日本オペレーションズ・リサーチ学会
2022年 春季研究発表会

非線形最適化用ライブラリの整備

01308414　関西大学　　*檀 寛成　DAN Hiroshige
01111894　大阪府立大学　楠木 祥文　KUSUNOKI Yoshifumi

1. はじめに

非線形最適化問題 (NLP) の求解アルゴリズムを実装しようとすると，例えば次のような問題に直面するであろう．

- 問題をどう入力するか？
  - 線形計画問題のように係数を与えるだけでは NLP は定義できない…
- 関数の偏導関数値をどうするか？
  - ベンチマーク問題だったら偏導関数値を返す関数も提供されているけど…
- コアになるアルゴリズム以外の実装は？
  - （例えば）内点法実装のためには線形方程式が解けないと…
  - KKT 条件のチェックはどうしよう…

このように，NLP や NLP をサブルーチンとして利用する求解アルゴリズムを実装するためには，多くの副次的な実装が必要になるため，「参入障壁」が高い．この状況は，NLP 関連の研究や実応用にとって望ましくない．

このような問題を（部分的にでも）解決するため，我々のグループでは，NLP を解くためのライブラリの整備を経年的に行っている（例えば [1]）．本発表では，我々が整備した（あるいは整備中の）ライブラリについて紹介し，NLP 関連の新しい研究や応用を誘発することを狙いたい．

2. ライブラリの概要

2.1. 開発コンセプト

本プロジェクトで開発しているライブラリには “NOA” という冠名を付している[1]．NOA ライブラリの整備に際しては，（問題を構成する関数が連続的微分可能であるような）NLP 関連のアルゴリズムを実装するにあたり必要な機能が一通り提供される，“all-in-one” な状態を目指している．また，最適化ソルバそのものも実装・公開することで，NOA ライブラリの使い方を示すとともに，（必ずしも NOA ライブラリを使わない方にも）NLP 関連アルゴリズムの実装例を提供したいと考えている．

2.2. 現在のライブラリ構成・動作環境

現在までに我々が実装した（あるいは実装中の）NOA ライブラリは以下の通りである[2]．

- NOA_base，NOA_nlpbase: NLP 用基本機能ライブラリ
  - 変数値/双対変数値の管理・更新機能
  - 典型的な部分問題の作成・求解機能
  - KKT 条件判定機能
  - ステップサイズルール
- NOA_ml: モデリング言語ライブラリ
  - モデリング言語 AMPL のサブセットを処理可能
- NOA_ad: 自動微分ライブラリ
  - 2 階微分まで計算可能
- NOA_ip，(#) NOA_ip_sparse: 内点法を用いた NLP ソルバ
  - NOA_ip_sparse は問題の疎性を利用
- (*) NOA_qp: 有効制約法に基づく凸二次計画問題ソルバ
- (*) NOA_sqp: 逐次二次計画法を用いた NLP ソルバ

上記ライブラリのうち，NOA_ad は，NOA_ml を通じて与えられた関数の偏導関数値を計算可能であるが，これは必ずしも最適化問題の形をしていなくてもよい．すなわち，NLP とは関係のない文脈でも自動微分を利用することが可能である．

本ライブラリは C/C++ で実装されており，Windows, Mac, Linux での動作実績がある．コンパイラは gcc/g++, icc/icpc が利用可能である[3]．

[1] 元来，任意精度計算可能な非線形最適化ソルバを開発していたため “Nonlinear Optimizer with Arbitrary precision arithmetic” の略で NOA としていたが，現在はライブラリの冠名にも利用している．現在も，一部のライブラリ・ソルバについては任意精度計算が可能である（任意精度計算用ライブラリ GMP を使用）．

[2] (#): 現在実装中．(*): 本プロジェクト以前に実装したが，本プロジェクト用に移植できていないもの．

[3] 一部のライブラリでは Intel MKL (Math Kernel Li-

3. ライブラリを用いた実装例

ここでは，自動微分ライブラリ NOA_ad を利用するサンプルと，内点法ソルバ NOA_ip の構成の一部を紹介することで，ライブラリの使い方や，ライブラリを用いたソルバ実装のイメージを示す[4].

