# 4．データを解析しよう

大舘　暁，後藤基志
（核融合科学研究所）

Let's Analyze the Data !

OHDACHI Satoshi and GOTO Motoshi
*National Institute for Fusion Science, 322-6, Oroshi-cho, Toki 509-5292, Japan*

(Received 19 January 2005)

Visualization in data acquisition process is discussed. In order to check the quality of the acquired data, quick plotting of signals is required. Softwares for the data analysis, e.g., plotting applications with GUI and interpreter languages, are compared. Selection of the numerical libraries might be important in determining the optimum language for the analysis.

**Keywords:**
data acquisition, visualization, computer language

## 4.1　データ解析の流れ

前章までで，実験装置からのデータがコンピュータで読める形で格納されました．本章では格納されたデータをどのように解析するかについて考えていきます．データを解析していく過程というのは典型的には Fig. 1 のようになるでしょう．

それぞれの過程で何らかのコンピュータプログラムを使ってデジタルデータを取り扱うことになります．主に PC での処理を想定して解析用のプログラムにはどんな選択肢があるかを考えていきます．

## 4.2　データを表示する

一般に実験というものは一度やっておしまいということはあり得ず，データを取得したら，得られたデータについて検討を行い，問題点があればそれを修正し，またもう一度実験してみる，という手続きを何度も繰り返すのがふつうです．その繰り返しのなかで，データに含まれるさまざまな誤差が取り除かれ，また，データの精度が高められていくからです．ですから，効率のよい実験を行うためには，データを簡単にそしてすばやく確認できるようにしておくことは非常に重要です．データを取るだけ取ってあとからまとめて解析しようとしたら，機器の設定値がまちがっていたり計測値が飽和していて，時間と労力とお金を無駄にしてしまった経験は，多かれ少なかれみなさんお持ちでしょう．

多くのプラズマ実験では放電周期が設定されており，例えば LHD では 3 分ごとにプラズマが生成されます．そのような場合には，この決められた時間内に，取得データの検討および次の放電へ向けた放電条件あるいは計測器設定値の変更等をおこなう必要があります．まさに簡単ですば

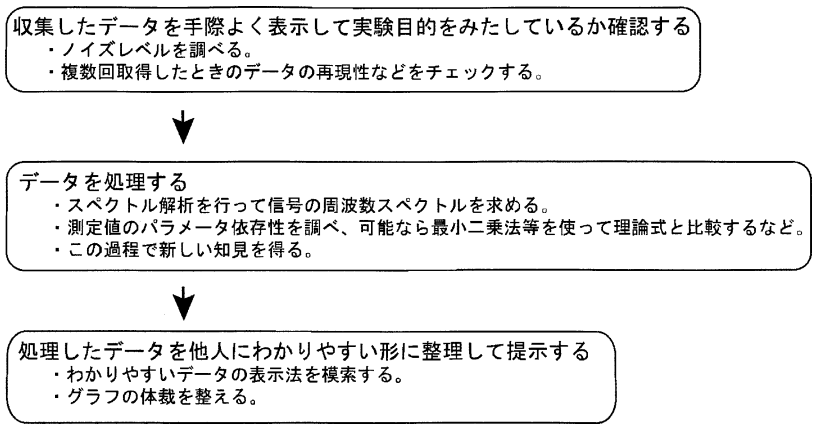


Fig. 1　データ解析の流れ

*author's e-mail: ohdachi@nifs.ac.jp, goto@nifs.ac.jp*



J. Plasma Fusion Res. Vol.81, No.3 (2005) 197-203

やいデータ表示が不可欠です．

もっとも簡単な例として，計測器に専用アプリケーションが付属していて，データ取得と同時に可視化も行ってくれるような場合もあるでしょう．また，オシロスコープを用いた検出器出力のリアルタイムモニターも伝統的な手法です．最近のデジタルオシロとPCの組み合わせなら，計測とほぼ同時に物理量へ変換するため何らかの演算を施してからグラフ表示させることもおそらく可能でしょう．確認すべきデータ数が少なければ，このような手法でも十分でしょう．しかし，規模が大きくなってきて実験に携わる人の数が増えてくると，同じデータを異なった場所で参照する必要があったり，異なる場所で取られたデータを一か所で同時に参照したりする必要が出てきます．その結果，すべてのデータを統一した規格のもとで一元的に管理するほうが都合が良いことに気づきます．これまでの章で，その具体的な方法について詳細に説明を行ってきました．この章では，これら取得されたデータを可視化する方法について考えます．

