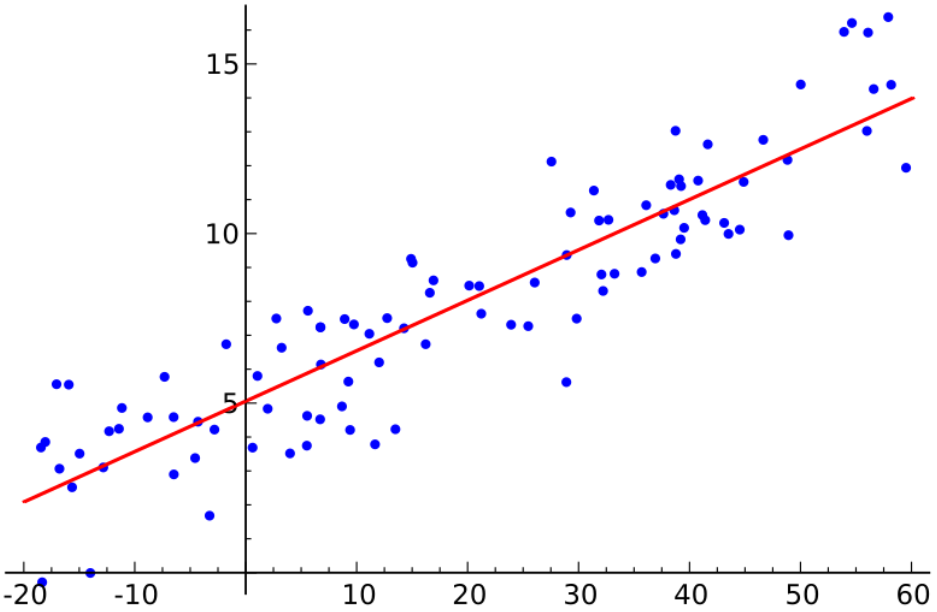


最小二乘问题的四种解法——牛顿法，梯度下降法，高斯牛顿法和列文伯格-马夸特法的区别和联系

yachen zhang
中山大学 软件工程硕士

在SLAM的过程中，我们可以构建机器人状态过程。通过对其概率的计算，最终将问题转化为了求最大似然估计的问题。

yc zhang：贝叶斯理论在SLAM状态估计中的应用
64 赞同 · 7 评论 文章


最终，可以将问题转换为求解最小二乘的问题。

那么如何求解最小二乘呢？本文将详细的介绍SLAM过程中求解最小二乘的方法。

阅读指引：

- 第1部分介绍**最小二乘**的基础流程；
- 第2部分介绍**牛顿法；**
- 第3部分介绍**梯度下降法；**
- 第4部分介绍**高斯牛顿法；**
- 第5部分介绍**列文伯格-马夸特法；**
- 第6部分介绍**几种方法的联系**。

如果觉得全文过长，可以直接定位到自己想要看的地方。书写不易，如果看完觉得对您有帮助的话，希望可以帮忙**点赞收藏**支持一下哈~

1、最小二乘

我们想要求解的问题是SLAM过程中的最大似然估计问题。由于最大似然估计问题可以转换为最小二乘问题，所以也就是最小二乘的求解问题。

我们假设目标函数为 $\boldsymbol{F(x)}$，误差函数为 $\boldsymbol{f(x)}$，则其最小二乘可以表示为[1]：

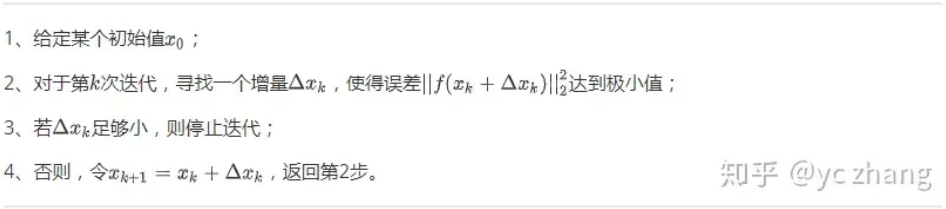
$$\min F(x) = \frac{1}{2}||f(x)||_2^2$$

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API | Printed with Pdfcrowd.com

那么如何求解这个问题呢?