3.1. 自動微分ライブラリの利用イメージ

自動微分ライブラリ NOA_ad は，以下のようなコードで利用できる.

```
int main(int argc, char** argv)
{
  // 準備
  noa_ad* na = new noa_ad;
  na->inputFile(2, "model.txt", "data.txt");
  na->prepFuncAndJac(); // 関数値/1 階微分計算の準備
  na->prepHess();       // 2 階微分計算の準備
  na->setValData();     // 変数値の設定

  // 必要領域の計算・確保
  int  Fsize = na->getFuncSize();
  double* Fval = new double[Fsize];
    // ↑関数値取得用のメモリ領域確保
    // Jval, sJval, sHval も同様に領域確保
  int* sJ_rind = new int[sJsize];
  int* sJ_cind = new int[sJsize];
    // ↑ 1 階微分値が非零な箇所の添字取得用
    // sH_rind, sH_cind1, sH_cind2 も同様

  // 関数値/1 階微分/2 階微分の計算
  // 疎性利用なし
  na->calcJac(Fval, Jval);
  // 疎性利用あり
  na->setSparseJacInd(sJ_rind, sJ_cind);
  na->calcSparseJac(Fval, sJval);
  na->setSparseHessInd(sH_rind, sH_cind1, sH_cind2);
  na->calcSparseHess(Fval, sJval, sHval);
  return 0;
}
```

上記コードのうち，“inputFile” の行では，モデルとデータがモデリング言語ライブラリ NOA_ml にて処理され，NOA_ad に渡されている.

3.2. 内点法の実装イメージ

本プロジェクトで実装している内点法ソルバ NOA_ip の実装イメージ（クラス宣言の一部）は以下の通りである.

```
class NLPIP
{
  // 内部変数の宣言（一部）
  int    maxitr;   // 反復回数上限
  double time_lim; // 計算時間上限
  double eps;      // 許容残差
  double mu;       // バリヤパラメータ

  // NOA ライブラリの利用
  Armijo  *ar; // Armijo のルール (NOA_nlpbase)
  BFGS    *bf; // BFGS 法         (NOA_nlpbase)

  // 内点法用メソッド（一部）
  void setInternalParam(); // 内部パラメータ設定
  void solveSubProb();     // 部分問題の求解
  void makeCoefMat();      // 部分問題生成
  void makeRightSideVec(); // 部分問題生成
  void update_mu();        // バリヤパラメータ更新

public:
  // クラスコンストラクタ・デストラクタ
  NLPIP(int filenum, char** filepath);
  ~NLPIP();
  // 外部への提供関数
  NLPbase *nb;        // NOA_nlpbase の利用
  int  findOptSol();  // 求解
  void makeSolFile(); // 解ファイル生成
};
```

上記コードでは，NLP 用の基本ライブラリ NOA_nlpbase 内で定義されているメソッドを利用している．さらにそれらのメソッドでは，自動微分ライブラリ NOA_ad や，Intel MKL などの外部ライブラリ（特に線形代数演算）を呼び出して利用している．この部分は，ライブラリ作成時にカプセル化されているため，その実装の詳細を陽に理解しなくとも，ライブラリの利用が可能である.

4. おわりに

本発表では，我々が開発・整備している非線形最適化用ライブラリ NOA を紹介した．現在は「β 版」と呼ぶべき状態だが，利用希望の方，また開発に興味のある方は是非連絡を下さい[5].

今後は，現在のライブラリの改良とともに，

- 様々な一次法
- 連続微分不可能な項を含む関数の最適化手法
- Python など他の言語から NOA ライブラリを呼び出す API

などを実装・整備したいと考えている.

なお，本研究の一部は JSPS 科研費 18K11185 の助成を受けたものです.

参考文献

[1] 檀 寛成，野口 将嗣，OpenMP による自動微分ソフトウェアの高速化，日本オペレーションズ・リサーチ学会 2021 年春季研究発表会アブストラクト集 2-A-3.

brary) の利用を前提としている．その場合は icc/icpc（インテル C/C++ コンパイラ，無償利用可）が必須である.

[4]紙幅の都合上，関数名等を略記したり，コードを大幅に省略している．発表にてコードの詳細を示す予定である.

[5]E-mail: dan@kansai-u.ac.jp