計測器付属のアプリケーションソフトを使ってデータを取得し保存している場合，基本的にはデータを表示すること自体に何ら問題はないでしょう．しかし，しばしばあることですが，例えば多くのデータをすばやく表示するのにマウスを使った操作が煩わしいなど，その操作性が気に入らなくなり，自分の使い慣れたグラフ描画ソフトを使いたくなったりします．デジタルオシロを使っている場合も同じことです．独自開発のデータ収集システムを採用している場合には，もちろん最初からデータ表示のことも考えておかなくてはなりません．ここで腕に覚えのあるプログラマなら，グラフ描画ソフト自体をゼロから作ってしまうかもしれませんが，世の中には多くの便利なソフトが存在するので，表示自体はそれらに任せるのもひとつの手です．

以下では，一般の計測機器によって取得されたデータを，汎用のグラフ描画ソフトで表示するための手続きについて具体的に考えていきます．まずは保存されたデータへのアクセスですが，いまやネットワークを介して届く範囲にデータファイルがありさえすれば，どこにいてもデータの参照は可能です．ここではネットワーク上のどこかにあるファイルを手元のPCの画面に表示させることを想定します．

データは，それを保存するハードディスクなどの記憶容量を節約するため，バイナリ形式で記録されるのがふつうです．「バイナリ形式で」とは，コンピュータのメモリ上のイメージそのものの形でという意味です．例えば16ビットの分解能を持つ整数値はバイナリ形式ならまさに1データあたり2バイトですが，アスキー形式，つまり目で見てわかる形で，縦1列に数値をファイルに書き込む場合，有効数字の桁数や書き方にもよりますが，少なくとも3，4倍程度の記憶領域を必要とします．

しかし，汎用のグラフ描画ソフトでは，バイナリ形式のデータそのままでは扱えない場合が多いため，一般に読み込む前にアスキー形式に変換してやる必要があります．計測器付属のアプリケーションを使っている場合は初めからそういった機能を持っていることもあり，ひとつやふたつのデータを変換するのには重宝しますが，例えば一度に100個のデータを変換したいときには不便です．そのような場合には簡単なフィルタプログラム，つまり，バイナリ形式のデータを読み込んでアスキー形式に変換してファイルに書き出すプログラムを書いて一括変換することになります．

フィルタプログラムにはどんな言語を使っても良いのですが，Perl*1やRuby*2などのインタプリタ系言語が便利です．描画ソフトによってはアスキー形式のファイルを読み込む際に，固有の厳密なフォーマットを要求する場合もありますが，その要求に従ってデータファイルを作成するのも，これらの言語を使えば朝飯前です．

一口にバイナリ形式といっても数値の型によって表現方法が違ったり，ひとつの数値あたり必要とするビット数もさまざまです．目的がバイナリ形式のデータをアスキー形式に変換するためだけであれば，必ずしもこれらの表現方法自体を理解している必要はありませんが，予期せぬ問題に出会ったときしばしばその知識が役に立ちます．整数の符号の有無による違いはハマりやすいところなので，一度意味を確認しておいた方が良いと思います．一般に用いられるIEEEの浮動小数点数形式[1]についても，計算精度と密接に関係することなので知っておいて損はないと思います．少なくとも収集したデータの数値がどの型で表現されたものであるのかは確認しておく必要があります．

またそれと同時に必要なのが，データファイルのフォーマットに関する情報です．バイナリ形式といっても，計測データだけが格納されているとは限らず，計測器の設定値などさまざまなパラメータが，データファイル自身の中に含まれているのがよくあるパターンです．これらのパラメータはファイルの先頭に書かれている場合が多く，これをヘッダ部と呼んだりします．独自にデータファイルのフォーマットを決めて収集システムを構築している場合は良いのですが，既製のシステムを使っていて，マニュアル等にもフォーマットに関する記載がない場合は，メーカーに問い合わせて教えてもらうか，自力で解読することになります．ヘッダ部はアスキー文字で書かれている場合が多いので，データファイルをエディタで開いてやれば，ヘッダ部がどのようなフォーマットになっているのかわかることもあります．

