実践論文

GTK+によるインタラクティブなデータ解析ソフトウェアの開発

Develpment of an Interactive Data Analysis System using GTK+

中村健治[*1]
Kenji Nakamura

Email: k13002@dokkyo.ac.jp

キーワード：データ解析、リナックス、GTK+、グラフィカルユーザインターフェース
Keywords: Data analysis, Linux, GTK+, GUI

多数の降水レーダの観測事例からなるデータをインタラクティブに解析するために、GTK+ (GIMP Toolkit)を使ったグラフィカルユーザーインターフェース(GUI)を持つソフトウェアを開発した。この解析では事例毎に結果を見ながらチューニングしなければならないパラメータがある。このためインタラクティブな解析ソフトウェアが必要となった。この開発により、効率的な作業ができるようになった。開発の中で、手続き型のプログラミングと event-driven 型のプログラミングの差異が明瞭に現れた。通常のデータ解析は手続き型であり、解析者は手続き型のプログラミングに慣れているので、手続き型プログラミングの中で GUI が構築できるようなソフトウェアの必要性が認められた。

An interactive data analysis system is developed using GTK+ (GIMP Tookit). GTK+ is a popular toolkit for development of a graphical user interface (GUI) used in Linux systems. The purpose of the system is to analyze radar observation data of precipitation effectively for many cases. The observation parameters are different case by case. The parameters are roughly known but they need to be more precisely determined. The results are subjectively checked by varing the parameters. The developed system works well and is very helpful for the data analysis. In developing the system, the program in the so-called event-driven type is found very different from that of the procedural type. Since many data analyses are with procedural programming and many data analysists are familiar with procedural programming, procedural programs including GUI may be desirable

*1：獨協大学

１．はじめに

衛星による地球観測は地球環境監視において現在必須の技術となっている。衛星上の測器の観測データから様々なアルゴリズムを適用して実際に必要とされる２次データが作成される。この２次データは宇宙機関で品質管理され、また明確なフォーマットのもとで「製品」として提供される。この２次データの精度検定、また２次データの導出アルゴリズムの向上のためには、地上での検証実験が必要である。衛星データは精度保証のついた均一かつ大量のデータである一方、地上実験のデータは必ずしも綺麗なデータとはなっていない。

現在、地上で降水特に雪のレーダ観測データを解析している。この実験では二つのレーダを対向させてその間に来た降水システムを双方向から観測している。この際、二つのレーダの方向を 0.1 度以内とかなり厳密に合わせる必要がある。ところが現場ではここまでの精密な方向合わせができず、0.2 度レベルにとどまっている。このため、データ処理の段階で、データを少し変更してデータ上で疑似的に方向合わせを行っている。この時、正確な角度は未知であるため、データ処理では解析結果を眺めながら、角度等のパラメータを主観的にチューニングして決める作業が必要となっている。二つのレーダの距離間隔は地図上では正確に分かるが、レーダのデータに現れる距離はレーダの送受信機とデータ処理部との間のケーブル長さ（約 50m）などがあり、不確定要素があるためこれもチューニングパラメータの一つとなっている。さらにはレーダ受信機の雑音の差引き処理の有無やデータ品質レベルの閾値もパラメータとなっている。

このチューニングは、これまではパラメータを変えてその都度画像を出力し、それを並べ、あるいはプリントして眺めるという作業となっていた。2011 年 4 月の測器の完成から 2015 年 3 月まで、総計 400 日以上の観測日の内、100 日以上の降水事例がある。これらを目で確認しながら効率的にチューニングしなければならない。この作業を GUI（Graphical User Interface）を使いインタラクティブな作業にして効率的に行うことを考えた。GUI での想定画面はパラメータの入力画面（スライドバー、スケールバー）と結果の出力画面である。

データ解析及び画像化プログラムは既に C 言語で作成してあるので、GUI の部分の作成が新たに必要であった。この部分の制作を目的とした。なお、これは定型的なやり方でもあるのでここでの経験と知識は Microsoft Windows 上での Visual Basic などによる制作、また他の GUI の制作などの理解にも通じるものがあり、GUI 一般の理解も目的の一つであった。

