深蓝学院
shenlanxueyuan.com

第十二章作业提示

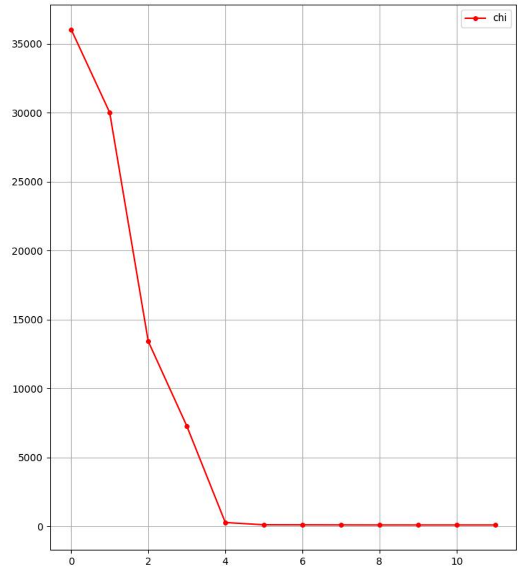
主讲人 会打篮球的猫

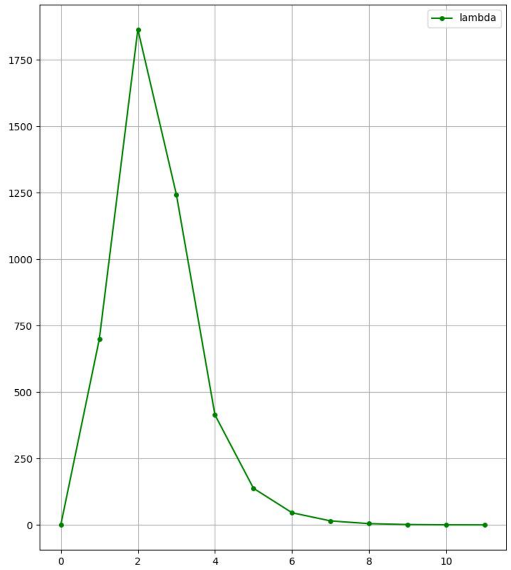
第一题

1 样例代码给出了使用 LM 算法来估计曲线 $y = \exp(ax^2 + bx + c)$ 参数 a, b, c 的完整过程。

❶ 请绘制样例代码中 LM 阻尼因子 μ 随着迭代变化的曲线图

```
Test CurveFitting start...
iter: 0 , chi= 36048.3 , Lambda= 0.001
iter: 1 , chi= 30015.5 , Lambda= 699.051
iter: 2 , chi= 13421.2 , Lambda= 1864.14
iter: 3 , chi= 7273.96 , Lambda= 1242.76
iter: 4 , chi= 269.255 , Lambda= 414.252
iter: 5 , chi= 105.473 , Lambda= 138.084
iter: 6 , chi= 100.845 , Lambda= 46.028
iter: 7 , chi= 95.9439 , Lambda= 15.3427
iter: 8 , chi= 92.3017 , Lambda= 5.11423
iter: 9 , chi= 91.442 , Lambda= 1.70474
iter: 10 , chi= 91.3963 , Lambda= 0.568247
iter: 11 , chi= 91.3959 , Lambda= 0.378832
problem solve cost: 0.581061 ms
   makeHessian cost: 0.349348 ms
LM results have been saved at: /home/lcx/lm_results.txt
-------After optimization, we got these parameters :
0.941939  2.09453 0.965586
-------ground truth:
1.0,  2.0,  1.0
```