常规的思路就是对目标函数 $F(x)$ 进行求导，当其导数为0的时候，求 x 的最优解，最终求得极值，这里需要注意导数为0的点，不一定就是极值点，可能会是鞍点。

然而，有些时候，若目标函数 $F(x)$ 的不是很容易进行求导，这个时候就需要使用一些迭代的方法，来使得目标函数 $F(x)$ 下降，这样就可以逐步迭代，求得最小值了。整个的迭代流程如下所示：



最小二乘问题的求解流程

因此，上诉问题，就变成了不断寻找下降增量 Δx_k 的问题。

为了方便求解，我们只需要关心误差函数 $f(x)$ 在迭代值处的局部性质，而不用考虑 $F(x)$ 在迭代值处的全局性质，这种方法在最优化，机器学习领域使用的非常广泛。

2、牛顿法

说到优化方法，那自然不能少了最基本的牛顿法。牛顿法在本科的时候大家应该都学过，这里进行一下回顾：

首先定义一个实变量 x，和其单变量函数 $f(x)$，函数的一阶导数记为 $f'(x)$，二阶导数记为 $f''(x)$ [2]。

那么首先看一个牛顿法最常用的场景------求根：

2.1、一元函数求根：

假设有根的近似解 x_0，则满足假设且更接近根的值为：

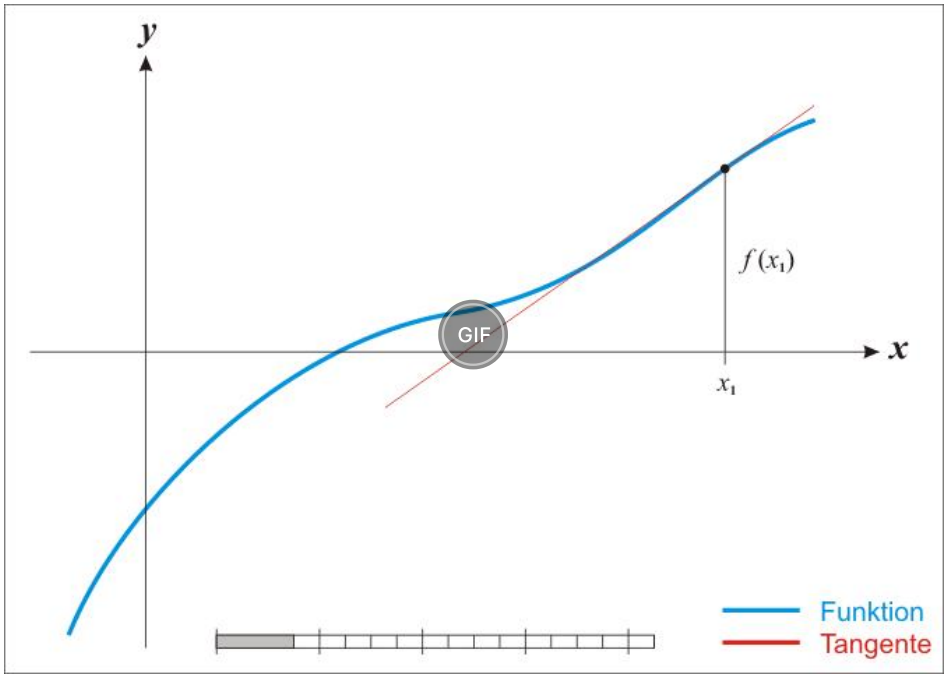
$$x_1 = x_0 - \frac{f(x_0)}{f'x_0}$$

以此类推，可以得到：

$$x_{n+1} = x_n - \frac{f(x_n)}{f'_{(x_n)}}$$

细节流程刻意参考下图：

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API | Printed with Pdfcrowd.com



牛顿法求根的逼近过程

因此，根据上面的流程，可以逐步求得 $f(x)=0$ 的根。

我们假设当前的近似解为 x_n ，那么迭代之后的下一个解，列出当前的切线方程为：

$$y=f'(x_n)(x-x_n)+f(x_n)$$

设定其在 x 轴的截距为 x_{n+1} ，将 $(x_{n+1},0)$ 代入方程，可以得到：

$$0=f'(x_n)(x_{n+1}-x_n)+f(x_n)$$

即： $x_{n+1}=x_n-\frac{f(x_n)}{f'(x_n)}$

因此，我们得到了在函数 $f(x)$ 上求解 $f(x)=0$ 的方法。

2.2、求函数的极值：

我们在前面知道了求解函数的根的方法，那么和函数的极值有什么关系呢？我们知道函数的极值的必要条件是其一阶微分等于0。那么求解微分方程为0时候的值，从而函数的极值了。

为了能够方便理解整个流程，我们还是拿一元函数 $f(x)$ 来举例。

我们首先对其在 $x=x_0$ 处进行二阶泰勒展开：

$$f(x)=f(x_0)+f'(x_0)(x-x_0)+\frac{f''(x_0)}{2}(x-x_0)^2+o(x-x_0)^2$$

其中，由于泰勒展开的特性，后面 $o(x-x_0)$ 部分不予考虑，我们只考虑前面展开部分的极值问题：

$$g(x)=f(x_0)+f'(x_0)(x-x_0)+\frac{f''(x_0)}{2}(x-x_0)^2$$

上面的式子是一个一元二次函数，那么其极值就是一阶导数为0的时候，我们可以先微分：

$$g'(x)=f'(x_0)+f''(x_0)(x-x_0)$$

令 $g'(x_1)=0$ ，则此时的 x_1 就是极值（为了方便说明，暂不考虑鞍点的情况）。

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API | Printed with Pdfcrowd.com