データ解析でプログラムを作ることは極く当たり前のことであるが、そこでは GUI は特には必要としない。GUI は他人が使うことを想定し user friendly なアクセスを実現するために作られることが多いと思われる。しかし今回の GUI の制作は、データ解析のためである。この制作過程で、データ解析者のための GUI というカテゴリの必要性が認められた。

本稿では、GUI の制作の動機と制作した GUI の概要を記す。そしてデータ解析者のための GUI の在り方について考える。

２．観測データと解析

本節では制作した GUI が目標としているデータ解析について述べる。

日本時間 2014 年 2 月 28 日に衛星による全球降水観測計画（GPM：Global Precipitation Measurement）の主衛星が宇宙航空研究開発機構（JAXA）の種子島宇宙センターから打ち上げられた。GPM は日米が主導する国際協力の計画であり、主衛星と多数の副衛星からなる衛星システムである。主衛星には日本の開発した 14/35GHz（電波波長 2.2cm/0.8cm）帯の降水レーダ（DPR）と米国の開発したマイクロ波放射計が搭載されている。これは日米協力により 1997 年 11 月に打ち上げられて以来 17 年以上にわたり観測を続けた熱帯降雨観測衛星（TRMM：Tropical Rainfall Measurement）の後継計画である。DPR は２周波数を使うことにより雨と雪の識別や高精度の降水強度推定を目的としている。この DPR による降水強度推定法の開発のためには降水粒子による 35GHz 帯（Ka 帯）の電波の散乱特性が必要である。雪などの固体降水粒子、特に融解途中の粒子の 35GHz 帯電波の散乱特性は実験的結果に乏しい。この散乱特性を実験的に測定するために JAXA は地上観測用の Ka帯レーダシステムを開発した。

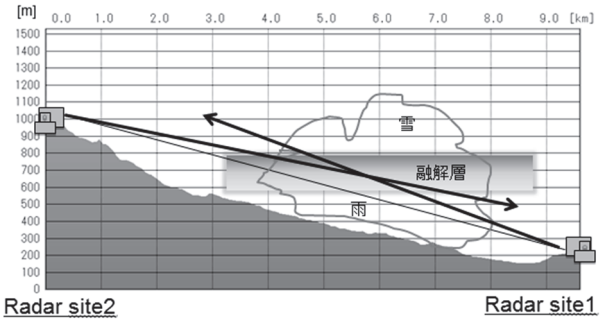

図１　蔵王の斜面での観測形態。二つのレーダをほぼ対向させて、間にある降水システムを観測する。

このシステムは同じ仕様の二つの Ka 帯レーダからなり、後に行った蔵王での観測の模式図（図１）にあるように、対向させてレーダの間に来た降水システムを同時に観測して、降水による等価レーダ反射因子と降雨減衰係数を同時に測定できるところに特長がある。

2011 年 3 月のシステムの完成後、4 月から 7 月までは情報通信研究機構（NICT）の沖縄の施設と沖縄科学技術大学院大学（OIST）の建物屋上にレーダを設置して観測を行った。図２は 2011 年 5 月に行われた NICT 沖縄と OIST との間での対向観測実験の結果を示す。二つのレーダの間の距離は 4.25km であり、この時の経験からこの後の観測は二つのレーダ間の距離は 10km 以内、距離分解能は 12.5m が適当とされた。図２（左図）に良いイベントであった 5 月 24 日夜から 25 日朝の降雨

エコーの一部（24日夜21時から22時）を示す。この時、途中の地上測器による最大降雨強度は25mm/h程度と強い雨であった。二つのKaレーダのエコーは同様の傾きを持ったパターンを示しており、降雨が南のOIST側から北のNICT沖縄側へ約5m/sの視線速度で動いたことが分かる。またレーダから離れるにしたがってエコー強度が下がっており、Ka帯電波の降雨減衰が顕著に表れていることも分かる。21:40の時の測定エコー強度も図２（右図）に示す。二つのKaレーダの降雨減衰量は距離についてほぼ単調増加（減少）であり、本システムにより降雨減衰が測定されていることの傍証となっている。この降雨減衰量の距離微分から降雨減衰係数が推定される。