第一题

❷ 将曲线函数改成 $y = ax^2 + bx + c$，请修改样例代码中残差计算，雅克比计算等函数，完成曲线参数估计。

- 生成数据
- 残差
- 雅克比

```cpp
    // 构造 N 次观测
    for (int i = 0; i < N; ++i) {

        double x = i/1000.;
        double n = noise(generator);
        // 观测 y
        // double y = std::exp( a*x*x + b*x + c ) + n;
        double y = a*x*x + b*x + c + n;
```

```cpp
    // 计算曲线模型误差
    virtual void ComputeResidual() override
    {
        Vec3 abc = verticies_[0]->Parameters();  // 估计的参数
        // residual_(0) = std::exp( abc(0)*x_*x_ + abc(1)*x_ + abc(2) ) - y_;  // 构建残差
        residual_(0) = abc(0)*x_*x_ + abc(1)*x_ + abc(2) - y_;  // 构建残差
    }

    // 计算残差对变量的雅克比
    virtual void ComputeJacobians() override
    {
        // Vec3 abc = verticies_[0]->Parameters();
        // double exp_y = std::exp( abc(0)*x_*x_ + abc(1)*x_ + abc(2) );

        Eigen::Matrix<double, 1, 3> jaco_abc;  // 误差为1维，状态量 3 个，所以是 1x3 的雅克比矩阵
        // jaco_abc << x_ * x_ * exp_y, x_ * exp_y , 1 * exp_y;
        jaco_abc << x_ * x_ , x_ , 1;
        jacobians_[0] = jaco_abc;
    }
```

第一题

深蓝学院 shenlanxueyuan.com

❷ 将曲线函数改成 $y = ax^2 + bx + c$，请修改样例代码中残差计算，雅克比计算等函数，完成曲线参数估计。

（观测数量的讨论）

```
lcx@lcx:~/Desktop/vio_homework/ch3/CurveFitting_LM/build$ ./

Test CurveFitting start...
iter: 0 , chi= 719.475 , Lambda= 0.001
iter: 1 , chi= 91.395 , Lambda= 0.000333333
problem solve cost: 0.097857 ms
   makeHessian cost: 0.055884 ms
LM results have been saved at: /home/lcx/lm_results.txt
-------After optimization, we got these parameters :
 1.61039  1.61853 0.995178
-------ground truth:
1.0,  2.0,  1.0
```

可以看出，最终的优化结果为（1.61039　1.61853 0.995178），与真值（1.0　2.0　1.0）相比，误差比较大，优化结果并不好。

因此，考虑提高观测数据的数量，由原先的 100 个数据提高到 1000 个，且依旧保证 x 在（0~1）范围内均分。

```
lcx@lcx:~/Desktop/vio_homework/ch3/CurveFitting_LM/build$ ./a

Test CurveFitting start...
iter: 0 , chi= 7114.25 , Lambda= 0.01
iter: 1 , chi= 973.88 , Lambda= 0.00333333
iter: 2 , chi= 973.88 , Lambda= 0.00222222
problem solve cost: 2.77373 ms
   makeHessian cost: 2.17039 ms
LM results have been saved at: /home/lcx/lm_results.txt
-------After optimization, we got these parameters :
0.958923  2.06283 0.968821
-------ground truth:
1.0,  2.0,  1.0
```

此时，可以看出使用 1000 个数据后，估计的结果为（0.958923　2.06283　0.968821），与真值相比已经非常接近了，估计结果较准。

为了做进一步验证，将数据增加至 2000 个，结果如下图所示：

```
lcx@lcx:~/Desktop/vio_homework/ch3/CurveFitting_LM/build$ ./a

Test CurveFitting start...
iter: 0 , chi= 14374.2 , Lambda= 0.02
iter: 1 , chi= 1979.22 , Lambda= 0.00666667
iter: 2 , chi= 1979.22 , Lambda= 0.00444444
problem solve cost: 1.78892 ms
   makeHessian cost: 1.46041 ms
LM results have been saved at: /home/lcx/lm_results.txt
-------After optimization, we got these parameters :
 1.07765  1.97101 0.983445
-------ground truth:
1.0,  2.0,  1.0
```

此时，发现结果变化已经微乎其微，即增加更多的观测对估计结果的精度已经不能再有显著提升。

第一题

深蓝学院
shenlanxueyuan.com

② **将曲线函数改成 $y = ax^2 + bx + c$，请修改样例代码中残差计算，雅克比计算等函数，完成曲线参数估计。** （初值的讨论）

再次回到 1000 个点，这次考虑将初值(0.0　0.0　0.0)修改为(0.9　2.1　0.9)，即给待估计参数一个较好的初值，再次运行结果如下：

```
lcx@lcx:~/Desktop/vio_homework/ch3/CurveFitting_LM/build$ ./app/testCurveFitting

Test CurveFitting start...
iter: 0 , chi= 978.82 , Lambda= 0.01
iter: 1 , chi= 973.88 , Lambda= 0.00333333
problem solve cost: 0.641336 ms
   makeHessian cost: 0.500614 ms
LM results have been saved at: /home/lcx/lm_results.txt
-------After optimization, we got these parameters :
0.958728  2.06304 0.968784
-------ground truth:
1.0,  2.0,  1.0
```

