

$$\boldsymbol{r}_{\Delta \boldsymbol{v}_{ij}} = \boldsymbol{R}_i^T \left(\boldsymbol{v}_j - \boldsymbol{v}_i - \boldsymbol{g} \Delta t_{ij} \right) - \Delta \tilde{\boldsymbol{v}}_{ij}$$

$$\boldsymbol{r}_{\Delta \boldsymbol{p}_{ij}} = \boldsymbol{R}_i^{\mathrm{T}} \left(\boldsymbol{p}_j - \boldsymbol{p}_i - \boldsymbol{v}_i \Delta t_{ij} - \frac{1}{2} \boldsymbol{g} \Delta t_{ij}^2 \right) - \Delta \tilde{\boldsymbol{p}}_{ij}$$

通常我们会把残差统一写成一个9维的残差变量，表面上关联两个时刻的旋转，平移，速度，但是由于预积分观测内部含有IMU零偏，所以也和 i 时刻的两个零偏有关。除了预积分因子本身之外，还需要约束IMU的随机游走，因此在IMU的不同时刻零偏会存在一个约束因子。

6. 预积分的雅可比矩阵

讨论预积分相比于状态变量的雅可比矩阵。由于预积分测量已经归纳了IMU在短时间内的读数，残差相对于状态变量的雅可比推导显得十分简单

旋转与 R_i , R_j 和 $b_{g,i}$ 有关，我们用 SO(3) 的右扰动推导它：

对 ϕ_i :

$$\begin{aligned}
\boldsymbol{r}_{\Delta \boldsymbol{R}_{ij}} \left(\boldsymbol{R}_i \operatorname{Exp}\left(\boldsymbol{\phi}_i \right) \right) &= \log \left(\Delta \tilde{\boldsymbol{R}}_{ij}^{\mathrm{T}} \left(\boldsymbol{R}_i \operatorname{Exp}\left(\boldsymbol{\phi}_i \right)^{\mathrm{T}} \boldsymbol{R}_j \right) \right) \\
&= \log \left(\Delta \tilde{\boldsymbol{R}}_{ij}^{\mathrm{T}} \operatorname{Exp}\left(-\boldsymbol{\phi}_i \right) \boldsymbol{R}_i^{\mathrm{T}} \boldsymbol{R}_j \right) \\
&= \log \left(\Delta \tilde{\boldsymbol{R}}_{ij}^{\mathrm{T}} \boldsymbol{R}_i^{\mathrm{T}} \boldsymbol{R}_j \operatorname{Exp}\left(-\boldsymbol{R}_j^{\mathrm{T}} \boldsymbol{R}_i \boldsymbol{\phi}_i \right) \right) \\
&= \boldsymbol{r}_{\Delta \boldsymbol{R}_{ij}} - \boldsymbol{J}_r^{-1} \left(\boldsymbol{r}_{\Delta \boldsymbol{R}_{ij}} \right) \boldsymbol{R}_j^{\mathrm{T}} \boldsymbol{R}_i \boldsymbol{\phi}_i
\end{aligned}$$

对 ϕ_j :

$$
\begin{aligned}
\boldsymbol{r}_{\Delta \boldsymbol{R}_{ij}}\left(\boldsymbol{R}_j \operatorname{Exp}\left(\boldsymbol{\phi}_j\right)\right) &= \log\left(\Delta \tilde{\boldsymbol{R}}_{ij}^{\mathrm{T}} \boldsymbol{R}_i^{\mathrm{T}} \boldsymbol{R}_j \operatorname{Exp}\left(\boldsymbol{\phi}_j\right)\right) \\
&= \boldsymbol{r}_{\Delta \boldsymbol{R}_{ij}} + \boldsymbol{J}_r^{-1}\left(\boldsymbol{r}_{\Delta \boldsymbol{R}_{ij}}\right) \boldsymbol{\phi}_j
\end{aligned}
$$

对于零偏量要稍微麻烦一些。注意在优化过程中，零偏量应该不断地更新，而每次更新时我们会利用当前零偏来修正预积分的观测量。由于这个过程是不断进行的，我们总会有一个初始的观测量和当前修正后的观测量，在推导时必须考虑到这一点。

假设优化初始的零偏为 $b_{g,i}$,在某一步迭代时，我们当前估计出来的零偏修正为 $\delta b_{g,i}$,而当前我们修正得到的预积分旋转观测量为 $\Delta \tilde{\boldsymbol{R}}_{ij}^{'} = \Delta \tilde{\boldsymbol{R}}_{ij}\left(\boldsymbol{b}_{g,i} + \delta \boldsymbol{b}_{g,i}\right)$ ，残差为 $\boldsymbol{r}'_{\Delta} \boldsymbol{R}_{ij}$