沖縄での観測結果を踏まえ、その後、固体降水を観測目標として、富士山北西斜面、長岡、札幌、そして蔵王山の西側斜面で観測を行ってきた[1]。

山形県蔵王山の斜面での観測目標は融解層である（図１）。融解層とは降水粒子が雪や氷晶などの固体粒子が周囲の気温の上昇から水粒子に変化する層であり、誘電率や形状の変化から電波散乱特性に大きな変化が見られる層である。ところが融解層観測では降水によるレーダエコーが大きく変化することによる誤差が大きいことが分かってきた。この誤差は二つのレーダのビームが一致していれば小さいものとなるが、実際には対向しているレーダの送信波がもう一方のレーダに入るため、ビームを一致させることができず、0.5度程度ずらすようにしている。降水の水平一様性あるいは鉛直一様性を仮定することによりこのずれによる誤差はデータ処理により軽減できる。通常の対向観測では鉛直一様性を仮定するが、融解層では鉛直方向のエコー強度の変化が大きいため、水平一様性を仮定せざるをえない。二つのレーダのビームの仰角は小さいので、わずかな仰角の差異が水平方向（距離方向）には大きな差となってしまう。レーダの設置に当たっては角度などをできるだけ正確に設置するが、それでも0.2度以下の精度はない。このためデータ解析の段階でレーダビームの仰角を解析する際のパラメータとして距離をずらすなどしてチューニングする必要がある。レーダビームの仰角は頻繁に変更するので、チューニングも事例毎に行う必要がある。

図３にはチューニングの例を示す。図の横軸は時間で８時間分である。縦軸は距離であり、この図の場合は9.6kmである。図が上から３つあるが、上と下の図にはほぼ横に縞模様が現れている。これは疑似模様であり、この模様はパラメータをチューニングすると中の図のように消える。

３．データセット

解析するデータは各観測期間に分かれる。観測場所は沖縄に始まり、蔵王まで、さらには観測船に搭載しての観測などいくつもあるが、二つのレーダの対向観測は、沖縄、富士山北西斜面、長岡、札幌、蔵王の５か所である。レーダは可搬であるとはいえ、一旦設置すると当該観測期間中は動かさない。レーダアンテナの仰角は手動で変えることができる。観測ではレーダアンテナを鉛直上方に向けた観測なども交えているので、同じ対向観測期間内でも仰角は細かく変化している。方位角は設置時でほとんど決まっている。このような設置条件から、各場所における観測パラメータの標準値はそれぞれ決まっている。これを踏まえて、データファイルは沖縄など各観測場所によりdirectoryに分けた。さらに降水のある日のみを集めたdirectoryを作った。ファイル名は例えばZao2-20141114-V1.bzmsなどとなっている。最初の「Zao2」は蔵王での２回目の冬の観測を、次の数字は2014年11月14日を示す。V1はversion１を示し、「bzms」はデータに適当につけた種別名であるが、一応、bはbothで対向観測の意、zはレーダで測られる降水エコー強度はZ因子と呼ばれることから、mはmeasured、sは複数の意味を持たせている。このように名前をつけることにより、ファイル名から観測パラメータの標準値を選択することができる。もちろん年月日から判別することもできるが、Zao2などと記されている方が分かり易い。また文字ソートすることにより、日付順に並ぶ。