データファイルのフォーマットがわかれば，あとはその中から必要なデータを読み込んでやればよいわけですが，そのときに気をつけなくてはならないのはデータのバイトオーダーです．CPUによって，マルチバイトのデータの並び順が逆になっていることがあります．たとえばインテル系（Windows）とモトローラ系（Macintosh）がそうです．Windowsで作成したバイナリデータを何も考えずにMacintoshで読み込むと，意味を成さない値として解釈されます．逆もまた然りです．さきほど例として挙げたRubyでバイナリデータを読み込むときにはStringクラスのun-

＊1 http://www.perl.org/

＊2 http://www.ruby-lang.org/

pack メソッドを使うことになると思いますが，unpack メソッドは読み込もうとするデータがリトルエンディアン（Windows）かビッグエンディアン（Macintosh）かを指定することができるのでとても便利です．また，最初からさまざまな CPU が混在するような状況が予想されるならば，たとえば，NetCDF*3のような，機種に依存しないフォーマットでデータを取り扱うためのライブラリを採用するべきかもしれません．

さて，必要なデータをプログラム中の変数に読み込むことができれば，あとはグラフ描画ソフトが対応している形式でファイルに書き出してやるだけです．もっとも基本的な形は X 軸，Y 軸に相当する値を縦 2 列に並べるものでしょう．ただ，データの大きさが問題になることもあります．核融合研の LHD で標準的に使われている CAMAC のデジタイザは，チャンネルあたり256キロバイトのメモリを持ち，1 回の放電で10万点以上のデータを吐き出します．これをそのままアスキー形式のファイルに落とせば10万行のファイルができあがります．本格的な解析をするのではなく，データをちょっと眺めたい，というときに10万行のファイルを扱うのは，現在の高性能 PC をもってしてもちょっと重い作業になります．快適なデータ閲覧のためには間引くとか，全データの中で注目している部分だけ取り出すとかの工夫が必要です．このようなデータのリフォーマットもフィルタプログラムの中に簡単に取り入れることができます．

できあがったデータをグラフ描画ソフトに読み込んで表示し加工する方法はそれぞれの描画ソフトに依存する話なのでここでは深くは言及しませんが，今考えているような簡単にすばやくデータを閲覧する目的には Gnuplot*4がよく利用されています．Gnuplot では OS のコマンドとパイプ機能を利用すると，かなり柔軟にファイル中のデータを操作してグラフ化することができます．詳細な使い方は本誌記事[2]等を参照してください．

Gnuplot はすばやいデータのグラフ化には向いていますが，いくつかのデータを並べて表示したり，論文投稿用に見栄えのするグラフを作るのは簡単ではありません．そのような目的には次節で紹介されるような商用のソフトが使われることが多いようです．しかし例えば，Grace*5はフリーですが，データを読み込んで GUI 操作によりグラフ加工をするという普通の使い方もできるし，その一方で，C や FORTRAN プログラムから呼び出してグラフを表示させたり，読み込んだデータセットに対してユーザープログラムをフィルタ的に作用させたりという玄人的な使い方も可能で，商用ソフトにも負けないほど高い機能を持っています．

## 4.3 データ解析とソフトウエア

数値データを解析するための専用のアプリケーションプログラムを分類すれば Fig. 2 のようになるでしょうか．データ解析を試行錯誤しながら行うときには，データをグラフとして可視化しつつ解析するのが便利ですし，結論を最終的にグラフの形で提示したいわけで，データ処理用のソフトウェアとデータの可視化用のソフトウェアとは近い関係にあり，簡単には分けることはできません．それでも，それぞれのソフトウエアの最終的な目的が「美しいグラフを作る」ところにあるか，「データを解析する」ところにあるかで分類してみました．別の評価軸として，ソフトの操作をマウスなどのポインティングデバイスからの操作（GUI）で行うか，キーボードからの操作（CUI）主体で行うかでも分類してあります．図中イタリックで表示したソフトウエアは非商用のもので基本的にはフリーに配布されています．

汎用のコンピュータ言語ももちろんデータ解析に使えます．解析に必要な数値演算ライブラリの入手性から，Fortran77，Fortran 90/95，C，C＋＋，Java あたりが有力な候補になります．前節でも説明のあったインタプリタ言語（スクリプト言語）も使えます．プログラムの開発効率が高く，数値演算ライブラリやグラフィック描画機能を拡張機能として持つことできますから有力な候補といえます．ファイルやディレクトリの操作や，データベースの操作が手軽に行えることはデータを整理する時に有用です．Perl，Ruby，Python が選択肢として考えられます．