与上上个 1000 个点的结果图对比，发现参数估计结果在精度上并没有太大变化，推测原因是：由于我们拟合的曲线函数较简单，且只有一个极小值点，因此初值的选择并不会使得优化至错误的极小值，所以最终的精度更多地取决于观测数据的噪声。但是，较好地初值带来的好处便是，迭代次数的减少、耗时的减少，能够更快地收敛。

当然了，在后续面对真正复杂的函数时，提供较好的初值还是很有必要的。

第一题

深蓝学院 shenlanxueyuan.com

❸ 实现其他更优秀的阻尼因子策略，并给出实验对比（选做，评优秀），策略可参考论文[a] 4.1.1 节。

1. $\lambda_0 = \lambda_o$; λ_o is user-specified [5].
   use eq'n (13) for $\mathbf{h}_{\mathsf{lm}}$ and eq'n (16) for ρ
   if $\rho_i(\mathbf{h}) > \epsilon_4$: $\mathbf{p} \leftarrow \mathbf{p} + \mathbf{h}$; $\lambda_{i+1} = \max[\lambda_i / L_{\downarrow}, 10^{-7}]$;
   otherwise: $\lambda_{i+1} = \min[\lambda_i L_{\uparrow}, 10^7]$;

2. $\lambda_0 = \lambda_o \max\left[\mathsf{diag}[\mathbf{J}^\mathsf{T}\mathbf{W}\mathbf{J}]\right]$; λ_o is user-specified.
   use eq'n (12) for $\mathbf{h}_{\mathsf{lm}}$ and eq'n (15) for ρ
   $\alpha = \left(\left(\mathbf{J}^\mathsf{T}\mathbf{W}(\mathbf{y} - \hat{\mathbf{y}}(\mathbf{p}))\right)^\mathsf{T}\mathbf{h}\right) / \left(\left(\chi^2(\mathbf{p}+\mathbf{h}) - \chi^2(\mathbf{p})\right)/2 + 2\left(\mathbf{J}^\mathsf{T}\mathbf{W}(\mathbf{y} - \hat{\mathbf{y}}(\mathbf{p}))\right)^\mathsf{T}\mathbf{h}\right)$;
   if $\rho_i(\alpha\mathbf{h}) > \epsilon_4$: $\mathbf{p} \leftarrow \mathbf{p} + \alpha\mathbf{h}$; $\lambda_{i+1} = \max[\lambda_i/(1+\alpha), 10^{-7}]$;
   otherwise: $\lambda_{i+1} = \lambda_i + |\chi^2(\mathbf{p}+\alpha\mathbf{h}) - \chi^2(\mathbf{p})|/(2\alpha)$;

3. $\lambda_0 = \lambda_o \max\left[\mathsf{diag}[\mathbf{J}^\mathsf{T}\mathbf{W}\mathbf{J}]\right]$; λ_o is user-specified [6].
   use eq'n (12) for $\mathbf{h}_{\mathsf{lm}}$ and eq'n (15) for ρ
   if $\rho_i(\mathbf{h}) > \epsilon_4$: $\mathbf{p} \leftarrow \mathbf{p} + \mathbf{h}$; $\lambda_{i+1} = \lambda_i \max\left[1/3, 1 - (2\rho_i - 1)^3\right]$; $\nu_i = 2$;
   otherwise: $\lambda_{i+1} = \lambda_i \nu_i$; $\quad \nu_{i+1} = 2\nu_i$;