４．プラットフォームと制作物

UNIX（Linux）上のX-Windowで行う。システムは組立てPCのUNIXマシンである[2]。OSはRed Hat系LinuxであるCentOSであり、標準のシェルはbash（Bourne Again Shell）である。巷でよく使われるLinuxはUbuntuというMS Windowsに良く似ておりWindowsの代替えを狙っているものであるがそれではない。言語はC言語とX-Window上でGUIを構築するための道具であるGTK+（GIMP（GNU Image Manipulation Program）Toolkit）を使う[3]-[6]。名称は元のGTKにオブジェクト指向の階層構造が加わりGTK+となったらしく「＋」を付けることが正式のようである[5]。Linux PCの代表的な画面表示にはKDEとGNOMEがあるが、GTK+はGNOMEの基礎となっている道具である。なおKDEではQtと呼ばれる道具が使われている。GTK+をより効率的に使用するためにGladeというソフトウェアもある。これらはすべてGPL（GNU General Public License）あるいはそれに類似の規約の下にありフリーでダウンロードして使用することができる。GTK+の最新のversionは3.20であるが、現在、GTK+のversion 2.とGladeのversion 3をPCにインストールしている。Version３はUbuntuでは簡単にインストールできる、あるいはデフォルトで入っているようであるが、CentOS version 6へのインストールは厄介なようなので、version 2であるGTK+2.0を使っている。なお、CentOS version 7ではGTK+3を使っているようである。

制作前に想定したGUI画面を図４に示す。パラメータの入力にはslide barを使う。またパラメータの説明として観測の模式図を表示して、slide barを関連する場所の近くに置くことを想定した。上部には処理したファイル名とともにパラメータを数値で表示する。下側にはscroll bar付きで処理画像を表示する。

実際に制作したGUI画面を図５に示す。GTK+ではボタンなどの画面上の個々のモジュールはwidgetと呼ばれる。Widgetという言葉はWindow gadgetからと言わ

れている。大枠のwindowは既存のwidgetをそのまま使っており終了ボタンや縮小ボタンなどがついている。最上段にはwindowの説明としてプログラム名と画像名を表示している。その下にはFile openボタンとPrev/Nextボタンがある。File openをクリックすることでファイル選択dialogが開く。Prev/Nextは選択されたデータファイルと同じdirectoryにあるその前後のファイルを選択するためのボタンである。ファイル選択を行うと、ファイル名からパラメータのあらかじめ与えられている初期値を使って画像が表示される。データの処理結果は図の中にある二つの画像であり、上の画像は指定されたチューニングパラメータを用いて計算された減衰係数の時間（横軸）距離（縦軸）断面であり、下は減衰補正されたレーダエコーの時間距離断面である。その下には各種の変更パラメータがspin buttonとcheck buttonにより与えられている。この場合は二つのレーダ間の距離、二つのレーダを結ぶ線の仰角、それぞれのレーダビームの仰角、降雨減衰を計算するときの差分距離、データ選択のための閾値、雑音レベル差し引きの有無、の７つのパラメータである。さらに二つの画像の色調整用のspin buttonを表示している。色調整はカラーバーの上限と下限、また表示有無の上限と下限で行う。画像の詳細を調べるには前者のカラーバーの上限、下限を調整することで行うことができる。後者はデータが範囲を外れるとその部分のデータを表示しない、実際には白あるいは黒にしてしまうためのものである。こうすると、雑音レベルにかかりそうなデータの除去などのデータの品質管理は後者でも行うことができる。また画像内のデータの値がカラーバーでは不明確であるが、後者の上限、下限を変化させることにより値を知ることができるという効果もある。最下段にはFile openで指定したデータファイル、あらかじめ与えられている初期値、そして終了ボタンがある。Spin buttonはクリックする度に数値を設定したステップ幅で変化させることができ、これにより画像が変化し、結果を見ながらパラメータをチューニングすることができる。当初はslide barを使うことを考えたが、数値を正確に与えるにはspin buttonがより適当であった。