故 $f^{'}(x_0)+f^{''}(x_0)(x_1-x_0)=0$

$$x_1=x_0-\frac{f^{'}(x_0)}{f^{''}(x_0)}$$

以此类推，可以得到迭代公式：

$$x_{n+1}=x_n-\frac{f^{'}(x_n)}{f^{''}(x_n)} \tag{1}$$

根据这个方法就可以不断的迭代下去直到收敛，最终找到极值了。

如果是多元的情况，则一阶导数 $f^{'}(x)$ 被叫做梯度，也称之为**雅可比矩阵** J [3]（**这里不太严谨。严格来说，矩阵的梯度为一阶导的转置，函数的梯度为一阶导，这里并没有进行详细的区分**），二阶导数矩阵 $f^{''}(x)$，也被叫做**海塞矩阵** H [4]。如果是收敛的话，$\Delta x=x_{n+1}-x_n\approx 0$，则式子可以转化为：

$$\Delta x=-\frac{f^{'}(x_n)}{f^{''}(x_n)}=-\frac{J}{H}$$

也就是说：

$$H\Delta x=-J \tag{2}$$

这样，就可以求出能够取得函数 $f(x)$ 的极值点，继而算出函数 $f(x)$ 的极值。

由于牛顿法需要算二阶导数，如果高阶的话，需要算海塞矩阵，这里是有三个**缺陷**：

- 要求给定的方程需要二阶可导
- 非凸函数的海森矩阵不一定有逆
- 数据较大的时候，海塞矩阵的计算量偏大

因此，需要思考别的方法来进行最小二乘问题的优化和求解。

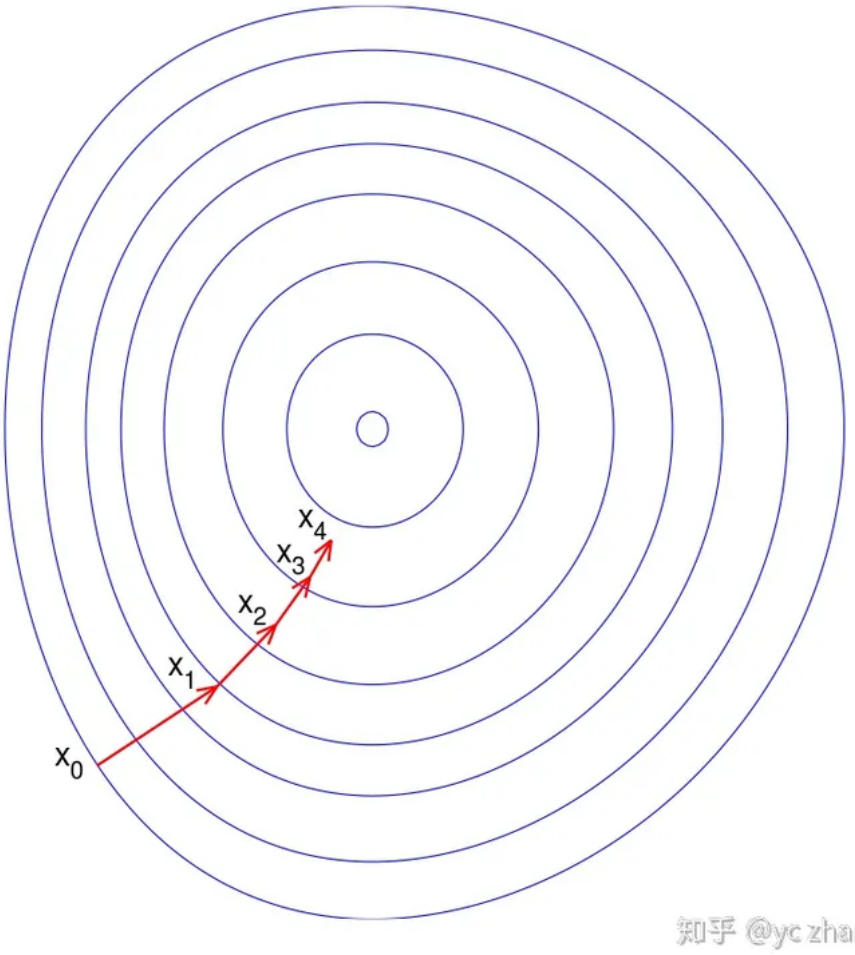
3、梯度下降法

为了能够更好的进行最值问题的优化求解，我们可以使用**高斯牛顿法（GN）**和**列文伯格-马夸特法（LM）**。

再介绍上面两个方法之前，我们首先介绍一下**梯度下降法**[5]。

梯度下降是用于找到可微函数的局部最小值的一阶迭代优化算法。为了使用梯度下降找到函数的局部最小值，我们采取与该数在当前点的梯度（或近似梯度）的负值成比例的步骤。

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API | Printed with Pdfcrowd.com