アプリケーションがどのプラットフォームで動作するかも注意すべき点です．大半の商用のアプリケーションは，Windows あるいは Macintosh を対象としています．他方，フリーのソフトウエアは Linux や FreeBSD などを含む UNIX 系の OS 上で開発されていますが，Windows 用のバイナリの配布もよく行われていますし，Cygwin*6という Windows 上で GNU 関連のソフトウエアを動作させる環境を使うと Windows 上でコンパイルできるケースがほとんどです．Macintosh も OSX 以降では UNIX 系のソフトウエアは簡単に動作させることが出来ます．データ収集装置で取得されたデータはネットワーク経由でアクセスできるケースが多いでしょうから，データ収集時と異なる環境で解析を行なうことも問題ありません．データ解析がもっともやりやすく，好みのアプリケーションが動作する環境を選ぶべきです．

これらのツールのどれを使うかは必要な解析方法によって決まるはずです．次節からはここにあげた各種のツールの特長を述べ，読者が自分のためのツールを選ぶ手がかりを提供したいと思います．

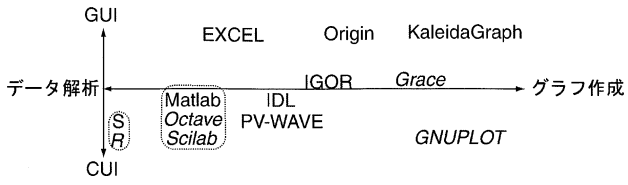


Fig. 2 データ解析用アプリケーションの分類図

＊3 http://my.unidata.ucar.edu/content/software/netcdf/index.html
＊4 http://www.gnuplot.info/
＊5 http://plasma-gate.weizmann.ac.il/Grace/
＊6 http://www.cygwin.com/

#### 4.3.1 専用アプリケーションでの解析

Fig. 2の右上に位置するアプリケーション，例えばKaleidagraph*[7]，Origin*[8]は，商用のグラフ作成プログラムです．論文に使うような高品質のグラフを作成することが主目的ですが，読み込んだデータに対して簡単な解析を行うことができます．アプリケーションによってできることは異なりますが，関数をデータに適用して加工すること，信号間の相関係数を算出すること，任意の関数を最小二乗法でフィットすること，Fast Fourier Transform（FFT）を使ってスペクトルを求めることなどは簡単に行えます．データ解析がこの範囲で済むならば十分な機能を持ちます．データは表計算ソフトウエアと同じで二次元の表の形で取り扱われますので，それ以上の次元を持つようなデータ（二次元画像データの時間変化など）の解析には向いていません．各操作がGUIで行なわれるため操作が容易な反面，同じ処理を多数のデータセットに対して適用する場合には，マウスで同じ操作を繰り返す必要があって面倒なことになります．マクロ言語がある場合にはある程度カバーできますが，同じような処理の繰り返しが多い場合にはCUIによる操作を取りいれたGnuplotやIGOR*[9]の方が適しています．

もう少しデータ解析よりのアプリケーションとしては，Interactive Data Language（IDL）*[10]やMATLAB*[11]があり，広く使われています．これらは，行列やベクトルを変数に割り当ててそれらの間の計算を一命令で行なうことができるという特徴があります．このため行列演算などをループを使わずに書くことができるので，プログラムの見通しがよくなります．静電プローブのデータを例にとって，最小二乗法による関数の当てはめを行なってみましょう．

Fig. 3にデータ収集装置から読み込んだ生データを示します*[12]．データは12ビットのADCを使用して取得しています．−5 V − +5 Vを0 −4095にマップする設定です．Fig. 3(A)はPV-WAVEというアプリケーションを使って以下の命令で描きました．