５．プログラムの書き方

GTK+は、プログラムの書き方がトップレベルで異なる。通常のデータ解析プログラムは手続き型であり、途中の様々な分岐はあっても一本の流れである。一旦プログラムを走らすと、その後は初期に入力を要求することはあっても、自動的に走り、最後にきちんと終了する。プログラムは処理の流れがそのまま構造となり、画像を出力するような場合は、最後に画像作成が副プログラムとして入る。これに対してGTK+によるプログラムはいわゆるevent-driven型のプログラムである。準備段階でボタン配置などの画面作成を行い、各ボタンがクリックされるとその都度呼び出されるcallback関数の中で実処理を行うように設定する。プログラムを起動した後はgtk_main() で常に画面監視を行う形となる。言い換えればデスクトップに現れるwindow等の作成が最初に来て、データの解析処理はボタンのクリックなどによって呼びだされる副プログラム内で行われる形となる。プログラムの終了は強制終了のような形となる。

作成したプログラムは約1000行である。主に菅谷（2004）[5]を参照した。その内容はおおよそ、コメント行が80行、データ処理本体が250行、画像化、これは既存のプログラムを外部参照しているだけであるが外部参照しているプログラムの中では100行である。残りの約700行はすべてGUIの記述となっている。プログラムの最初のコメントにはこれまでの開発履歴を各数行で記しているので、長くなっている。ここをみると開発期間に半年かかっていることなどがわかる。GUIの部分は、かなり重複しているが、ボタン等のwidgetの配置とボタン等に発生したeventに対する処理（callback関数）の記述に充てられている。実際の主副プログラム群を下に記す。

main
make_interface：widgetを配置する。
　Widgetにはwindow、entry button、check button、spin button、画像表示のためのscrolled window、ボタン等を横に配置するhbox、縦に配置するvboxなどがある。
callback関数群：それぞれのボタン等のすべてについて作成する。
file_read_and_preparation：指定されたファイルを読み込み、データを揃える。
cb_open:File openしたときのcallback関数であるが、データファイルリストを作るなど、ファイル関係の準備をおこなう。
kzGeneration：データから目標の物理量を計算する。
imageGenK, imageGenZ：計算された物理量からビットマップ型画像を作成する。

プログラムの動作は以下である。まずFile openでデータファイルを選択すると、ファイル名から二つのレーダの間隔やレーダアンテナビームの仰角などの標準のパラメータがセットされる。そしてデータを標準値で処理した画像が表示される。同時に各ボタン、また処理したデータファイル名も表示される。画面上には現れないが、Prev/Nextボタンがクリックされたときに、前後のデータファイルを同定するために、File openで選択したファイルの入っているdirectoryの中のファイル名をソートしたリストが作成される。Spin buttonでチューニングパラメータを変更すると、再計算した画像が再表示される。またカラースケールを変更した場合は、画像の色割り当てのみが変更されて表示される。

画像作成については、GTK+ではPostscript形式を扱うことができないため、bitmap形式のファイルを出力するプログラムを新たに開発した。既に開発したPostscriptファイルを出力するプログラムがあり[7]、それを参照している。具体的にはheaderとして縦横のピクセル数、ピクセル当たりのビット数やレイヤー数

を記し、その後、カラーのRGB値を流しこんでいる。画像ファイルは一旦出力して、その後、再度取り込んでいる。直接に取り込んだ方が速くなるとは考えられるが、慣れている既存のプログラムの形を踏襲した。ファイルとして出力すると、imageMajikなどのプログラムにより画像を確認することができるという副効果もある。

Widgetとcallback関数の部分の記述は定型的であり、より簡単な記述ができそうに思える。実際この部分に関してはGladeと呼ばれるGUIを利用した制作ソフトウェアがある。またGladeを使ったより統合されたシステムとしてAjutaと呼ばれるものがあるようだがこれについては試していない。

GUI型のプログラムにはオブジェクト型というもう一つの特徴がある。Widgetはオブジェクトとして内容は隠されており、事前に与えられている関数を使う。Widgetには様々な操作が考えられ、それぞれに関数が付随する。このため、関数を知ることがプログラム作成の効率にとって重要となる。例えばspin buttonでは作成するときに入力できる上限と下限を設定する。しかし、何もしないと、最小値が初期値となってしまう。これをデフォルト値に替えるにはgtk_spin_button_set_valueという関数を使う。このようにspin buttonの内容を知らずとも関数により操作ができる。これはwidgetを内容まで立ち入って自由に使うことはできない代わりに、widgetは安全となっており、安心して使うことができる。その一方、個々のwidgetの関数を知らなくてはならないため、manualなどを頻繁に参照しなければならない。他人の作成したソフトウェア群を使うには多数の副プログラムの名前と機能を知る必要のあることは当然であるが、それと同じことがGTK+にも言える。