梯度下降法示意图，来自wiki

梯度下降法是基于以下的观测原理而来的：

如果实值函数 $f(x)$ 在点 $x = a$ 处可微且有定义，那么函数 $f(x)$ 在点 a 处，沿着梯度的反方向 $-\Delta f(a)$ 下降的最快。

因此，假设有个点 b，满足：

$$b = a - \gamma \Delta f(a) \quad s.t \quad \gamma > 0 \quad \& \quad \gamma \to 0$$

那么我们就可以得到：

$$f(a) \geq f(b)$$

通过这种方法，就可以找到极小值。

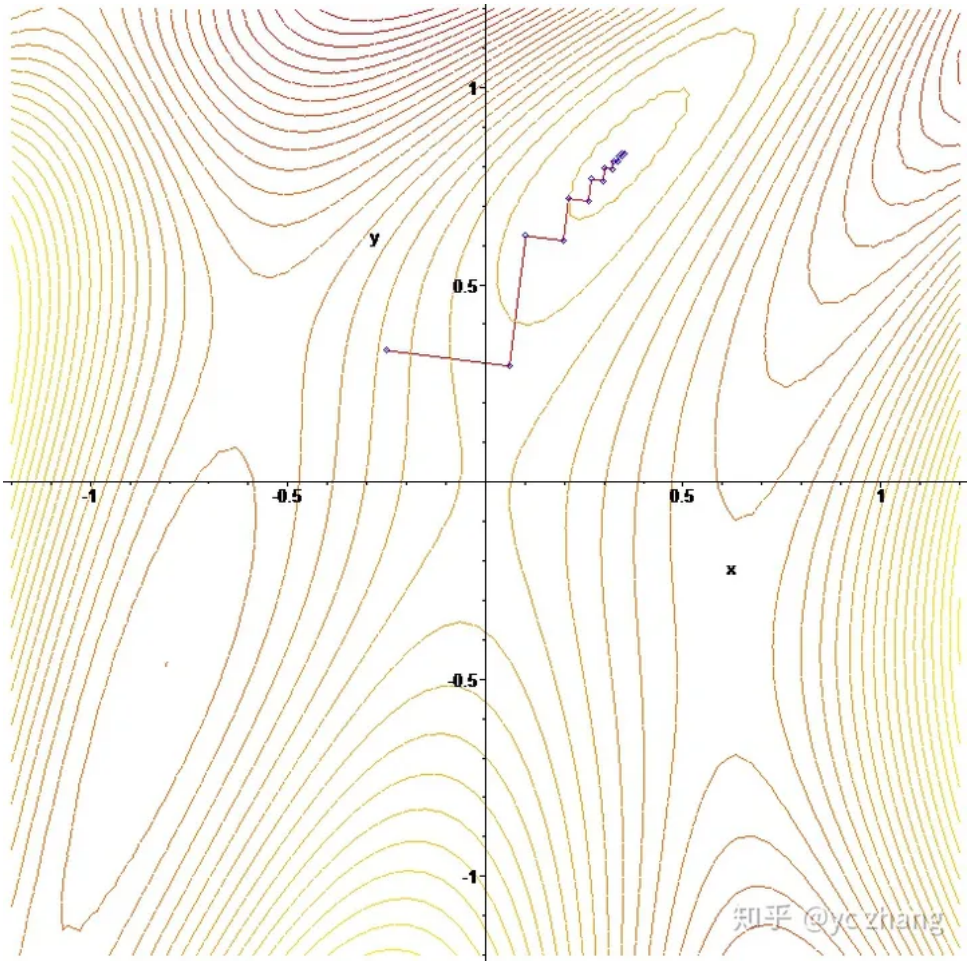
因此，得到迭代公式：

$$x_{n+1} = x_n - \gamma \Delta f(x_n) \tag{3}$$

其中，γ 是我们人为设定的参数，通过迭代，就可以得到极值。

梯度下降法每次都以梯度的反方向下降，所以，有可能会**容易走出锯齿路线，从而增加迭代次数**。

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API | Printed with Pdfcrowd.com