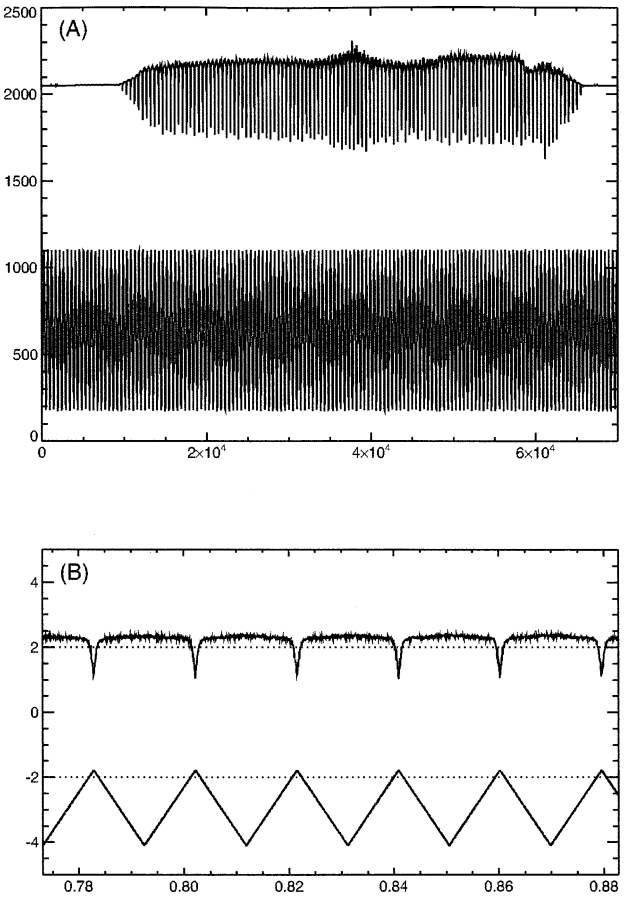


Fig. 3 時系列データの表示例．(A)に全時間，(B)にその拡大図を示す．

```
plot, cur0, nsum = 10, xstyle = 1
oplot, vol0 − 1024, nsum = 10
```

Fig. 3(B)に示すように整数を電圧に変換して，定常状態の時間帯を拡大表示してみます．List 1に使用したスクリプトを示しますが，配列全体への演算が1行で簡潔に書け

```
n11 = 2.0^11                                        ; -5V - +5V (12 bit)
cur = (float(cur0) - n11) / n11  * 5.0
vol = (float(vol0) - n11) / n11  * 5.0

time = findgen(1024*128L) * (1.0 / 25000.0)         ; 25kHz sampling

xr = [0.77316, 0.88284]                             ; time window for plotting
plot, time, cur + 2.0, xstyle = 1, xrange = xr, yrange = [-5, 5], ystyle = 1
oplot, time, vol - 2.0
oplot, [xr(0), xr(n_elements(xr)-1)], [2,2],linestyle = 1
oplot, [xr(0), xr(n_elements(xr)-1)], [-2,-2],linestyle = 1
```

List 1

＊7 http://www.hulinks.co.jp/software/kaleida/
＊8 http://www.lightstone.co.jp/products/origin/origin.htm
＊9 http://www.hulinks.co.jp/software/igor/
＊10 http://www.adamnet.co.jp/scs/products/idl/
＊11 http://www.cybernet.co.jp/matlab/
＊12 このデータはLHDのダイバータプローブのものです．

ることがわかります．このときサンプリング周波数の情報を使って時間軸も実際の時間に直してあります．

今度は Fig. 4 のように電圧・電流特性に静電プローブの理論曲線を当てはめてみます（List2）．

ここで，probe という関数（$= I_{is} - I_{es} \times \exp((V - V_s)/kT_e)$）を定義しています．このように任意関数への最小 2 乗法を使った当てはめや，そのチェックなどは対話型で簡単に操作でき，グラフ表示も容易に行えるため快適に解析できます．解析中の試行錯誤の過程がログに残るので，そのログを参照して処理法が固まったらプログラムの形で書き下すことができることは大きな長所で，開発のやりやすさが数値解析型インタプリタ言語の最大のメリットです．

インタプリタ型の言語ということで，処理速度を心配する向きもあるかもしれません．確かに，FortranやC言語といったコンパイラ型の言語とは比較できませんが，なるべくループを使わずに組み込みの演算を使うようにするなどのテクニックでかなり改善します．組み込み演算自体は定評のある数値計算ライブラリを使用しており信頼できるうえ速度も高速です．