データ処理・解析では、元データの構造を知れば十分である。元データはテキストならばそのままで、binaryならばそのフォーマットが分かればよい。HDFやNetCDFのようなmachine independentな形式の場合はそのためのtoolkitを必要とするが、知らなければならない副プログラムの数は、通常は読み出し部分のみであり少ない。処理そのものはC言語などの通常の言語でプログラムを組めばよい。このように通常のプログラムでは覚えなければならないあらかじめ定められている単語（予約語）は少ないが、GTK+では逆の状態となって多くの単語あるいは関数名を覚えなければならない。また望みの単語あるいは関数が無い場合はほとんどお手上げ状態となる。今回の開発でも、File openではGtkFileChooserと呼ばれる関数群を使っている。これを使うと極く普通のファイル選択のdialog画面が現れる。このobjectのプログラムは複雑な筈であるが、既存のwidgetとして極く簡単に呼び出すことができる。この時、「最近使ったファイル」群が自動的に最初に表示される。ここをプログラムを走らせた時のdirectoryの内容を表示させようとしたができなかった。GTK+のreferenceマニュアルを見てもそのような操作ができるかは不明であった。このため目標とするファイルを選択するためにはデータファイルのあるdirectoryを選ぶという操作が必要となっている。これはデータファイルを次々に処理しようとした時には煩わしいことになっている。その一方、ファイルを拡張子で判別し選択して表示する関数は準備されている。これは便利である。

Widget間の通信が簡単にはできないことは不便である。一つのentry widgetから数値等を入力し、その値により、spin buttonの初期値を変えたいのであるが簡単ではない。一旦windowを決めボタンの配置などを行ってしまうと後の変更は困難なようである。各widgetの識別子があればそのwidgetにアクセスすることができると思われるがそのような形にはなっていない。実際、通常のGUIの画面では、どこかのボタン等をクリックすると操作は行うことができるが、ボタン等が変更されるようなことは余り無い。結局、必要なspin buttonをgtk_widget_destroyという関数により一旦消去して再度作成表示する方法をとった。また副プログラム間のデータのやりとりも困難であり、結局global変数を多用する形となってしまっている。

6．GUIのチューニング

GUIを一応完成させた後、使い勝手の向上、新たなパラメータの入力機能、また画面の洗練化を図った。使い勝手については、マウスクリックまたentry boxへの入力回数をできるだけ少なくすることを心掛けた。Tuningパラメータの初期値は沖縄、富士山、長岡、札幌、蔵王と観測場所により異なる。開発段階ではコマンドラインから開始するときに引数として ”Okinawa” などと入れるように工夫した。しかしこうすると観測場所が異なる度にプログラムを再度走らせなければならない。これを避けるため、ファイル名から自動的に初期パラメータが設定されるようにした。またコマンドラインからの引数は極力少なくするために改訂した。このような細かい措置を施すことができるのも自前で開発しているためである。現在は新たなファイルを選択するときは、File open、directory選択の2回のシングルクリックとfile選択の1回のダブルクリックで目標のファイルからの画像を得ることができるようになった。さらに選択したファイルと同じdirectoryに入っているファイルについては、Prev/Nextのボタンをシングルクリックすることにより、すぐ前あるいは後のファイルを処理することができるようになっている。これにより非常にスムースに画像をチェックすることができるようになり、またストレスを感じさせなくなっている。高速化することにより、約100日分、約100個のデータファイルに含まれる事例の特徴も把握することができるようになった。以前はshell scriptにより自動処理で画像を作りだし、それをプリントして紙として積み上げていた。この作業は現在でも必要であるが、解析結果の検討のためにはGUI経由での作業のほうがはるかに効率が良い。さらにスムースに次々にデータを見ることができることにより、個々のデータのパラメータチューニングだけでなく、データ群全体をbrowsingできるようになった。そしてbrowsingの途中でもデータの細かい検討ができるようになった。