为了求导，又在上面的基础上加上了 $\tilde{\delta} \boldsymbol{b}_{g,i}$

则：

$$
\begin{aligned}
\boldsymbol{r}_{\Delta \boldsymbol{R}_{ij}}(\boldsymbol{b}_{g,i} + \delta \boldsymbol{b}_{g,i} + \tilde{\delta} \boldsymbol{b}_{g,i}) &= \operatorname{Log}\left(\left(\Delta \tilde{\boldsymbol{R}}_{ij} \operatorname{Exp}\left(\frac{\partial \Delta \tilde{\boldsymbol{R}}_{ij}}{\partial \boldsymbol{b}_{g,i}}\left(\delta \boldsymbol{b}_{g,i} + \tilde{\delta} \boldsymbol{b}_{g,i}\right)\right)\right)^{\mathrm{T}} \boldsymbol{R}_i^{\mathrm{T}} \boldsymbol{R}_j\right) \\
&\overset{\mathrm{BCH}}{=} \operatorname{Log}\left(\left(\underbrace{\Delta \tilde{\boldsymbol{R}}_{ij} \operatorname{Exp}\left(\frac{\partial \Delta \tilde{\boldsymbol{R}}_{ij}}{\partial \boldsymbol{b}_{g,i}} \delta \boldsymbol{b}_{g,i}\right)}_{\Delta \tilde{\boldsymbol{R}}_{ij}^{\prime}} \operatorname{Exp}\left(\boldsymbol{J}_{r,b} \frac{\partial \Delta \tilde{\boldsymbol{R}}_{ij}}{\partial \boldsymbol{b}_{g,i}} \tilde{\delta} \boldsymbol{b}_{g,i}\right)\right)^{\mathrm{T}} \boldsymbol{R}_i^{\mathrm{T}} \boldsymbol{R}_j\right) \\
&= \operatorname{Log}\left(\operatorname{Exp}\left(-\boldsymbol{J}_{b,r} \frac{\partial \Delta \tilde{\boldsymbol{R}}_{ij}}{\partial \boldsymbol{b}_{g,i}} \tilde{\delta} \boldsymbol{b}_{g,i}\right) \underbrace{\left(\Delta \tilde{\boldsymbol{R}}_{ij}^{\prime}\right)^{\mathrm{T}} \boldsymbol{R}_i^{\mathrm{T}} \boldsymbol{R}_j}_{\operatorname{Exp}\left(\boldsymbol{r}'_{\Delta \boldsymbol{R}_{ij}}\right)}\right) \\
&= \operatorname{Log}\left(\operatorname{Exp}\left(\boldsymbol{r}'_{\Delta \boldsymbol{R}_{ij}}\right) \operatorname{Exp}\left(-\operatorname{Exp}\left(\boldsymbol{r}'_{\Delta \boldsymbol{R}_{ij}}\right)^{\mathrm{T}} \boldsymbol{J}_{b,r} \frac{\partial \Delta \tilde{\boldsymbol{R}}_{ij}}{\partial \boldsymbol{b}_{g,i}} \tilde{\delta} \boldsymbol{b}_{g,i}\right)\right) \\
&= \boldsymbol{r}'_{\Delta \boldsymbol{R}_{ij}} - \boldsymbol{J}_r^{-1}(\boldsymbol{r}'_{\Delta \boldsymbol{R}_{ij}}) \operatorname{Exp}\left(\boldsymbol{r}'_{\Delta \boldsymbol{R}_{ij}}\right)^{\mathrm{T}} \boldsymbol{J}_{b,r} \frac{\partial \Delta \tilde{\boldsymbol{R}}_{ij}}{\partial \boldsymbol{b}_{g,i}} \tilde{\delta} \boldsymbol{b}_{g,i}
\end{aligned}
$$

所以最后得到：

$\frac{\partial \boldsymbol{r}_{\Delta \boldsymbol{R}_{ij}}}{\partial \boldsymbol{b}_{g,i}} = -\boldsymbol{J}_r^{-1}\left(\boldsymbol{r}'_{\Delta \boldsymbol{R}_{ij}}\right) \operatorname{Exp}\left(\boldsymbol{r}'_{\Delta \boldsymbol{R}_{ij}}\right)^{\mathrm{T}} \boldsymbol{J}_{b,r} \frac{\partial \Delta \tilde{\boldsymbol{R}}_{ij}}{\partial \boldsymbol{b}_{g,i}} \tilde{\delta} \boldsymbol{b}_{g,i}$

速度项更为简单，它与 v_i, v_j 呈线性关系，不难得到：

$\frac{\partial \boldsymbol{r}_{\Delta \boldsymbol{v}_{ij}}}{\partial \boldsymbol{v}_i} = -\boldsymbol{R}_i^{\mathrm{T}}, \quad \frac{\partial \boldsymbol{r}_{\Delta \boldsymbol{v}_{ij}}}{\partial \boldsymbol{v}_j} = \boldsymbol{R}_i^{\mathrm{T}}$

对旋转部分只需要做一阶泰勒展开即可：

$$
\begin{aligned}
\boldsymbol{r}_{\Delta \boldsymbol{v}_{ij}}\left(\boldsymbol{R}_i \operatorname{Exp}\left(\delta \boldsymbol{\phi}_i\right)\right) &= \left(\boldsymbol{R}_i \operatorname{Exp}\left(\Delta \boldsymbol{\phi}_i\right)\right)^{\mathrm{T}}\left(\boldsymbol{v}_j - \boldsymbol{v}_i - \boldsymbol{g} \Delta t_{ij}\right) - \Delta \tilde{\boldsymbol{v}}_{ij} \\
&= \left(\boldsymbol{I} - \Delta \boldsymbol{\phi}_i^{\wedge}\right) \boldsymbol{R}_i^{\mathrm{T}}\left(\boldsymbol{v}_j - \boldsymbol{v}_i - \boldsymbol{g} \Delta t_{ij}\right) - \Delta \tilde{\boldsymbol{v}}_{ij} \\
&= \boldsymbol{r}_{\Delta \boldsymbol{v}_{ij}}\left(\boldsymbol{R}_i\right) + \left(\boldsymbol{R}_i^{\mathrm{T}}\left(\boldsymbol{v}_j - \boldsymbol{v}_i - \boldsymbol{g} \Delta t_{ij}\right)\right)^{\wedge} \Delta \boldsymbol{\phi}_i
\end{aligned}
$$

速度残差相对 $b_{g,i}, b_{a,i}$ 的雅可比只和 $\Delta \tilde{\boldsymbol{v}}_{ij}$ 有关，由于速度的残差项与它只相差一个负号，所以我们只需要在前面推导的零偏雅可比的前面添加一个负号即可