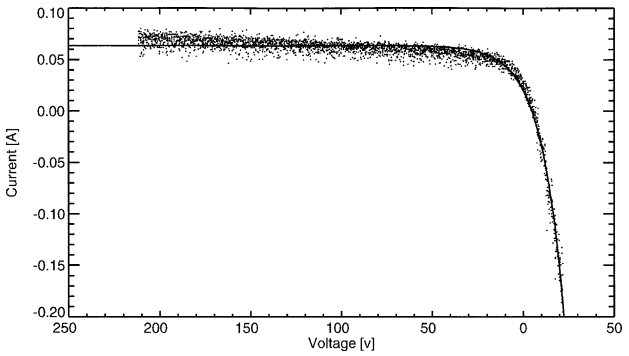


Fig. 4 静電プローブの測定信号への関数の当てはめの例

筆者は核融合プラズマの揺動データの解析を行なっていて，128 KWord×200 ch 程度のデータを PV-WAVE を使って日常的に処理していますが，速度に大きな不満を感じたことはありません．

数値演算型のインタプリタはいろいろな種類があってそれぞれ違った長所があります．ここで使った PV-WAVE*13 は IDL と先祖を同じくするプログラムで，文法にはかなり共通点があります．可視化や GUI への対応といった点では異なった発展をとげています．MATLAB はより行列演算志向が強い人気のあるソフトウエアです．かなり高価なため，同じような機能をもったフリーのソフトが作成されています．Octave*14 や Scilab*15 が有名で，MATLAB 用に開発されたパッケージプログラムが使用できなかったり，可視化の能力が本家 MATLAB には及ばなかったりという欠点はありますが，行列演算の処理では遜色ない性能を持っているようです．S 言語*16 も数値処理型インタプリタ言語のひとつですが，少し違った方向性をもち統計処理に強いのが特長です．文法は C 言語風で Fortran に類似した IDL/PV-WAVE よりこちらを好む人も多いかと思います．S 言語に対しても，R 言語*17 が開発されていてフリーに配布されています．

これ以外にも 3 次元データの表示に強い Yorick*18，C＋＋インタプリタをベースにした Root*19 など，強力な機能を持ったものが続々開発されています．

### 4. 3. 2 汎用のプログラミング言語による解析

次は汎用のプログラミング言語によって解析する場合です．処理すべきデータが膨大であったり，処理内容が複雑

```
function probe, x, param                                ; function to be fitted
   return, param(0) + param(1) * exp( (x - param(2)) / param(3) )
end

@stat_startup                                           ; using IMSL stat library
x = vol(mask) * 1010.0 / 10.0                           ; applied voltagbe
y = cur(mask) / 0.5 / 10.0                              ; probe current
plot, x, y, psym = 3, xtitle = 'Voltage [v]', ytitle = 'Current [A]'

mx = findgen(100)/ 100 * 300. - 250.
param = [100.0, -100.0, 0.0, 10]                        ; initial guess
n_param = 4
param = NONLINREGRESS("probe", n_param, x, y, Theta_Guess = param)
                                                        ; non-linear fitting
my = probe(mx, param)
oplot, mx, my                                           ; plot fitted curve
```

List 2

*13 http://www.vnij.com/products/wave/index.html
*14 http://www.octave.org/
*15 http://www.scilab.org/
*16 http://s.isac.co.jp/, http://www.msi.co.jp/splus/
*17 http://www.r-project.org/
*18 http://www.maumae.net/yorick/doc/index.html
*19 http://root.cern.ch/

で時間がかかる場合には，実行速度を稼ぐためにいったん実行形式にまでコンパイルされる言語を使うべきです．インタプリタ言語と異なり，実行途中で停止させたり，変数の値を調べることなどが難しくなるため開発に手間はかかりますが，高速な処理が期待できます．

数値解析をする上では，行列演算やフーリエ変換といった定番の処理が必要です．これらを自力でプログラムするのはアルゴリズムの勉強にはなりますが，信頼性が低く速度も十分に出ないおそれがありますので，十分テストされている「枯れた」ライブラリを使うべきです．フリーのものでは，行列演算用の Linear Algebra Package（LAPACK）*[20]，FFT 演算用の FFTPACK*[21]，C 言語用の科学技術用の演算ライブラリ群である GSL*[22]などが良く使われています．商用では IMSL*[23]，NAG*[24]，SSL II*[25]などの総合的な数値演算ライブラリがあります．IMSL と NAG はそれぞれPV-WAVEとMATLABの下位ルーチンとしても利用されています．