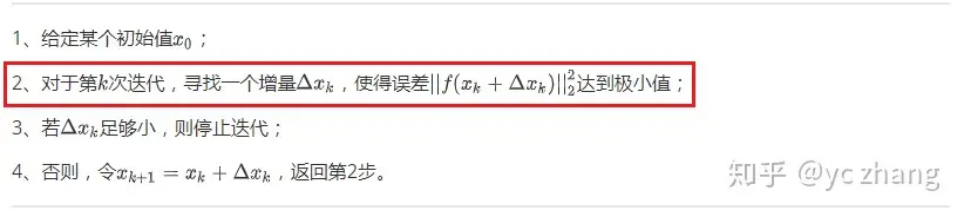
梯度下降法容易出现锯齿状，来自wiki

4、高斯牛顿法

如果代入到最小二乘问题中，牛顿法和梯度下降法都是针对目标函数 $\boldsymbol{F(x_k)}$ 来进行求解的，这样，就不可避免的需要求得海塞矩阵 $\boldsymbol{H}$ ，所以，为了避免这个问题，我们选取了误差函数 $\boldsymbol{f(x)}$ 来进行优化求解：

$$\min F(x) = \frac{1}{2}||f(x)||_2^2$$

那么，我们从上面的迭代步骤2中可以看到：



最小二乘法求解流程

那么，我们对 $\boldsymbol{f(x+\Delta x)}$ 进行一阶泰勒展开。

$$f(x+\Delta x) \approx f(x) + J(x)^T\Delta x + o(\Delta x)$$

我们需要求 $\boldsymbol{\Delta x}$ 使得上面的式子 $\boldsymbol{||f(x+\Delta x)||_2^2}$ 有最小值，所以，我们可以得到最小二乘问题为：

$$\Delta x^* = \arg\min \frac{1}{2}||f(x+\Delta x)||_2^2 \approx \arg\min \frac{1}{2}||f(x) + J(x)^T\Delta x||_2^2$$

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API | Printed with Pdfcrowd.com

为了求极值，对其求导：

$$m(x) = \frac{1}{2}||f(x) + J(x)^T\Delta x||^2 = \frac{1}{2}(f(x) + J(x)^T\Delta x)^T(f(x) + J(x)^T\Delta x)$$

$$= \frac{1}{2}(||f(x)||^2 + 2f(x)J(x)^T\Delta x + \Delta x^T J(x)J(x)^T\Delta x$$

故，对其求导可以得到：

$$m`(x) = J(x)f(x) + J(x)J(x)^T\Delta x$$

则，此时可以转化为线性求解问题：

$$m`(x) = 0 \quad \rightarrow \quad J(x)J(x)^T\Delta x = -J(x)f(x)$$

令$J(x)J(x)^T$定义为 $H(x)$，令 $-J(x)f(x)$ 定义为 $g(x)$，则此时变为了：

$$H\Delta x = g \quad s.t \quad H = JJ^T \quad \& \quad g = -Jf \tag{4}$$

这样，就可以优化求解了。上面的最小二乘的优化步骤就可以变为[6]：

1、给定某个初始值x_0；

2、对于第k次迭代，**求出当前的雅可比矩阵**$J(x_k)$**和误差**$f(x_k)$；

3、**求解增量方程：** $H\Delta x_k = g$；

4、若Δx_k足够小，则停止迭代；

5、否则，令$x_{k+1} = x_k + \Delta x_k$，返回第2步。

知乎 @yc zhang

高斯牛顿法求解最小二乘的步骤

相比较于传统的最小二乘求解方法，只更改了两个步骤。该方法的优点和缺点如下：

优点：

- 避免了求海塞矩阵，大大减少了计算量。

缺点：

- 为了求解 H^-，需要 H 矩阵可逆，但是实际上 JJ^T 只有半正定性，所以，当为奇异矩阵的时候，稳定性较差，算法不收敛。
- 如果求出来的步长 Δx_k 太大，会导致其局部近似不精确，严重的时候，可能无法保证迭代收敛。
- 容易和梯度下降法一样，陷入锯齿状，导致迭代次数较长。

不过，为了能够更好的进行最小二乘问题的求解，我们可以使用**列文伯格-马夸特法**(LM)来进行求解。

5、列文伯格-马夸特法

该方法是在高斯牛顿法的基础上进行的改进，基本的思路和原理和高斯牛顿法一样。

在高斯牛顿法的缺点中，可以看到，有一点使容易进入锯齿状，导致迭代的次数较长。所以，为了避免其步长过大导致的问题，该方法提出了**信赖区域**，设定一个区域。使得步长能够受到控制[7]。

在更新迭代的过程中，为了判定近似值的好坏，我们设定了一个评判指标：

$$\rho = \frac{f(x+\Delta x) - f(x)}{J(x)^T\Delta x}$$

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API | Printed with Pdfcrowd.com