Nielsen策略（作业源码中实现的）

第一题

深蓝学院 shenlanxueyuan.com

❸ 实现其他更优秀的阻尼因子策略，并给出实验对比（选做，评优秀），策略可参考论文[a] 4.1.1 节。

$$\left[\mathbf{J}^{\top}\mathbf{W}\mathbf{J}+\lambda\mathbf{I}\right]\mathbf{h}_{\mathsf{lm}}=\mathbf{J}^{\top}\mathbf{W}(\mathbf{y}-\hat{\mathbf{y}})\,, \tag{12}$$

$$\left[\mathbf{J}^{\top}\mathbf{W}\mathbf{J}+\lambda\,\mathsf{diag}(\mathbf{J}^{\top}\mathbf{W}\mathbf{J})\right]\mathbf{h}_{\mathsf{lm}}=\mathbf{J}^{\top}\mathbf{W}(\mathbf{y}-\hat{\mathbf{y}})\,. \tag{13}$$

$$\rho_i(\mathbf{h}_{\mathsf{lm}}) = \frac{\chi^2(\mathbf{p})-\chi^2(\mathbf{p}+\mathbf{h}_{\mathsf{lm}})}{(\mathbf{y}-\hat{\mathbf{y}})^{\mathsf{T}}(\mathbf{y}-\hat{\mathbf{y}})-(\mathbf{y}-\hat{\mathbf{y}}-\mathbf{J}\mathbf{h}_{\mathsf{lm}})^{\mathsf{T}}(\mathbf{y}-\hat{\mathbf{y}}-\mathbf{J}\mathbf{h}_{\mathsf{lm}})} \tag{14}$$

$$= \frac{\chi^2(\mathbf{p})-\chi^2(\mathbf{p}+\mathbf{h}_{\mathsf{lm}})}{\mathbf{h}_{\mathsf{lm}}^{\mathsf{T}}\left(\lambda_i\mathbf{h}_{\mathsf{lm}}+\mathbf{J}^{\mathsf{T}}\mathbf{W}\left(\mathbf{y}-\hat{\mathbf{y}}(\mathbf{p})\right)\right)} \quad \text{if using eq'n (12) for } \mathbf{h}_{\mathsf{lm}} \tag{15}$$

$$= \frac{\chi^2(\mathbf{p})-\chi^2(\mathbf{p}+\mathbf{h}_{\mathsf{lm}})}{\mathbf{h}_{\mathsf{lm}}^{\mathsf{T}}\left(\lambda_i\mathsf{diag}(\mathbf{J}^{\mathsf{T}}\mathbf{W}\mathbf{J})\mathbf{h}_{\mathsf{lm}}+\mathbf{J}^{\mathsf{T}}\mathbf{W}\left(\mathbf{y}-\hat{\mathbf{y}}(\mathbf{p})\right)\right)} \quad \text{if using eq'n (13) for } \mathbf{h}_{\mathsf{lm}} \tag{16}$$

方法2、3

方法1

第一题

❸ 实现其他更优秀的阻尼因子策略，并给出实验对比（选做，评优秀），策略可参考论文[a] 4.1.1 节。

通过对比，主要的修改主要在 AddLambdatoHessianLM()、 RemoveLambdatoHessianLM() 和IsGoodStepInLM() 三个函数中。此外还有�初值的设定：ComputeLambdaInitLM()

```
void Problem::AddLambdatoHessianLM() {
    ulong size = Hessian_.cols();
    assert(Hessian_.rows() == Hessian_.cols() && "Hessian is not square");
    for (ulong i = 0; i < size; ++i) {
        // Hessian_(i, i) += currentLambda_;
        Hessian_(i, i) *= (currentLambda_ + 1.0);
    }
}

void Problem::RemoveLambdaHessianLM() {
    ulong size = Hessian_.cols();
    assert(Hessian_.rows() == Hessian_.cols() && "Hessian is not square");
    // TODO:: 这里不应该减去一个，数值的反复加减容易造成数值精度出问题？而应该保存叠加lambda前的值，在这里直接赋值
    for (ulong i = 0; i < size; ++i) {
        // Hessian_(i, i) -= currentLambda_;
        Hessian_(i, i) /= (currentLambda_ + 1.0);
    }
}
```