平移部分：

$$\frac{\partial \boldsymbol{r}_{\Delta \boldsymbol{p}_{ij}}}{\partial \boldsymbol{p}_i} = -\boldsymbol{R}_i^{\mathrm{T}}$$
$$\frac{\partial \boldsymbol{r}_{\Delta \boldsymbol{p}_{ij}}}{\partial \boldsymbol{p}_j} = \boldsymbol{R}_i^{\mathrm{T}}$$
$$\frac{\partial \boldsymbol{r}_{\Delta \boldsymbol{p}_{ij}}}{\partial \boldsymbol{v}_i} = -\boldsymbol{R}_i^{\mathrm{T}} \Delta t_{ij}$$
$$\frac{\partial \boldsymbol{r}_{\Delta \boldsymbol{p}_{ij}}}{\partial \boldsymbol{\phi}_i} = \left(\boldsymbol{R}_i^{\mathrm{T}} \left(\boldsymbol{p}_j - \boldsymbol{p}_i - \boldsymbol{v}_i \Delta t_{ij} - \frac{1}{2} \boldsymbol{g} \Delta t_{ij}^2 \right) \right)^{\wedge}$$

零偏的残差也只需在零偏雅可比基础上添加负号即可

至此，我们推导了预积分观测量对所有状态变量的导数形式。如果我们愿意，也可以将预积分观测写成一列，将状态变量写成一列，然后把这些雅可比矩阵都写在一起即可

3. VIO部分预积分

vins-mono 下预积分和上述公式大同小异，最主要的区别在于其用四元数来表达旋转

理论部分：

对于旋转部分：

$$q_{b_{k+1}}^{w} = q_{b_k}^{w} \bigotimes \int_{t \in [k, k+1]} \frac{1}{2} \Omega \left(\widehat{\omega}_t - b_{\omega_t} \right) q_t^{b_k} dt$$

证明：

公式(1)的 IMU 连续形式下的旋转状态推导如下，首先可写成：

$$q_{b_{k+1}}^{w} = q_{b_k}^{w} \otimes \int_{t\in[k,k+1]} \dot{q}_t dt \tag{A1}$$

根据《视觉 SLAM 十四讲》3.4.2 的四元数乘法，我们引入左乘和右乘符号如下：

$$\begin{aligned} q_a \otimes q_b = \mathcal{R}(q_b)q_a &= \begin{bmatrix} s_b & z_b & -y_b & x_b \\ -z_b & s_b & x_b & y_b \\ y_b & -x_b & s_b & z_b \\ -x_b & -y_b & -z_b & s_b \end{bmatrix} \begin{bmatrix} x_a \\ y_a \\ z_a \\ s_a \end{bmatrix} \\ &= \mathcal{L}(q_a)q_b = \begin{bmatrix} s_a & -z_a & y_a & x_a \\ z_a & s_a & -x_a & y_a \\ -y_a & x_a & s_a & z_a \\ -x_a & -y_a & -z_a & s_a \end{bmatrix} \begin{bmatrix} x_b \\ y_b \\ z_b \\ s_b \end{bmatrix} \end{aligned} \tag{A2}$$

注意上式中的形式与文献[2]的式(10)是不相同的，这是因为两者的左右手系不同。

为了简化，令$q = [x\ y\ z\ s] = [\omega\ s]$，则有：

$$\begin{aligned} \mathcal{R}(q) &= \Omega(\omega) + sI_{4\times4} = \begin{bmatrix} -\omega^{\wedge} & \omega \\ -\omega^{T} & 0 \end{bmatrix} + sI_{4\times4} \\ \mathcal{L}(q) &= \Psi(\omega) + sI_{4\times4} = \begin{bmatrix} \omega^{\wedge} & \omega \\ -\omega^{T} & 0 \end{bmatrix} + sI_{4\times4} \end{aligned} \tag{A3}$$

其中根据论文[2]公式(86)可推导四元数的导数如下：

$$\begin{aligned} \dot{q}_t &= \lim_{\delta t\to 0} \frac{1}{\delta t}(q_{t+\delta t} - q_t) \\ &= \lim_{\delta t\to 0} \frac{1}{\delta t}\left(q_t \otimes q_{t+\delta t}^{t} - q_t \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix}\right) \\ &= \lim_{\delta t\to 0} \frac{1}{\delta t}\left(q_t \otimes \begin{bmatrix} \widehat{\boldsymbol{k}} \sin\frac{\theta}{2} \\ \cos\frac{\theta}{2} \end{bmatrix} - q_t \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix}\right) \\ &\approx \lim_{\delta t\to 0} \frac{1}{\delta t}\left(q_t \otimes \begin{bmatrix} \widehat{\boldsymbol{k}} \frac{\theta}{2} \\ 1 \end{bmatrix} - q_t \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix}\right) \end{aligned}$$

$$= \lim_{\delta t \to 0} \frac{1}{\delta t} \left[\mathcal{R}\left(\begin{bmatrix} \widehat{\boldsymbol{k}} \frac{\theta}{2} \\ 1 \end{bmatrix} \right) - \mathcal{R}\left(\begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) \right] q_t$$

$$= \lim_{\delta t \to 0} \frac{1}{\delta t} \begin{bmatrix} -\frac{\theta^\wedge}{2} & \frac{\theta}{2} \\ -\frac{\theta^T}{2} & 0 \end{bmatrix} q_t$$