这个指标就是我们的近似指标，可以看到其分为以下几种情况：

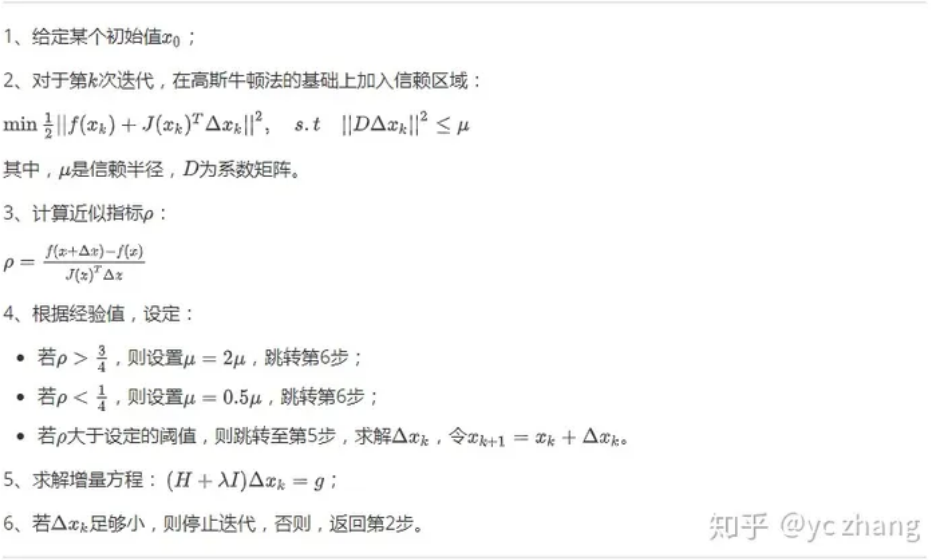
- ρ 接近1，近似是好的，不需要更改；
- ρ 太小，则实际减少的值小于近似减少的值，近似较大，需要缩小近似的范围；
- ρ 太大，则实际减少的值大于近似减少的值，近似较小，需要扩大近似的范围。

这样的话，就可以动态调整步长了。

通过近似指标，我们可以设定信赖区域的大小。当没有接近我们设定的阈值，则不断调整动态区域，直到找到好的近似结果。

当找到符合要求的近似结果后，就可以进行后续正常的迭代更新了。

因此，使用该信赖区域后，可以更新算法流程：



Levenberg–Marquardt 求解最小二乘问题的算法流程

至于增量方程的获取，可以通过拉格朗日函数来求解：

$$\min \frac{1}{2}||f(x)+J(x)^T\Delta x|| \quad s.t \quad ||D\Delta x < \mu||_2$$

构建拉格朗日函数，λ 是系数因子：

$$L(\Delta x,\lambda)=\frac{1}{2}||f(x)+J(x)^T\Delta x||^2+\frac{\lambda}{2}(||D\Delta x||^2-\mu)$$

这样的话，化简后求导就可以得到：

$$J(x)f(x)+J(x)J^T(x)\Delta x+\lambda D^TD\Delta x=0$$

我们化简后得到：

$$(JJ^T+\lambda D^TD)\Delta x=-Jf$$

在本文中，我们令 $H=JJ^T$，$g=-Jf$。在实际使用中，通常用 I 来代替 D^TD。所以，公式就变为：

$$(H+\lambda I)\Delta x_k=g \tag{5}$$

这样，就可以得到对应的增量方程了。

代入算法流程中，最终就可以优化得到最小二乘问题的极小值了。

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API | Printed with Pdfcrowd.com

6、几种方法的联系

我们已经知晓了四种优化的方法，分别是：

- 牛顿法：$H\Delta x = -J \quad s.t \quad H = f``(x_k), J = f`(x_k)$
- 梯度下降法：$\Delta x = -\gamma J \quad s.t \quad J = f`(x_k)$
- 高斯牛顿法：$H\Delta x = g \quad s.t \quad H = JJ^T$，$g = -Jf$
- 列文伯格-马夸特法：$(H + \lambda I)\Delta x_k = g \quad s.t \quad H = JJ^T$，$g = -Jf$

其实，这四种方法在最小二乘的问题求解中，也是有着联系的。

我们设定最小二乘问题为：

$$\min F(x) = \frac{1}{2}||f(x)||_2^2$$

根据针对求解的是目标函数还是误差函数，可以将问题进行分类：

6.1 针对目标函数 $F(x)$ 优化

对于目标函数$F(x)$ 进行一阶泰勒展开：

$$F(x_k + \Delta x_k) \approx F(x_k) + J(x_k)^T \Delta x_k$$

此时，此时变成求最小值的问题，则：

$$\Delta x_k^* = \arg\min(F(x_k) + J(x_k)^T \Delta x_k)$$

故，对其求最小值，可以进行求一阶导数为0：

$$\Delta x_k = -J(x_k)^T$$

可以看到，如果增加一个步长 λ， 此时的方法就是**梯度下降法**。

如果对目标函数 $F(x)$ 其进行二阶泰勒展开：

$$F(x_k + \Delta x_k) \approx F(x_k) + J(x_k)^T \Delta x_k + \frac{1}{2}\Delta x_k^T H(x_k)\Delta x_k$$