第一题

深蓝学院
shenlanxueyuan.com

❸ 实现其他更优秀的阻尼因子策略，并给出实验对比（选做，评优秀），策略可参考论文[a] 4.1.1 节。

```cpp
bool Problem::IsGoodStepInLM() {
    size_t size = Hessian_.rows();
    Eigen::MatrixXd diagH(size, size);
    diagH.setZero();
    for (size_t i = 0; i < size; ++i) {
        diagH(i, i) += Hessian_(i, i);
    }

    double scale = 0;
    scale = delta_x_.transpose() * (currentLambda_ * diagH * delta_x_ + b_);
    scale += 1e-3;    // make sure it's non-zero :)
```

```cpp
    double rho = (currentChi_ - tempChi) / scale;
    double upL = 11.0;
    double downL = 9.0;

    if (rho > 0 && isfinite(tempChi))   // last step was good, 误差在下降
    {
        // double alpha = 1. - pow((2 * rho - 1), 3);
        // alpha = std::min(alpha, 2. / 3.);
        // double scaleFactor = (std::max)(1. / 3., alpha);
        // currentLambda_ *= scaleFactor;
        // ni_ = 2;
        currentLambda_ = std::max(currentLambda_ / downL, 1e-7);
        currentChi_ = tempChi;
        return true;
    } else {
        // currentLambda_ *= ni_;
        // ni_ *= 2;
        currentLambda_ = std::min(currentLambda_ * upL, 1e7);
        return false;
    }
```

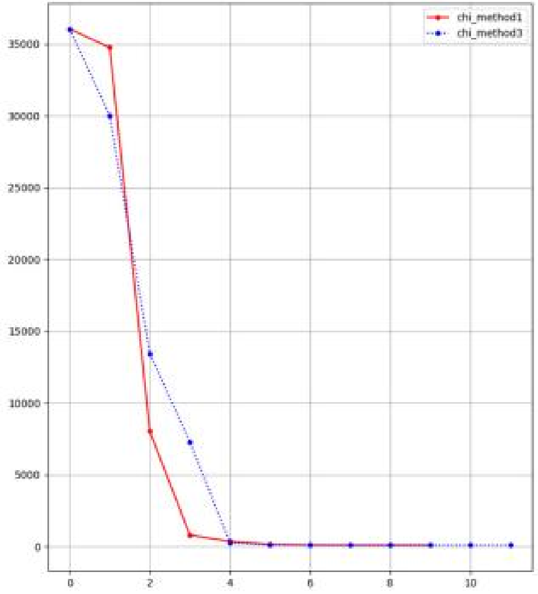
第一题

❸ 实现其他更优秀的阻尼因子策略，并给出实验对比（选做，评优秀），策略可参考论文[a] 4.1.1 节。

为了与Nielsen策略（第一问结果）进行有效对比，数据量不变，初值不变，函数依旧用exp的原函数。

```
lcx@lcx:~/Desktop/vio_homework/ch3/1_3_CurveFitting_LM/build$ ./app/testCurveFitting

Test CurveFitting start...
iter: 0 , chi= 36048.3 , Lambda= 0.001
iter: 1 , chi= 34760.2 , Lambda= 17.8946
iter: 2 , chi= 8020.58 , Lambda= 1.98828
iter: 3 , chi= 779.997 , Lambda= 0.22092
iter: 4 , chi= 348.805 , Lambda= 0.0245467
iter: 5 , chi= 145.33 , Lambda= 0.00272741
iter: 6 , chi= 101 , Lambda= 0.000303046
iter: 7 , chi= 92.3181 , Lambda= 3.36718e-05
iter: 8 , chi= 91.3999 , Lambda= 3.74131e-06
iter: 9 , chi= 91.3959 , Lambda= 4.15701e-07
problem solve cost: 0.44031 ms
   makeHessian cost: 0.276777 ms
LM results have been saved at: /home/lcx/lm_results.txt
-------After optimization, we got these parameters :
0.941955    2.0945    0.9656
-------ground truth:
1.0,  2.0,  1.0
```



两种策略最终估计的参数结果近似，且都接近真值。但方法一迭代次数更少，耗时更短。

第二题

1、推导 f15。

$$\mathbf{f}_{15} = \frac{\partial \boldsymbol{\alpha}_{b_i b_{k+1}}}{\partial \delta \mathbf{b}_k^g}$$

$$\boldsymbol{\omega} = \frac{1}{2}\left(\omega^{b_k} + \omega^{b_{k+1}}\right) - \mathbf{b}_k^g$$

$$\alpha_{b_j b_{k+1}} = \alpha_{b_i b_k} + \beta_{b_i b_k}\delta t + \frac{1}{2} a \delta t^2$$

$$a = \frac{1}{2}(R_{b_i b_k}(a^{b_k} - b_k^a) + R_{b_i b_k} \underline{exp([\omega \delta t]_\times)}(a^{b_{k+1}} - b_k^a))$$