又有角速度为：$\omega = \lim_{\delta t \to 0} \frac{\theta}{\delta t}$，则上式可化为：

$$\dot{q}_t = \frac{1}{2} \begin{bmatrix} -\omega^\wedge & \omega \\ -\omega^T & 0 \end{bmatrix} q_t = \frac{1}{2} \Omega(\omega) q_t = \frac{1}{2} \mathcal{R}\left(\begin{bmatrix} \omega \\ 0 \end{bmatrix} \right) q_t = \frac{1}{2} q_t \otimes \begin{bmatrix} \omega \\ 0 \end{bmatrix} \tag{A4}$$

代码部分：

- predict()

```
// 中值积分计算PVQ
    Eigen::Vector3d un_acc_0 = tmp_Q * (acc_0 - tmp_Ba) - estimator.g;

    Eigen::Vector3d un_gyr = 0.5 * (gyr_0 + angular_velocity) - tmp_Bg;
    tmp_Q = tmp_Q * Utility::deltaQ(un_gyr * dt);

    Eigen::Vector3d un_acc_1 = tmp_Q * (linear_acceleration - tmp_Ba) - estimator.g;

    Eigen::Vector3d un_acc = 0.5 * (un_acc_0 + un_acc_1);

    tmp_P = tmp_P + dt * tmp_V + 0.5 * dt * dt * un_acc;
    tmp_V = tmp_V + dt * un_acc;

    acc_0 = linear_acceleration;
    gyr_0 = angular_velocity;
```

$$\begin{aligned} p_{b_{i+1}}^w &= p_{b_i}^w + v_{b_i}^w \delta t + \frac{1}{2} \bar{\hat{a}}_i \delta t^2 \\ v_{b_{i+1}}^w &= v_{b_i}^w + \bar{\hat{a}}_i \delta t \\ q_{b_{i+1}}^w &= q_{b_i}^w \otimes \begin{bmatrix} 1 \\ \frac{1}{2} \bar{\hat{\omega}}_i \delta t \end{bmatrix} \end{aligned} \tag{2}$$

其中：

$$\bar{\hat{a}}_i = \frac{1}{2} \left[q_i (\hat{a}_i - b_{a_i}) - g^w + q_{i+1} (\hat{a}_{i+1} - b_{a_i}) - g^w \right] \quad \leftarrow \text{中值法}$$

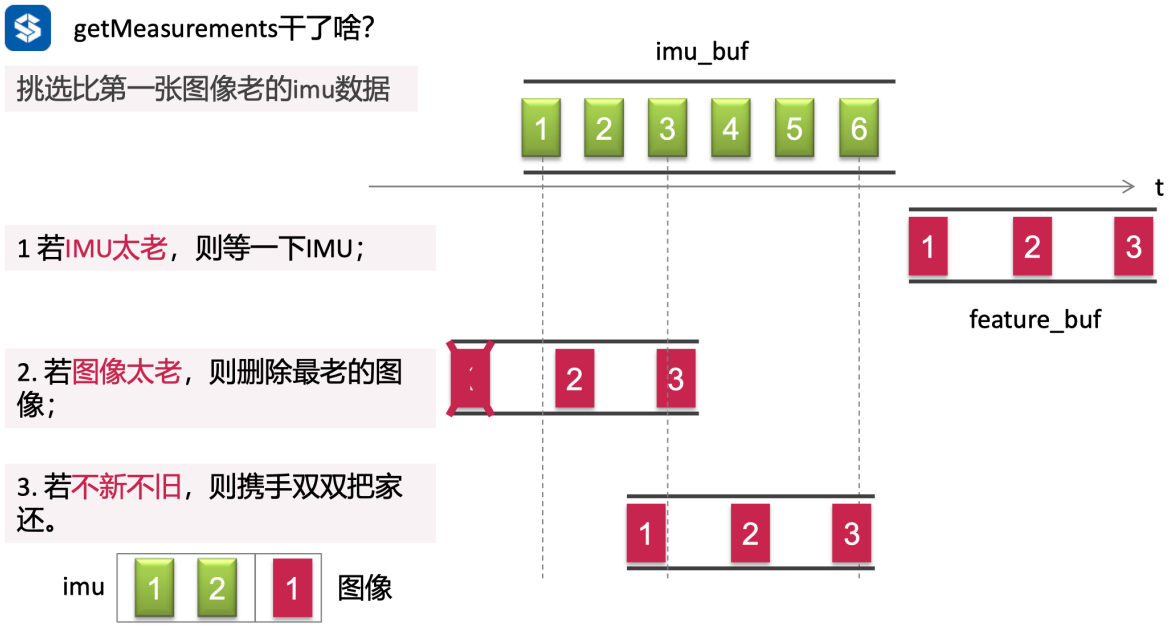
$$\bar{\hat{\omega}}_i = \frac{1}{2} (\hat{\omega}_i + \hat{\omega}_{i+1}) - b_{\omega_i} \tag{3}$$

- getMeasurements()

```
while (ros::ok())
    {
        if (imu_buf.empty() || featurebuf.empty())
            return measurements;
        //
        if (!(imu_buf.back()->header.stamp.toSec() > feature_buf.front()->header.stamp.toSec() + estimator.td))
        {
            return measurements;
        }

        if (!(imu_buf.front()->header.stamp.toSec() < feature_buf.front()->header.stamp.toSec() + estimator.td))
        {
            ROS_WARN("throw img, only should happen at the beginning");
            feature_buf.pop();
            continue;
        }
        sensor_msgs::PointCloudConstPtr img_msg = feature_buf.front();
        feature_buf.pop();

        std::vector<sensor_msgs::ImuConstPtr> IMUs;
        while (imu_buf.front()->header.stamp.toSec() < img_msg->header.stamp.toSec() + estimator.td)
        {
            IMUs.emplace_back(imu_buf.front());
            imu_buf.pop();
        }
        IMUs.emplace_back(imu_buf.front());
        if (IMUs.empty())
            ROS_WARN("no imu between two image");
        measurements.emplace_back(IMUs, img_msg);
    }
```