则，此时的增量方程为最小二乘问题：

$$\Delta x_k^* = \arg\min(F(x_k) + J(x_k)^T \Delta x_k + \frac{1}{2}\Delta x_k^T H(x_k)\Delta x_k)$$

则，为了求其最小值，对 Δx_k^* 进行求导：

$$J(x_k) + H(x_k)\Delta x_k = 0 \rightarrow H * \Delta x = -J$$

则此时的方法为**牛顿法**。

所以，如果从目标函数 $F(x)$ 入手的话，梯度下降法和牛顿法就是其一阶泰勒展开和二阶泰勒展开的求解方法。

6.2 针对误差函数 $f(x)$ 优化

上面正文部分的4和5，就是对误差函数的优化，可以得到两种方法：高斯牛顿法和列文伯格-马夸特法。

其中，列文伯格-马夸特法是在高斯牛顿法的基础上得来的。他们之间的联系，主要取决于参数 λ：

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API | Printed with Pdfcrowd.com

- λ较大的时候，公式变为： $\lambda I \Delta x_k = g$，此时为**梯度下降法**
- λ较小的时候，公式变为： $H \Delta x_k = g$， 此时为**高斯牛顿法**。

所以， LM 法可以在一定程度上避免线性方程组的系数矩阵的非奇异和病态的问题，从而得到更精准，更稳定的增量 Δx_k 。

通过上面内容，我们知道了牛顿法，梯度下降法，高斯牛顿法和列文伯格-马夸特法的区别和联系了。

不管是什么方法，**初始参数的选取**很重要，给一个好的参数，可以减少迭代次数，更快的收敛，从而减少很多的计算量。

在实际的使用中，可以根据具体的情况来最终决定，到底要使用那种方法来进行优化求解。

参考

1. ^ https://en.wikipedia.org/wiki/Least_squares
2. ^ https://en.wikipedia.org/wiki/Newton%27s_method
3. ^ https://en.wikipedia.org/wiki/Jacobian_matrix_and_determinant
4. ^ https://en.wikipedia.org/wiki/Hessian_matrix
5. ^ https://en.wikipedia.org/wiki/Gradient_descent
6. ^ https://github.com/gaoxiang12/slambook2
7. ^ https://en.wikipedia.org/wiki/Levenberg%E2%80%93Marquardt_algorithm

编辑于 2020-04-23 13:43

视觉SLAM十四讲（书籍） 同时定位和地图构建（SLAM） 最小二乘法

▲ 赞同 636 ▼ 43 条评论 分享 喜欢 收藏 申请转载 ···

写下你的评论...

43 条评论 默认 最新

Dreamer
前面H代表海森阵，后面H代表J乘J的转置，建议把符号改一下，不然容易把符号理解错
2020-12-06 回复 11

皮蛋皮蛋
雅各比矩阵相乘就是海森矩阵吧，一个一阶导，一个二阶导
04-11 回复 喜欢

问君能歪几层楼
一般就是这么定义的吧
03-07 回复 喜欢

Jack Sigmoid
博主写的真好，我也在看了大量的博客和书籍，发现博主有个地方其实写的不是太清楚，比如牛顿法的思路如何可以运用到求解增量？还有关于LM算法这块，历史上是有两种推导过程的。还有，如果博主能对应写出代码就最好了！在博主的基础上，2023年伊始，我带领团队攻坚公司重要项目：高精度3D骨架以及数字人，由于需要大量涉及Ceres库和Eigen库的使用，并且里面涉及到很多最优化相关的理论知识，其中最典型的是关于最小二乘问题的求解方法，以及如何将一个问题转换为最小二乘问题。我从7月份开始，写了一个非常详细的技术文档，专门写几篇博文进行总结，具体包含：优化算法类型、步长搜索约束条件、步长区间插值类型，梯度下降法、共轭梯度法、牛顿法、高斯牛顿法、L-M、Dog-Leg算法，每种算法我都自己结合理论推导，编写代码实现，有助于加深理解！同时也会结合著名的开源非线性优化算法库Ceres进行对应说明，有助于大家在以后的工作中遇到类似问题后，对于Ceres中的一些选项有更深入的理解，如有不足或错误之处，还请多多指教。
Jack Sigmoid：深度解读：最小二乘问题的一系列优化算法及代码实现（一）
Jack Sigmoid：深度解读：最小二乘问题的一系列优化算法及代码实现（二）
Jack Sigmoid：深度解读：最小二乘问题的一系列优化算法及代码实现（三）