$\alpha_{b_j b_{k+1}}$中只有a的ω与b_k^g有关，所以$\alpha_{b_j b_{k+1}}$关于b_k^g的偏导仅对相关项求导

问题：怎么加b_g扰动？？ b_g 扰动加在哪里？？

第二题

1、推导 f15。

错误示范！！！

$$\mathbf{f}_{15}=\frac{\partial \boldsymbol{\alpha}_{b_i b_{k+1}}}{\partial \delta \mathbf{b}_k^g}$$

$$=\frac{\partial \frac{1}{4}\mathbf{q}_{b_i b_k}\otimes\begin{bmatrix}1\\ \frac{1}{2}\boldsymbol{\omega}\delta t\end{bmatrix}\otimes\boxed{\begin{bmatrix}1\\ -\frac{1}{2}\delta\mathbf{b}_k^g\delta t\end{bmatrix}}\left(\mathbf{a}^{b_{k+1}}-\mathbf{b}_k^a\right)\delta t^2}{\partial \delta \mathbf{b}_k^g}$$

正确位置：

$$\mathbf{f}_{15}=\frac{\partial \boldsymbol{\alpha}_{b_i b_{k+1}}}{\partial \delta \mathbf{b}_k^g}=\frac{\partial \frac{1}{4}\mathbf{q}_{b_i b_k}\otimes\begin{bmatrix}1\\ \frac{1}{2}\left(\boldsymbol{\omega}\boxed{-\delta\mathbf{b}_k^g}\right)\delta t\end{bmatrix}\left(\mathbf{a}^{b_{k+1}}-\mathbf{b}_k^a\right)\delta t^2}{\partial \delta \mathbf{b}_k^g}$$

- 对谁加扰动，扰动项就直接跟在该变量后面
- 只有旋转变量加扰动才是乘一个微小旋转，其他均是加号！

$$\boldsymbol{\omega}=\frac{1}{2}\left(\omega^{b_k}+\omega^{b_{k+1}}\right)-\mathbf{b}_k^g$$

$$-\delta\mathbf{b}_k^g$$

$$\begin{bmatrix}1\\ a+b\end{bmatrix}\neq\begin{bmatrix}1\\ a\end{bmatrix}\otimes\begin{bmatrix}1\\ b\end{bmatrix}$$

第二题

深蓝学院 shenlanxueyuan.com

1、推导 f15。

$$
\begin{aligned}
\mathbf{f}_{15} = \frac{\partial \boldsymbol{\alpha}_{b_b b_{k+1}}}{\partial \delta \mathbf{b}_k^g} &= \frac{\partial \frac{1}{4}\mathbf{q}_{b_i b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\left(\boldsymbol{\omega} - \delta \mathbf{b}_k^g\right)\delta t \end{bmatrix} \left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\delta t^2}{\partial \delta \mathbf{b}_k^g} \\
&= \frac{1}{4}\frac{\partial \mathbf{R}_{b_k b_k} \exp\left(\left[\left(\omega - \delta \mathbf{b}_k^g\right)\delta t\right]_\times\right)\left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\delta t^2}{\partial \delta \mathbf{b}_k^g} \\
&\approx \frac{1}{4}\frac{\partial \mathbf{R}_{b_k b_k} \exp\left([\boldsymbol{\omega}\delta t]_\times\right)\exp\left(\left[-J_r(\boldsymbol{\omega}\delta t)\delta \mathbf{b}_k^g \delta t\right]_\times\right)\left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\delta t^2}{\partial \delta \mathbf{b}_k^g} \\
&= \frac{1}{4}\frac{\partial - \mathbf{R}_{b_i b_{k+1}}\left(\left[\left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\delta t^2\right]_\times\right)\left(-J_r(\boldsymbol{\omega}\delta t)\delta \mathbf{b}_k^g \delta t\right)}{\partial \delta \mathbf{b}_k^g} \\
&= -\frac{1}{4}\left(\mathbf{R}_{b_t b_{k+1}}\left[\left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\right]_\times \delta t^2\right)\left(-J_r(\boldsymbol{\omega}\delta t)\delta t\right) \\
&\approx -\frac{1}{4}\left(\mathbf{R}_{b_i b_{k+1}}\left[\left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\right]_\times \delta t^2\right)(-\delta t)
\end{aligned}
$$