- midPointIntegration()

  1.两帧之间PVQ增量的计算

```
// 中值积分计算PVQ
        Vector3d un_acc_0 = delta_q * (_acc_0 - linearized_ba);
        Vector3d un_gyr = 0.5 * (_gyr_0 + _gyr_1) - linearized_bg;     // w
```

```
    result_delta_q = delta_q * Quaterniond(1, un_gyr(0) * _dt / 2, un_gyr(1) * _dt / 2, un_gyr(2) * _dt / 2); // Q
    Vector3d un_acc_1 = result_delta_q * (_acc_1 - linearized_ba);
    Vector3d un_acc = 0.5 * (un_acc_0 + un_acc_1);                 // a
    result_delta_p = delta_p + delta_v * _dt + 0.5 * un_acc * _dt * _dt; // P
    result_delta_v = delta_v + un_acc * _dt;                           // V
    result_linearized_ba = linearized_ba;
    result_linearized_bg = linearized_bg;
```

$$\hat{\alpha}_{i+1}^{b_k} = \hat{\alpha}_i^{b_k} + \hat{\beta}_i^{b_k}\delta t + \frac{1}{2}\bar{\hat{a}}_i^{'}\delta t^2$$

$$\hat{\beta}_{i+1}^{b_k} = \hat{\beta}_i^{b_k} + \bar{\hat{a}}_i^{'}\delta t \tag{8}$$

$$\hat{\gamma}_{i+1}^{b_k} = \hat{\gamma}_i^{b_k} \otimes \hat{\gamma}_{i+1}^{i} = \hat{\gamma}_i^{b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\bar{\hat{\omega}}_i^{'}\delta t \end{bmatrix}$$

其中，

$$\bar{\hat{a}}_i^{'} = \frac{1}{2}\left[q_i\left(\hat{a}_i - b_{a_i}\right) + q_{i+1}\left(\hat{a}_{i+1} - b_{a_i}\right)\right]$$

$$\bar{\hat{\omega}}_i^{'} = \frac{1}{2}\left(\hat{\omega}_i + \hat{\omega}_{i+1}\right) - b_{\omega_i} \tag{9}$$

2.PVQ增量误差的Jacobian和协方差的更新

```
if(update_jacobian)
        {
            Vector3d w_x = 0.5 * (_gyr_0 + _gyr_1) - linearized_bg;
            Vector3d a_0_x = _acc_0 - linearized_ba;
            Vector3d a_1_x = _acc_1 - linearized_ba;
            Matrix3d R_w_x, R_a_0_x, R_a_1_x;

            //反对称矩阵
            R_w_x<<0, -w_x(2), w_x(1),
                w_x(2), 0, -w_x(0),
                -w_x(1), w_x(0), 0;
            R_a_0_x<<0, -a_0_x(2), a_0_x(1),
                a_0_x(2), 0, -a_0_x(0),
                -a_0_x(1), a_0_x(0), 0;
            R_a_1_x<<0, -a_1_x(2), a_1_x(1),
                a_1_x(2), 0, -a_1_x(0),
                -a_1_x(1), a_1_x(0), 0;

            MatrixXd F = MatrixXd::Zero(15, 15);
            F.block<3, 3>(0, 0) = Matrix3d::Identity();
            F.block<3, 3>(0, 3) = -0.25 * delta_q.toRotationMatrix() * R_a_0_x * _dt * _dt +
                                  -0.25 * result_delta_q.toRotationMatrix() * R_a_1_x * (Matrix3d::Identity() - R_w_x * _dt) * _dt
            F.block<3, 3>(0, 6) = MatrixXd::Identity(3,3) * _dt;
            F.block<3, 3>(0, 9) = -0.25 * (delta_q.toRotationMatrix() + result_delta_q.toRotationMatrix()) * _dt * _dt;
            F.block<3, 3>(0, 12) = -0.25 * result_delta_q.toRotationMatrix() * R_a_1_x * _dt * _dt * -_dt;
            F.block<3, 3>(3, 3) = Matrix3d::Identity() - R_w_x * _dt;
            F.block<3, 3>(3, 12) = -1.0 * MatrixXd::Identity(3,3) * _dt;
            F.block<3, 3>(6, 3) = -0.5 * delta_q.toRotationMatrix() * R_a_0_x * _dt +
                                  -0.5 * result_delta_q.toRotationMatrix() * R_a_1_x * (Matrix3d::Identity() - R_w_x * _dt) * _dt;
            F.block<3, 3>(6, 6) = Matrix3d::Identity();
            F.block<3, 3>(6, 9) = -0.5 * (delta_q.toRotationMatrix() + result_delta_q.toRotationMatrix()) * _dt;
            F.block<3, 3>(6, 12) = -0.5 * result_delta_q.toRotationMatrix() * R_a_1_x * _dt * -_dt;
            F.block<3, 3>(9, 9) = Matrix3d::Identity();
            F.block<3, 3>(12, 12) = Matrix3d::Identity();
            //cout<<"A"<<endl<<A<<endl;

            MatrixXd V = MatrixXd::Zero(15,18);
            V.block<3, 3>(0, 0) =  0.25 * delta_q.toRotationMatrix() * _dt * _dt;
            V.block<3, 3>(0, 3) =  0.25 * -result_delta_q.toRotationMatrix() * R_a_1_x  * _dt * _dt * 0.5 * _dt;
            V.block<3, 3>(0, 6) =  0.25 * result_delta_q.toRotationMatrix() * _dt * _dt;
```

```
        V.block<3, 3>(0, 9) =  V.block<3, 3>(0, 3);
        V.block<3, 3>(3, 3) =  0.5 * MatrixXd::Identity(3,3) * _dt;
        V.block<3, 3>(3, 9) =  0.5 * MatrixXd::Identity(3,3) * _dt;
        V.block<3, 3>(6, 0) =  0.5 * delta_q.toRotationMatrix() * _dt;
        V.block<3, 3>(6, 3) =  0.5 * -result_delta_q.toRotationMatrix() * R_a_1_x  * _dt * 0.5 * _dt;
        V.block<3, 3>(6, 6) =  0.5 * result_delta_q.toRotationMatrix() * _dt;
        V.block<3, 3>(6, 9) =  V.block<3, 3>(6, 3);
        V.block<3, 3>(9, 12) = MatrixXd::Identity(3,3) * _dt;
        V.block<3, 3>(12, 15) = MatrixXd::Identity(3,3) * _dt;

        jacobian = F * jacobian;
        covariance = F * covariance * F.transpose() + V * noise * V.transpose();
    }
```