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API Printed with Pdfcrowd.com

Jack Sigmoid：深度解读：最小二乘问题的一系列优化算法及代码实现（四）
码字不易，觉得写得好，还请点个赞+收藏+喜欢吧！
09-09 回复 2

应其是呵呵
博主您好。我想问一下LM法中，lambda和mu有什么关系，根据rho改变了mu，怎么改变lambda呀？谢谢博主🙏🙏
2021-07-15 回复 2

Jack Sigmoid
建议看看我的博客，第四讲，写得比较清楚
10-25 回复 喜欢

zechyung
rho改变mu，等同于改变补偿权值，lambda也等于权值，两者是并行关系
03-24 回复 喜欢

bumblelee
我想问一下，目标函数和误差函数有啥区别呀？还是说是同一个东西在该文章中。
2020-11-27 回复 2

刘思黎
我的理解是，误差函数是衡量两种分布之间的差异，你可以理解为一种指标；而目标函数的作用是一种工具，目的是优化这种指标，有时最终的目的并不是让误差函数为0，因为我们获得的抽样数据是有噪音的或者outlier的，所以我们需要构造一种工具既要让误差函数尽可能低，又要防止过拟合等其他问题。
2021-01-27 回复 4

errors
答主，你好，请教你一个观点，既然梯度下降法和高斯牛顿迭代法都存在相同的问题(容易陷入锯齿状，导致迭代次数增加)，
那么为啥在LM中还要将这两者结合呢，是因为LM可以通过信赖区域来调整步长来避免这种“陷入锯齿状”的现象吗？
我看网上有人说“一开始采用梯度下降法来迭代得到一个好的结果，作为接下来高斯牛顿迭代的初值，这样迭代的速度最快”
我的理解：首先是速度快慢，肯定是GN快，他是二阶收敛，只不过可能会存在“步长太大时，函数值不一定下降的情况”；其次对于梯度下降法和高斯牛顿法的谁先谁后，我觉得时没有规定的，但是对于拉格朗日乘子lamda的计算我是太理解的，想听听答主的解答
以上是我个人的理解，若有问题，麻烦答主不吝赐教🙏🙏
2021-11-29 回复 1

信息门下舔狗
最小二乘里面f(x)是一个标量函数为什么要用表示向量或者矩阵的范数的一对双竖线抱起来？
2021-09-26 回复 1

leon说要做大事
我说个问题 高斯牛顿的一阶泰勒展开，那个雅克比矩阵怎么变成是转置的？
2022-08-20 回复 1

超级无敌么么哒
梯度跟雅各比矩阵的区分很明显，梯度是由标量对向量求导得来，而雅各比矩阵是向量对向量求导。
01-25 回复 喜欢

Keith
感谢🙏
2020-06-30 回复 喜欢

精神病院荣誉院士
感谢感谢，另外想问一下，那个等高线图是怎么画的
2020-05-27 回复 喜欢

yachen zhang 作者
这个是wiki的， 有标注出处，不是我画的🤣
2020-05-27 回复 喜欢

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API | Printed with Pdfcrowd.com

点击查看全部评论 >

文章被以下专栏收录

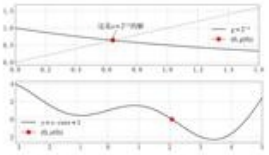
激光雷达SLAM算法大全
从原理深入解析激光SLAM算法

推荐阅读

【数学分析新讲笔记】9.3变上限积分和牛顿-莱布尼兹公式…

9.3.0前言上一节: 9.2可积函数类下一节: 9.4积分不等式数学分析新讲笔记整理在： 数学分析新讲笔记目录本节将阐述变上限积分，并对牛顿-莱布尼兹公式再讨论。 最后会谈到曲率和曲率半径的概…

LordB... 发表于分析学



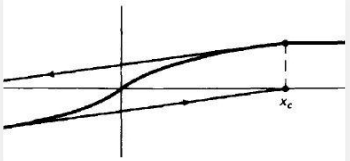
不动点法和牛顿法求解非线性方程的数值解（Python）

Bin H... 发表于科学计算学...

$$\rho_k = x_1^k + \cdots + x_n^k, k \in \{1, 2, 3 \ldots\}$$

【国际数学竞赛】牛顿恒等式的应用

双木止月T... 发表于国际理科



比较牛顿法、有限差分法、割线法的计算效率和稳定性

tomm 发表于有限元模拟

Convert web pages and HTML files to PDF in your applications with the Pdfcrowd HTML to PDF API
Printed with Pdfcrowd.com