$$
\begin{aligned}
f_{15} &= \frac{\partial \delta \alpha_{b_{k+1}}}{\partial \delta b_k^g} \\
&= \frac{\partial \frac{1}{4}(R_{b_i b_k} exp([(\omega - \delta b_k^g)\delta t]_\times)(a^{b_{k+1}} - b_k^a))\delta t^2}{\partial \delta b_k^g} \\
&= \frac{\partial \frac{1}{4}(R_{b_i b_k} exp([\omega\delta t]_\times)exp([-J_r(\omega\delta t)\delta b_k^g \delta t]_\times)(a^{b_{k+1}} - b_k^a))\delta t^2}{\partial \delta b_k^g} \\
&= \frac{\partial \frac{1}{4}(R_{b_i b_k} exp([\omega\delta t]_\times)(I + [-J_r(\omega\delta t)\delta b_k^g \delta t]_\times)(a^{b_{k+1}} - b_k^a))\delta t^2}{\partial \delta b_k^g} \\
&= \frac{-\partial \frac{1}{4}(R_{b_i b_k} exp([\omega\delta t]_\times)[a^{b_{k+1}} - b_k^a]_\times(-J_r(\omega\delta t)\delta b_k^g \delta t)\delta t^2}{\partial \delta b_k^g}
\end{aligned}
$$

当$\omega\delta t$极小时，$J_r(\omega\delta t) = I$，所以有

$$
\begin{aligned}
f_{15} &= \frac{\partial \delta \alpha_{b_{k+1}}}{\partial \delta b_k^g} \\
&= \frac{-\partial \frac{1}{4}(R_{b_i b_k} exp([\omega\delta t]_\times)[a^{b_{k+1}} - b_k^a]_\times(-J_r(\omega\delta t)\delta b_k^g \delta t)\delta t^2}{\partial \delta b_k^g} \\
&= -\frac{1}{4}R_{b_i b_{k+1}}[a^{b_{k+1}} - b_k^a]_\times \delta t^2(-\delta t)
\end{aligned}
$$

第二题

深蓝学院
shenlanxueyuan.com

1、推导 g12。

$$
\begin{aligned}
\mathbf{g}_{12} &= \frac{\partial \delta \boldsymbol{\beta}_{b_{k+1}}}{\partial \delta \mathbf{n}_k^g} \\
&= \frac{1}{4}\frac{\partial \mathbf{q}_{b_i b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\left(\boldsymbol{\omega} - \frac{1}{2}\delta \mathbf{n}_k^g\right)\delta t \end{bmatrix} \left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\delta t^2}{\partial \delta \mathbf{n}_k^g} \\
&= \frac{1}{4}\frac{\partial \mathbf{R}_{b_i b_k} \exp\left(\left[\left(\boldsymbol{\omega} - \frac{1}{2}\delta \mathbf{n}_k^g\right)\delta t\right]_\times\right)\left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\delta t^2}{\partial \delta \mathbf{n}_k^g} \\
&= \frac{1}{4}\frac{\partial \mathbf{R}_{b_i b_k} \exp\left([\boldsymbol{\omega}\delta t]_\times\right)\exp\left(\left[-J_r(\boldsymbol{\omega}\delta t)\frac{1}{2}\delta \mathbf{n}_k^g \delta t\right]_\times\right)\left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\delta t^2}{\partial \delta \mathbf{n}_k^g} \\
&= -\frac{1}{4}\frac{\partial \mathbf{R}_{b_i b_{k+1}}\left(\left[\left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\delta t^2\right]_\times\right)\left(-J_r(\boldsymbol{\omega}\delta t)\frac{1}{2}\delta \mathbf{n}_k^g \delta t\right)}{\partial \delta \mathbf{n}_k^g} \\
&= -\frac{1}{4}\left(\mathbf{R}_{b_i b_{k+1}}\left[\left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\right]_\times \delta t^2\right)\left(-J_r(\boldsymbol{\omega}\delta t)\frac{1}{2}\delta t\right) \\
&= -\frac{1}{4}\left(\mathbf{R}_{b_i b_{k+1}}\left[\left(\mathbf{a}^{b_{k+1}} - \mathbf{b}_k^a\right)\right]_\times \delta t^2\right)\left(-\frac{1}{2}\delta t\right)
\end{aligned}
$$