$$\begin{bmatrix}\delta\alpha_{k+1}\\ \delta\theta_{k+1}\\ \delta\beta_{k+1}\\ \delta b_{a_{k+1}}\\ \delta b_{w_{k+1}}\end{bmatrix}=\begin{bmatrix}I & f_{01} & \delta t & f_{03} & f_{04}\\ 0 & f_{11} & 0 & 0 & -\delta t\\ 0 & f_{21} & I & f_{23} & f_{24}\\ 0 & 0 & 0 & I & 0\\ 0 & 0 & 0 & 0 & I\end{bmatrix}\begin{bmatrix}\delta\alpha_k\\ \delta\theta_k\\ \delta\beta_k\\ \delta b_{a_k}\\ \delta b_{w_k}\end{bmatrix}$$

$$+\begin{bmatrix}v_{00} & v_{01} & v_{02} & v_{03} & 0 & 0\\ 0 & \frac{-\delta t}{2} & 0 & \frac{-\delta t}{2} & 0 & 0\\ -\frac{R_k\delta t}{2} & v_{21} & -\frac{R_{k+1}\delta t}{2} & v_{23} & 0 & 0\\ 0 & 0 & 0 & 0 & \delta t & 0\\ 0 & 0 & 0 & 0 & 0 & \delta t\end{bmatrix}\begin{bmatrix}n_{a_k}\\ n_{w_k}\\ n_{a_{k+1}}\\ n_{w_{k+1}}\\ n_{b_a}\\ n_{b_w}\end{bmatrix} \tag{15}$$

其中，推导可参考附录 10.3：

$$f_{01}=\frac{\delta t}{2}f_{21}=-\frac{1}{4}R_k\left(\hat{a}_k-b_{a_k}\right)^{\wedge}\delta t^2-\frac{1}{4}R_{k+1}\left(\hat{a}_{k+1}-b_{a_k}\right)^{\wedge}\left[I-\left(\frac{\hat{\omega}_k+\hat{\omega}_{k+1}}{2}-b_{\omega_k}\right)^{\wedge}\delta t\right]\delta t^2$$

$$f_{03}=-\frac{1}{4}(R_k+R_{k+1})\delta t^2$$

$$f_{04}=\frac{\delta t}{2}f_{24}=\frac{1}{4}R_{k+1}\left(\hat{a}_{k+1}-b_{a_k}\right)^{\wedge}\delta t^3$$

$$f_{11}=I-\left(\frac{\hat{\omega}_k+\hat{\omega}_{k+1}}{2}-b_{\omega_k}\right)^{\wedge}\delta t$$

$$f_{21}=-\frac{1}{2}R_k\left(\hat{a}_k-b_{a_k}\right)^{\wedge}\delta t-\frac{1}{2}R_{k+1}\left(\hat{a}_{k+1}-b_{a_k}\right)^{\wedge}\left[I-\left(\frac{\hat{\omega}_k+\hat{\omega}_{k+1}}{2}-b_{\omega_k}\right)^{\wedge}\delta t\right]\delta t$$

$$f_{23}=-\frac{1}{2}(R_k+R_{k+1})\delta t$$

$$f_{24}=\frac{1}{2}R_{k+1}\left(\hat{a}_{k+1}-b_{a_k}\right)^{\wedge}\delta t^2$$

$$v_{00}=-\frac{1}{4}R_k\delta t^2$$

$$v_{01}=v_{03}=\frac{\delta t}{2}v_{21}=\frac{1}{4}R_{k+1}\left(\hat{a}_{k+1}-b_{a_k}\right)^{\wedge}\delta t^2\frac{\delta t}{2}$$

$$v_{02}=-\frac{1}{4}R_{k+1}\delta t^2$$

$$v_{21}=v_{23}=\frac{1}{4}R_{k+1}\left(\hat{a}_{k+1}-b_{a_k}\right)^{\wedge}\delta t^2$$

将公式(15)简写为：

$$\delta z_{k+1}{}^{15\times1} = F^{15\times15}\delta z_k{}^{15\times1} + V^{15\times18}Q^{18\times1}$$

则 Jacobian 的迭代公式为：

$$J_{k+1}{}^{15\times15} = FJ_k \tag{16}$$

其中，Jacobian 的初始值为$J_k = I$。这里计算出来的J_{k+1}只是为了给后面提供对 bias 的 Jacobian。

协方差的迭代公式为：

$$P_{k+1}{}^{15\times15} = FP_kF^T + VQV^T \tag{17}$$

其中，初始值$P_k = 0$。Q 为表示噪声项的对角协方差矩阵：

$$Q^{18\times18} = \begin{bmatrix} \sigma_a^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_w^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_a^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_w^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{b_a}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{b_w}^2 \end{bmatrix} \tag{18}$$

4. LIO部分预积分

GTSAM基础：

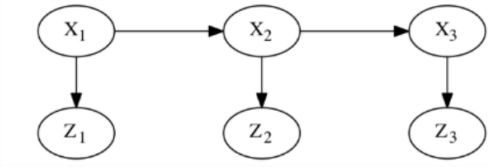


Figure 1: An HMM, unrolled over three time-steps, represented by a Bayes net.