データ解析に使う言語を選ぶときには上記のライブラリへのインターフェースが用意されているものを使うべきでしょう．たとえば，LAPACK はもともと FORTRAN77でかかれたライブラリですが，f2c（Fortran から C 言語へのトランスレータ）を利用して，C 言語からの利用が可能な CLAPACK*[26]というライブラリが作られていますし，Lapack＋＋*[27]という C＋＋言語へのインターフェースもあります．ですから，Lapack をつかう場合には Fortran，C，C＋＋の中から使用言語を選んでおくのが無難です．Fortran には FORTRAN77と Fortran90，Fortran95という異なった規格があります．Fortran はもっとも古いコンピュータ言語のひとつで，77は70年代に規格が決められたため言語仕様が古くなりました．構造化をサポートする機能が不十分で読みにくいプログラムが書けてしまうので，新たに学ぶのであれば新しい規格である Fortran95を選ぶべきです．他方，FORTRAN77のプログラム資産は膨大ですから既存のプログラムやライブラリを利用するという意味で FORTRAN77を使う機会は多いかもしれません．Fortran のフリーのコンパイラとしては FORTRAN77の g77*[28]，Fortran95の，gfortran*[29]，f 言語*[30]などがあります．商用の処理系としては，Intel Visual Fortran*[31]，富士通の Linux 用の Fortran & C Package*[32]があげられます．

C 言語はもともと OS を記述するために開発された言語で，Fortran より自由度が高く汎用性の高いプログラム言語です．ベクトルや，行列を多用するプログラムには Fortran ほどは適していません*[33]．C＋＋は C を拡張したオブジェクト指向言語のひとつで，うまく設計された数値演算クラスと組み合わせれば，行列やベクトルをオブジェクトとして扱って見通しの良いプログラムが作成可能な反面，Fortran のようなベクトルマシン用の高度な最適化は難しいのではないかといわれています．その意味で処理速度がもっとも重要となるようなデータ解析では選びにくいところもあります．そんなわけで C も C＋＋もデータ解析に最適の言語とはいえませんが，GNU project*[34]の開発している GCC が使えることは大きなメリットです．GCC はフリーの処理系ですが，Linux や FreeBSD などの UNIX 系の OS 上で大変広く使われているため十分なテストが行われています．この点でフリーの Fortran 処理系より安心感があり，フリーの処理系を使う場合には C/C＋＋も良い選択肢と思います．前述の Windows 上でも Unix-like な環境を構築する Cygwin を利用すれば GCC を Windows 上で使用することもできます．フリーではありませんが Windows 用で無償で配布されている Borland C++ Compiler5.5*[35]もあります．商用の処理系としては Microsoft Visual C＋＋．NET 2003*[36]，Intel C compiler*[37]などがあげられます．

商用の処理系では統合型の開発環境を使って快適な開発ができますし，intel 製コンパイラの計算速度などはフリーの処理系をかなり上回るようです．

データ処理結果をグラフの形で出力するには，グラフィックライブラリを使用することになります．商用の製品にはライブラリが付属していることが多いですが，フリーの処理系の場合には好みのものを自前でインストールすることになります．グラフィックライブラリは非常に多くの種類があり，本稿ではとても紹介しきれませんし，筆者も全貌を把握していません．PLplot*[38]，三次元の出力用の vtk*[39]をあげておきます．これら可視化関連のツールについては，本誌記事[2]とそのサポートページ*[40]が大変参考になります．

直接グラフィックライブラリを使用しなくとも，コンパイラ言語の出力を一度データファイルにおとせば，前節で述べたようなグラフ化のソフトが使えます．筆者は前述の記事[2]を参考に C 言語で作った解析プログラムから外部の Gnuplot を呼び出して，中間データのチェックをよく行