根据$\omega = \frac{1}{2}((\omega^{b_k} + n_k^g - b_k^g) + (\omega^{b_{k+1}} + n_{k+1}^g - b_k^g))$，经过$\delta n_k^g$抖动后变为$\omega + \frac{1}{2}\delta n_k^g$，与2.1中$\omega - \delta b_k^g$相比$g_{12} = \frac{\partial \delta \alpha_{k+1}}{\partial \delta n_k^g}$推导仅仅是$-\delta b_k^g$和$\frac{1}{2}\delta n_k^g$的区别，所以把$-\delta b_k^g$替换为$\frac{1}{2}\delta n_k^g$就可以得到结果：

$$
\begin{aligned}
g_{12} &= \frac{\partial \delta \alpha_{k+1}}{\partial \delta n_k^g} \\
&= \frac{-\partial \frac{1}{4}(R_{b_i b_k} exp([\omega \delta t]_\times)[a^{b_{k+1}} - b_k^a]_\times (J_r(\omega \delta t)\frac{1}{2}\delta n_k^g \delta t)\delta t^2}{\partial \delta n_k^g} \\
&= -\frac{1}{4} R_{b_i b_{k+1}} [a^{b_{k+1}} - b_k^a]_\times \delta t^2 (\frac{1}{2}\delta t)
\end{aligned}
$$

两道题完全同理

第三题

证明：

阻尼因子 μ 大小是相对于 $\mathbf{J}^\top\mathbf{J}$ 的元素而言的。半正定的信息矩阵 $\mathbf{J}^\top\mathbf{J}$ 特征值 $\{\lambda_j\}$ 和对应的特征向量为 $\{\mathbf{v}_j\}$。对 $\mathbf{J}^\top\mathbf{J}$ 做特征值分解分解后有：$\mathbf{J}^\top\mathbf{J} = \mathbf{V}\Lambda\mathbf{V}^\top$ 可得：

$$\Delta\mathbf{x}_{\mathrm{lm}} = -\sum_{j=1}^{n}\frac{\mathbf{v}_j^\top\mathbf{F}'^\top}{\lambda_j+\mu}\mathbf{v}_j \tag{9}$$

$$(V\Lambda V^T + uI)\cdot \Delta X_{lm} = -F'^T$$

$$\therefore\ (V\Lambda V^T + V\cdot uI\cdot V^T)\cdot \Delta X_{lm} = -F'^T$$

$$V(\Lambda+uI)V^T\cdot \Delta X_{lm} = -F'^T$$

$$\therefore\ \Delta X_{lm} = -\left(V(\Lambda+uI)V^T\right)^{-1}F'^T$$

$$= -V(\Lambda+uI)^{-1}\cdot V^T\cdot F'^T$$

$$= -(v_1\cdots v_n)\cdot\begin{pmatrix}\frac{1}{\lambda_1+u} & & \\ & \ddots & \\ & & \frac{1}{\lambda_n+u}\end{pmatrix}\begin{pmatrix}v_1^T F'^T \\ \vdots \\ v_n^T F'^T\end{pmatrix}$$

$$= -(v_1\cdots v_n)\begin{pmatrix}\frac{v_1^T F'^T}{\lambda_1+u} \\ \vdots \\ \frac{v_n^T F'^T}{\lambda_n+u}\end{pmatrix}$$

$\frac{v_j^T F'^T}{\lambda_j+u}$ 维数为 1×1，常数

$$= -\sum_{j=1}^{n}\frac{v_j^T F'^T}{\lambda_j+u}\cdot v_j$$

证毕！

深蓝学院
shenlanxueyuan.com

感谢各位聆听！

Thanks for Listening