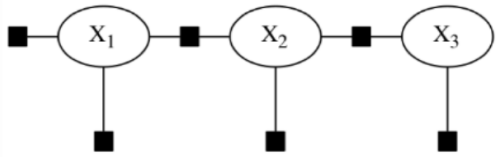
Figure 1 shows the **Bayes network** for a hidden Markov model (HMM) over three time steps

In a Bayes net, each node is associated with a conditional density

- 顶层马尔科夫链展示了先验概率$P(x_1)$ 和 转移概率 $P(X_2|X_1)$, $P(X_3|X_2)$
- 每个状态的观测量 Z 仅与当前时刻的状态X 有关,服从条件概率密度$P(Z_t|X_t)$
- 给定一组测量值z_1, z_2, z_3 ,可以通过最大后验概率来得到状态X_1, X_2, X_3 即求 $P(X_1, X_2, X_3|Z_1 = z_1, Z_2 = z_2, Z_3 = z_3)$

根据贝叶斯定理,后验概率可以拆分为六个因子的乘积(三个先验,三个似然),即如下图所示

$$P(X_1, X_2, X_3|Z_1, Z_2, Z_3) \propto P(X_1)P(X_2|X_1)P(X_3|X_2)L(X_1; z_1)L(X_2; z_2)L(X_3; z_3)$$

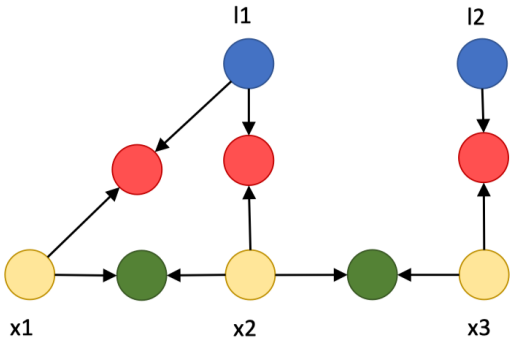


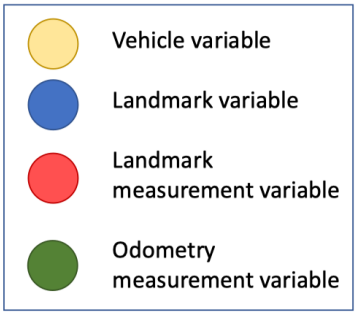
To do maximum a-posteriori (MAP) inference, we then maximize the product

$$f\left(X_1, X_2, X_3\right) = \prod f_i\left(\mathscr{X}_i\right)$$

对于一个简单的SLAM问题来说：

Modeling SLAM by Bayes Net





$$P(X, Z) = P(Z|X)P(X)$$

$$P(Z|X) = \prod_i P_i(Z_i|X_i)$$

$$\mathrm{P}(z_{11}, z_{12}, z_{13}, z_{21}, z_{22}|x_1, x_2, x_3, l_1, l_2) =$$

$$P(z_{11}|x_1, l_1)P(z_{12}|x_2, l_1)P(z_{13}|x_3, l_2)P(z_{21}|x_1, x_2)P(z_{22}|x_2, x_3)$$

Bayes Net ⟺ Factor Graph

- Bayes net: generative model vs. Factor graph: inference model
- Apply Bayes rule:

$$P(X|Z) = \frac{P(Z|X)P(X)}{P(Z)} \propto P(Z|X)P(X)$$

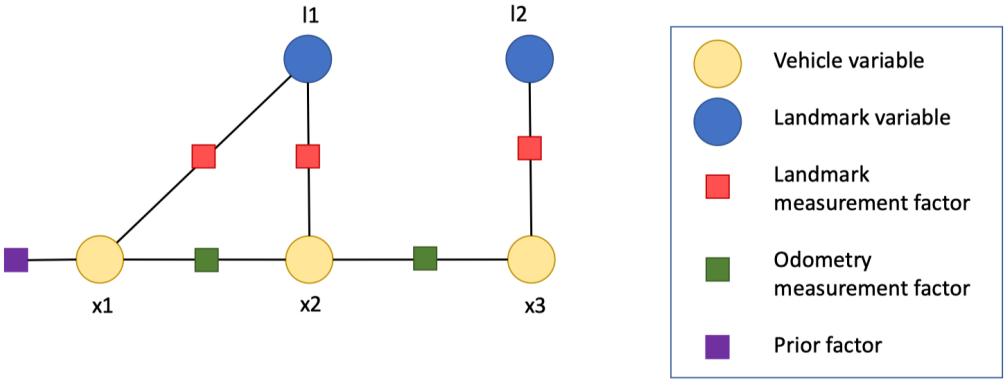
$$X^* = \arg\max_X P(X|Z) = \arg\max_X P(Z|X)P(X)$$

- MAP (maximum posterior inference)
- Prior/likelihood terms factorized as factors

$$\phi(X_i) \doteq \psi(X_i, Z_i) \propto P_i(Z_i|X_i)$$

$$X^* = \arg\max_X \phi(X) = \arg\max_X \prod_i \phi_i(x_i).$$

Modeling a SLAM Problem by a Factor Graph



$$\phi(X) \doteq \phi(x_1, l_1)\phi(x_2, l_1)\phi(x_3, l_2)\phi(x_1, x_2)\phi(x_2, x_3)\phi(x_1)$$

$$X^* = \arg\max_X \phi(X) = \arg\max_X \prod_i \phi_i(X_i)$$

[1] Frank Dellaert and Michael Kaess. "Factor graphs for robot perception."

