这是标题

陈烁龙

2022 年 8 月 10 日

目录

插图

表格

1 RANSCA

RANSAC 算法假设数据中包含正确数据和异常数据 (或称为噪声)。该算法核心思想就是随机性和假设性，随机性是根据正确数据出现概率去随机选取抽样数据，根据大数定律，随机性模拟可以近似得到正确结果。假设性是假设选取出的抽样数据都是正确数据，然后用这些正确数据通过问题满足的模型，去计算其他点，然后对这次结果进行一个评分。该算法的步骤表述为：

1. 给出迭代次数和容许的误差阈值。容许的误差阈值即我们可以接受的，点到拟合的曲线的容许偏差。该值需要根据具体问题给出；
2. 根据要拟合的模型，随机选择最少的拟合数据集，拟合出一个初始的模型；
3. 根据当前拟合得到的模型和容许误差，选择出符合点 (inliers)。如果符合点比较少，那么抛弃掉当前的模型；
4. 基于满足容许误差的数据点，再次拟合模型；
5. 基于新的模型，计算每个符合点的残差。而后计算平均残差和作为该模型的衡量数值 (越小越好)；
6. 重复迭代。迭代完成后，返回平均残差和最小的模型。

当然，为了结果更优，可以加入均值漂移算法。具体来说：

1. 基于 RANSAC 算法拟合的模型，选择出符合点 (inliers)；
2. 基于满足容许误差的数据点，再次拟合模型；
3. 基于新的模型，计算每个符合点的残差；
4. 当相邻迭代之间的模型残差不再变化时，结束迭代；

2 代码框架

本次基于 RANSAC 算法，给出了一般问题的代码框架。用户只需要继承虚拟类并且重载一些纯虚函数即可。具体的代码如下所示：

Listing 1: RANSAC 虚拟类

```cpp
/**
* @brief virtual class to solve ransac problem
*
* @tparam ElemType the element type
* @tparam EigenParamVecSize the param size [for 'eigen' vector ]
* @tparam SubsetSize the minimum data set to fit the model
*/
template <typename ElemType, int EigenParamVecSize, int SubsetSize>
class Ransac {
public:
// the default constructor
Ransac() = default;

public:
/**
* @brief solve a ransac problem with mean shift
*
* @param data the dataset
* @param modelParams the params for the final model ['eigen' vector type]
* @param modelAvgResiual the average resiual for the final model
* @param inlierResidualThd to decide an element is an outlier or an inlier
* @param inliersRate the rate for inliners in the total dataset
* @param iterCount the loop count for ransac
* @return true if the result is good
* @return false if some error happened when solving the ransac problem
*/
bool solveWithMeanShift(const std::vector<ElemType> &data,
Eigen::Vector<double, EigenParamVecSize> &modelParams,
double &modelAvgResiual,
const double inlierResidualThd,
const double inliersRate = 0.3,
const std::size_t ransacIterCount = 20);

/**
* @brief solve a ransac problem
*
* @param data the dataset
* @param modelParams the params for the final model ['eigen' vector type]
* @param modelAvgResiual the average resiual for the final model
```

```
* @param inlierResidualThd to decide an element is an outlier
    or an inlier
* @param inliersRate the rate for inliners in the total
    dataset
* @param iterCount the loop count for ransac
* @return true if the result is good
* @return false if some error happened when solving the ransac
     problem
*/
bool solve(const std::vector<ElemType> &data,
Eigen::Vector<double, EigenParamVecSize> &modelParams,
double &modelAvgResiual,
const double inlierResidualThd,
const double inliersRate = 0.3,
const std::size_t iterCount = 20);

protected:
virtual bool fit(const std::vector<ElemType> &subset,
Eigen::Vector<double, EigenParamVecSize> &params) const = 0;

virtual bool residual(const Eigen::Vector<double,
    EigenParamVecSize> &params,
const ElemType &inlier,
double &resiual) const = 0;

private:
// ...
};
```

