Minuit2 の使い方

2021/05/16

Yutaka Tsuzuki

yutaka.tsuzuki@ipmu.jp

概要

この文書は, ROOT (https://root.cern.ch) に含まれるパッケージである Minuit2 (https://root.cern.ch/root/htmldoc/guides/minuit2/Minuit2.html) の使い方をまとめたものである.

はじめに

Minuit2 とは何か

Minuit2 は, ROOT のパッケージの一つであり, 数値最適化を実行するコードです. ROOT には古くから Minuit というパッケージが含まれていますが, Minuit はあまり複雑な最適化ができない仕様になっているため, 昨今の複雑な解析をするためには使い勝手がよくありません. 一方, Minuit2 は対象となるオブジェクトを自分で作ることができ, さまざまな最適化問題に応用することができます.

お前は誰だ

Yutaka Tsuzuki (都築 豊) は, 2021年現在, 東京大学大学院理学系研究科物理学専攻博士課程の学生 (D2) です. 原子を対象にした硬X線の偏光観測や理論計算に取り組んでいます.

準備

Minuit2 を使うためには, ROOT をビルドする時にオプションをつける必要があります. 具体的には, `cmake` を実行する際に `-DMINUIT2` オプションをつけます. 詳しくは, https://root.cern/install/build_from_source/ を参照してください.

使い方

Minuit2 の使用者が書く部分は, 大きく分けて2つあります.

1. 最適化の対象となる誤差 (のようなもの) を記述するクラス
2. 最適化を実行するスクリプト

これらの他にも, 最適化の対象となる関数を記述するオブジェクトを書く場合もあります (Minuit2 の公式チュートリアルなど) が, これは1. に含めることができるので本文書では~~手抜き~~割愛します.

実例

次のようなデータ点の列 (x, y) を多項式でフィットする問題を考えましょう.

```
0.0,0.0
71.8990,59.5412
97.9973,80.9971
145.0261,122.0614
161.9725,136.4743
316.9642,276.3980
344.0851,302.8530
398.9420,356.0170
426.0541,383.8510
528.9643,511.0000
```

以下では, フィッティング関数は3次多項式を仮定します.

誤差クラス

まず, 誤差クラスを定義します.

```
#include "Minuit2/FCNBase.h"
#include <vector>
#include <cmath>

class Fcn : public ROOT::Minuit2::FCNBase {
private:
  // array of data points
  std::vector<double> a_x;
  std::vector<double> a_y;
  // error
  double error_def;
public:
  // inherited methods
  void SetErrorDef(double edef) { error_def = edef; }
  virtual double Up() const { return error_def; }
  // const, dest
  Fcn() {}
  virtual ~Fcn() {}
  // add data points
  void add_point(double, double);
  // call
  virtual double operator()(const std::vector<double>&) const;
};

void Fcn::add_point(double x, double y) {
  a_x.push_back(x);
  a_y.push_back(y);
}

double Fcn::operator()(const std::vector<double>& a_par) const {
  double err = 0.0;
```

```
  int n_point = a_x.size();
  for(int i_point = 0; i_point < n_point; ++i_point) {
    double val_fcn = 0.0;
    int n_par = a_par.size();
    for(int i_par = 0; i_par < n_par; ++i_par) {
      val_fcn += a_par[i_par] * std::pow(a_x[i_point], i_par);
    }
    err += (val_fcn - a_y[i_point]) * (val_fcn - a_y[i_point]);
  }
  return err;
}
```

`ROOT::Minuit2::FCNBase` クラスを継承する必要があります. 必要となるメソッドは主に3つあり, `SetErrorDef`, `Up`, `operator()` です. 前者2つに関しては, 上に記したような簡単な実装で十分です. 重要なのは `operator()` メソッドで, これは呼び出された時に誤差 (のようなもの) を返す関数だと思ってください. 呼び出し時に, 引数としてパラメータの列が Minuit2 から渡されます. ここでは, 多項式の係数をパラメータとして, 二乗誤差を返しています.