Defining Factors: SLAM Example

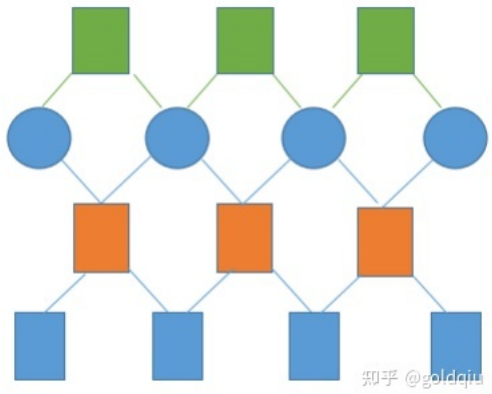
- A factor has an exponential form

$$\phi_i(X_i) = \exp\Big(-\frac{1}{2} \parallel f_i(X_i) \parallel^2_{\Sigma_i}\Big)$$

- A factor has an *error function* of variables, and a *measurement*
  - Prior factor: $f_{\text{prior}}(x_i) = x_i - z$
  - Odometry factor: $f_{\text{odometry}}(x_i, x_{i+1}) = (x_{i+1} - x_i) - z$
  - Landmark factor: $f_{\text{landmark}}(x_i, l_j) = \text{projection}(x_i, l_j) - z$

$$X^* = \arg\max_X \phi(X) = \arg\max_X \prod_i \phi_i(X_i)$$

代码部分：



蓝色圆圈代表关键帧位姿，蓝色矩形代表关键帧速度和零偏，橙色矩形代表IMU预积分约束，它可以约束相邻帧的位姿、速度和零偏，绿色矩形代表lidar里程记的帧间约束，其约束相邻两帧的位置和姿态。这里包括了lidar里程计因子和预积分因子，是预积分节点因子图的优化模型。

GTSAM 中跟 IMU 预积分相关的接口定义

```
gtsam::PreintegrationParams
```

预积分相关参数，我们对 IMU 数据进行预积分之前通常需要事先知道 IMU 的噪声，重力方向等参数

```
gtsam::PreintegratedImuMeasurements
```

跟预积分相关的计算就在这个类中实现

这个类有一些重要的接口

```
(1) resetIntegrationAndSetBias
```

将预积分量复位，也就是说清空刚刚的预积分量，重新开始一个新的预积分量，

比如刚刚计算了一个第一帧到第二帧的预积分，然后不想重新创建一个预积分类，复用之前的对象，就需要reset

注意：预积分的计算依赖一个初始的 IMU 零偏，因此，在复位之后需要输入零偏值，所以这里复位和重设零偏在一个接口里。

```
(2) integrateMeasurement
```

输入 IMU 的测量值，其内部会自动实现预积分量的更新以及协方差矩阵的更新

```
(3) deltaTij
```

预积分量跨越的时间长度

```
(4) predict
```

使用之前的状态和零偏，预测现在的状态。

预积分量可以计算出两帧之间的相对位置、速度、姿态的变化量，那结合上一帧的状态量就可以计算出下一关键帧根据预积分结果的推算值

该模块涉及到的变量结点

```
gtsam::Pose3 // 表示六自由度位姿
gtsam::Vector3 // 表示三自由度速度
gtsam::imuBias::ConstantBias //表示 IMU 零偏
```

以上也是预积分模型中涉及到的三种状态变量

涉及到的因子结点

```
gtsam::PriorFactor<T>
```

先验因子，表示对某个状态量 T 的一个先验估计，约束某个状态变量的状态不会离该先验值过远

```
gtsam::ImuFactor
```

imu 因子，通过 IMU 预积分量构造出 IMU 因子，即 IMU 约束，相当于一个帧间约束

```
gtsam::BetweenFactor
```

状态量间的约束，约束相邻两状态量之间的差值不会距离该约束过远

odometryHandler()

odometry的回调函数是最重要的回调函数，步骤为

1. 取出里程计的位置和方向
2. 这帧雷达的pose转为gtsam
3. 第一次进入，初始化系统：
   3.1 GTSAM初始化
   3.2 添加先验约束
   3.3 添加实际的状态量
   3.4 更新isam优化器
   3.5 预积分接口重置
4. 每隔100帧激光里程计，重置ISAM2优化器，保证优化效率

5. 加入imu数据，自动实现预积分量的更新以及协方差矩阵的更新
6. 添加ImuFactor，零偏约束
7. 添加雷达的位姿约束
8. 失败检测（速度大于30，零偏太大）
9. 根据最新的imu状态进行传播

参考文献：

[1] imu预积分原理的个人理解 - 知乎 (zhihu.com)

[2] 简明预积分推导 - 知乎 (zhihu.com)

[3] VINS论文推导及代码解析 — 崔华坤

[4] VINS-Mono理论学习——IMU预积分 Pre-integration （Jacobian 协方差）_Manii的博客-CSDN博客

[5] 深蓝学院 — VINS-Mono代码讲解 （崔华坤）

[6] Factor Graphs and GTSAM | GTSAM

[7] GTSAM 官方教程学习_说海似云的博客-CSDN博客_gtsam

[8] 深蓝学院 — 因子图的理论基础 —董靖

[9] 十七.激光和惯导LIO-SLAM框架学习之IMU和IMU预积分 - 知乎 (zhihu.com)

[10] LIO-SAM源码解析(三)：IMUPreintegration - 知乎 (zhihu.com)

[11] 因子图及GTSAM库介绍 | 大杂烩 (chargerkong.github.io)