＊20 http://www.netlib.org/lapack/
＊21 http://www.netlib.org/fftpack/
＊22 http://www.gnu.org/software/gsl/
＊23 http://www.vnij.com/products/imsl/index.html
＊24 http://www.nag-j.co.jp/index.html
＊25 http://www.fqs.co.jp/fort-c/products.html
＊26 http://www.netlib.org/clapack/
＊27 http://math.nist.gov/lapack++/
＊28 http://www.gnu.org/software/fortran/fortran.html
＊29 http://gfortran.org/
＊30 http://www.fortran.com/F/
＊31 http://www.xlsoft.com/jp/products/intel/compilers/iftnwin.html
＊32 http://www.fqs.co.jp/fort-c/index.html
＊33 例えば要素数が可変の二次元配列を関数に渡す時に見やすくすっきり書くことが難しい．
＊34 http://www.gnu.org/
＊35 http://www.borland.co.jp/cppbuilder/freecompiler/
＊36 http://www.microsoft.com/japan/msdn/visualc/
＊37 http://www.xlsoft.com/jp/products/intel/compilers/index.html
＊38 http://plplot.sourceforge.net/examples/index.html
＊39 http://public.kitware.com/VTK/
＊40 http://ayapin.film.s.dendai.ac.jp/~matuda/PlotUtils/index.html

なっています．外部ツールなしでもデバッガを使えば，プログラムの実行中に止めて変数の値を調べることができるわけですが，データがうまく処理されているかを調べるのに大きな配列の値をチェックしたい場合が多く，可視化した上でチェックしたほうが簡単にできるわけです．

さらに解析にメモリ容量やスピードが必要な場合には大型コンピュータの使用が考えられます．現在では解析用のデータをネットワーク経由で各大学のコンピュータセンターに送ることは容易ですし，ネットワークでつながれた多数のコンピュータの資源を透過的に使える GRID コンピューティングの技術も進歩しつつあります．国内でも ITBL*41等のプロジェクトが立ち上がっており，研究室から大規模な計算資源を容易に使える時代に入りつつあるといえるでしょう．

### 4.3.3 バージョン管理システムの利用

おすすめしたいのは，解析用のプログラムを開発しているときにバージョン管理システム*42を導入してソースコードを管理することです．バージョン管理システムは，ファイルの差分を記録していくもので，プログラムの開発に伴い節目節目で記録していくことで，開発の過程を記録することができます．解析プログラムは日々進歩するものですし，その間にバグが入り込むこともあります．以前に解析したデータがどのプログラムで行われたかを記録しおくことは絶対に必要です．バージョン管理システムでは解析プログラムを，「昔のある時点のもの」に戻すことが簡単にできますから，なにかおかしな結果が出たときに原因を追求することが簡単にできます．

また，動作しているプログラムを改良するために書き換えるのは気が重いもので，オリジナルをとっておいて，少しだけ内容が異なった多数のプログラムを作ってしまう（analyze → analyze1 → newanalyze1 といったファイル名が同じディレクトリ内に散乱したり）ことがありますが，昔の状態にいつでも戻せるという安心感からプログラムを大胆に改良していくことができ，結果的に見通しの良いプログラミングができるようです．バージョン管理システムは多人数での開発にも向いています．研究室で複数の人によってプログラムを開発する時も，履歴を記録していくことができるため威力を発揮します．

## 4.4 まとめ

科学実験の目的は結果が完全には予想できないことをやるわけですから，試行錯誤がやりやすい環境で作業することが良いと思います．この意味ではインタプリタ型の数値データ処理言語はなかなか優れた特徴をもっていて，データの表示，解析，図面化が１つのアプリケーションでかなりカバーできます．コンパイラ型言語を利用する場合には数値演算ライブラリの選択が重要な要素になるでしょう．

ダイバータプローブのデータの使用を許可いただいた核融合科学研究所の増崎　貴博士に感謝いたします．

### 参 考 文 献

［1］ S.A.T. William H. Press, Brian P. Flannery and W.T. Vetterling, *Numerical Recipes in C : The Art of Scientific Computing,* Cambridge Univer-sity Press, 1992.
［2］ N. Matsuda, J. Plasma Fusion Res. **78**, 144 (2002).

大舘　暁（おおだち　さとし）

1963年５月17日生まれ．1994年東京大学理学系研究科物理学専攻単位取得退学．同年４月核融合科学研究所に就職（助手）．軟Ｘ線計測による MHD 研究，揺動データの解析法などの研究を行なってきた．博士（工学）．趣味は濫読，僻地への旅行など．

後藤基志（ごとうもとし）

主な研究分野：専門はプラズマ分光学．LHD　において分光計測を担当しています．

＊41 http://www-riken.riken.go.jp/

＊42 Concurrent versions system (CVS) http://www.gnu.org/software/cvs/, Subversion (SVN) http://subversion.tigris.org/ などがよく使われています．筆者は CVS を使用しています．