最適化を実行するやつ

以上のクラスを読み込んで, 最適化を実行してみましょう. コードは以下のようになります.

```
#include <iostream>
#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include "Fcn.h"
#include "Minuit2/FunctionMinimum.h"
#include "Minuit2/MnUserParameterState.h"
#include "Minuit2/MnMigrad.h"


void read_csv(std::string fname, std::vector<double> &a_x, std::vector<double> &a_y) {
  // file stream
  std::ifstream ifs(fname);
  // data buffer
  std::string buf_str, bufbuf_str;
  // read each line
  while(std::getline(ifs, buf_str)) {
    // something ridiculous
    std::istringstream strstr(buf_str);
    std::vector<double> a_var;
    while(std::getline(strstr, bufbuf_str, ',')) {
      a_var.push_back(std::stod(bufbuf_str));
    }
    a_x.push_back(a_var[0]);
```

```
    a_y.push_back(a_var[1]);
  }
}


int fit_data(std::string fname) {
  // read csv file
  std::vector<double> a_x, a_y;
  read_csv(fname, a_x, a_y);

  // function object
  Fcn fcn;

  // add data points
  int n_point = a_x.size();
  for(int i_point = 0; i_point < n_point; ++i_point) {
    fcn.add_point(a_x[i_point], a_y[i_point]);
  }

  // fit
  ROOT::Minuit2::MnUserParameters par;
  par.Add("p0", 0.0, 0.01);
  par.Add("p1", 1.0, 0.01);
  par.Add("p2", 0.0, 0.01);
  par.Add("p3", 0.0, 0.01);
  ROOT::Minuit2::MnMigrad migrad(fcn, par);
  ROOT::Minuit2::FunctionMinimum min = migrad();

  // print parameters
  std::cout << min << std::endl;

  return 0;
}
```

`fit_data` 関数がメインの関数です. まず, データ点を誤差クラスに読み込みます. 次に, `ROOT::Minuit2::MnUserParameters` でパラメータを定義します. さらに, 最適化を実行してくれる人 (`ROOT::Minuit2::MnMigrad` 君) に誤差クラスとパラメータを渡して最適化を実行します. 結果は `ROOT::Minuit2::FunctionMinimum` オブジェクトとして返され, 最後にそれを出力しています. 関数の詳しい仕様については, 下記参考文献を参照してください.

実行結果

ROOT のインタプリタから動かせます. コンパイルは筆者の環境ではなぜかエラーが出て通りませんでした.

```
 ------------------------------------------------------------------
| Welcome to ROOT 6.25/01                        https://root.cern |
| (c) 1995-2021, The ROOT Team; conception: R. Brun, F. Rademakers |
| Built for macosxarm64 on Apr 27 2021, 15:09:33                   |
| From heads/master@v6-25-01-768-g2a10b6173b                       |
```

```
   | With Apple clang version 12.0.5 (clang-1205.0.22.9)              |
   | Try '.help', '.demo', '.license', '.credits', '.quit'/'.q'       |
    ------------------------------------------------------------------

root [0] .x fit_data.cpp("../data/adc_epi.dat");

  Valid         : yes
  Function calls: 116
  Minimum value : 19.80857172
  Edm           : 2.973300443e-17
  Internal parameters:
    -0.9302816606
     0.8666914265
 -0.0002593869795
  8.447647724e-07
  Internal covariance matrix:
     0.88473142   -0.011790987  4.0957838e-05 -4.1574599e-08
   -0.011790987  0.00023665432 -9.6366121e-07  1.0723834e-09
  4.0957838e-05 -9.6366121e-07  4.3016205e-09  -5.095849e-12
 -4.1574599e-08  1.0723834e-09  -5.095849e-12  6.3150547e-15
  External parameters:
  Pos |     Name     |  type    |      Value        |    Error +/-
    0 |          p0  |  free    |    -0.9302816606 | 9.164669104e-162
    1 |          p1  |  free    |     0.8666914265 |
    2 |          p2  |  free    | -0.0002593869795 |
    3 |          p3  |  free    |  8.447647724e-07 |
(int) 0
```

参考文献

「Minuit2を使う」 https://www.cns.s.u-tokyo.ac.jp/~masuoka/post/tminuit2/

- この文書の100倍くらい詳しい Minuit2 の解説です.

"Minuit2" https://root.cern.ch/root/htmldoc/guides/minuit2/Minuit2.html

- 一応, 公式説明書だと思われます.

「C++でCSVファイルを読み書きする方法」 https://qiita.com/shirosuke_93/items/d5d068bb15c8e8817c34

- データの読み込みにあたり参考にしました.