注意：纯虚函数一共有两个。一个用于计算残差，一个用来拟合模型。该两个纯虚函数会在解算函数 solve(...) 和 solveWithMeanShift(...) 中使用。在对于具体问题时，用户只要继承该基类，并且实现该两个纯虚函数即可。

3 案例问题

现在，使用 GPS 进行单点定位，基于具体的定位算法，在同一个点位多次进行平面位置的确定。由于误差的存在，当不存在粗差时，会呈现正态分布。但是，如果在定位的时候，有多路径效应，导致了巨大的系统误差，如下图所示。其中，绿色的是没有粗差时的测量点位，红色的点是存在粗差时的测量点位。现根据当前数据，给出所测点位的平面坐标平差结果。

为使用我们之前的 RANSAC 类，我们需要继承它，然后实现相应的算法。

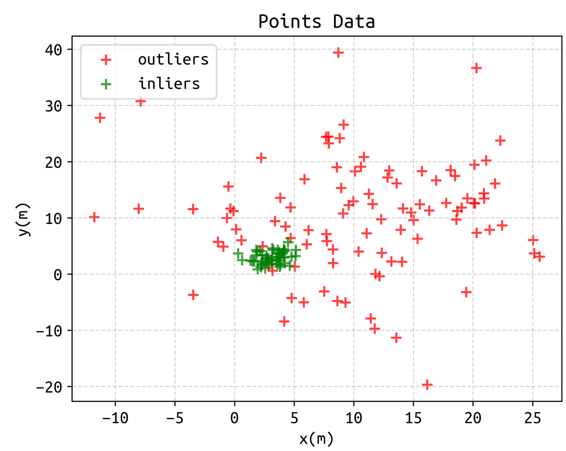

图 1: 原始数据

首先是拟合算法。我们使用最小二乘法。假设我们用 $p^i(x^i, y^i)$ 表示点集中的第 i 个测得的点，用 $p(x, y)$ 表示点位真值。那么，误差函数为：

$$\begin{cases} e_x^i = x - x^i \\ e_y^i = y - y^i \end{cases}$$

即：

$$\begin{pmatrix} e_x^i \\ e_y^i \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} x^i \\ y^i \end{pmatrix}$$

$$\rightarrow V^i = B^i x - l^i$$

我们基于 $min(V^T V)$ 进行最小二乘法，那么有：

$$\begin{aligned} \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} =& (B^T B)^{-1} B^T l \\ =& \frac{1}{n} B^T l \\ =& \frac{1}{n} \sum_{i=1}^{n} \begin{pmatrix} x^i \\ y^i \end{pmatrix} \end{aligned}$$

也就是说，所有点的重点即为最小二乘的平差结果。

其次是每一个点的残差，我们直接用拟合平差结果和实际数据点的距离作为残差 (模型不符值)。即：

$$e^i = \sqrt{(x^i - \hat{x})^2 + (y^i - \hat{y})^2}$$

4 测试

我们使用了三种方法进行测试，即：直接最小二乘法、最小二乘法 +RANSAC、最小二乘法 +RANSAC+MeanShift。结果如下图所示。

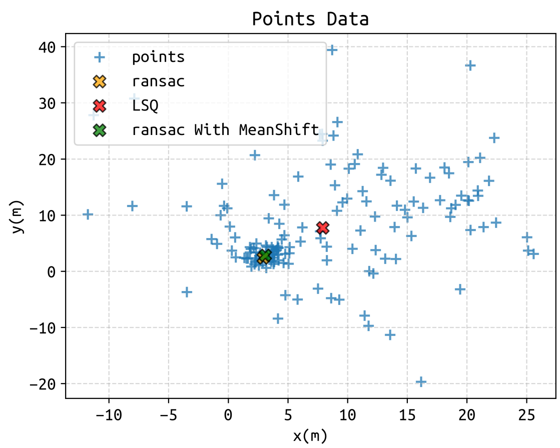

图 2: 解算结果

可见，当数据中存在粗差时，直接最小二乘法的结果是错误的。而最小二乘法 +RANSAC 很大程度上改善了结果。而最小二乘法 +RANSAC+MeanShift 结果是最优的。