これまでは画像を多数作成しておいて、display 上でサムネイルの形で browsing していたが、この場合は適当な事例についてその場での細かい検討はできなかった。このように本ソフトウェアにより解析が非常にスムースかつ明確になった。

改良点としては、画像の枠、軸説明を入れる、カラーバーを表示する、処理ファイル名を大きく表示する、などがある。これらは GTK+ に含まれている cairo などのプログラム群を使うと可能であるが、これらは二義的なので改訂してはいない。不適切なファイルを読み込むと終了してしまう。これもメッセージ dialog を出すなどするとよいが、自分用の解析プログラムであるのでそこまでの手当は行っていない。

パラメータのチューニングでは spin button により次々にパラメータを変更する。この時、十分な速度が出ないと使用に耐えないものとなる。一つのファイルを一つのパラメータセットで処理して画像化する時間は 1 秒を十分に下回ることは事前に分かっていた。しかし、GUI を含めた状況下ではやってみなと分からない面があった。実際に行ってみると、spin button を連続してクリックしてパラメータを速く変更するとついてこなくなるものの、変更速度を少し緩めると追随するので十分に使用に耐えるものとなっている。画像のカラーレベルの変更では、基礎データは計算されており、色の割り当てを変更するだけなので計算量は少ない。このため、カラーレベルの変更では spin button を速く変更しても十分に追随しており、アニメーション的な動きもできる速度となっている。

加速するためには、画像ファイルのディスクへの出力を省略する、などの polish up も考えられるが、ソフトウェアは実用のレベルに達しているので、行っていない。

７．IDL の GUI

科学計算でかなりよく使われている商用の言語に IDL (Interactive Data Language) がある。ベクトル計算などのプログラムを簡便に書くことができ、また特殊関数、FFT や連立一次方程式系の LU 分解などの数学的関数の副プログラムや地図データなども備えている[8]。この IDL には GUI の機能も付加されている。比較のためこの IDL の GUI を少し試してみた。Widget の配置など、画面設定は GTK+と同様の構造であるが、ずっと簡便な書き方ができる。GTK+ のような細かい設定もできるとは思われるが、試してはいない。現れるウィンドウやボタンなどは見た目がきれいであり、商用であることを感じさせる。画面がきれいであることは、処理をスムースかつ楽しくさせる面があるので、多方面の応用を考えた場合には重要であることを認識させる。

IDL は強力なデータ可視化機能があり、3 次元画像の回転などが簡単にできる。このような計算機能をバックに持つと 3 次元で可視化してボタンにより回転などを行うことは有用でかつ IDL に合った機能であると思われる。その一方、商用のソフトウェアではよく見かける画像をマウスでドラッグして回転させることなどは、一見できないようである。できる可能性はあるが、IDL のマニュアルは膨大であるといわれており未確認である。

８．考察

インタラクティブな操作は Xerox の Palo Alto の研究所でアイコンとマウスによる GUI が考案され[9]、Apple 社の Macintosh により広まった。そして Microsoft Windows などで一般化し現在では web browser などでも極く当たり前の GUI となっている。その一方、データ解析の分野では、３次元画像を回転させたり、断面を表示するなどの定型的な処理では使われるものの、個々の解析処理一般では必ずしも広まっていない。今回作成したソフトウェアはデータ解析上の要求から必要に迫られて作成したものである。本来の目的は GUI を作ることではなく、データ処理を効率的に行うことである。Glade など GUI の作成を支援するシステム、統合環境として Anjuta と呼ばれるシステムもあるが、GUI 部分の作成を簡便化する道具が必要ではないかと考える。また event-driven 型と手続き型ではプログラムの書き方に大きな差異がある。手続き型では、処理の流れの中で、C 言語ならば printf や scanf などを使って入力要求すればよい。データ処理を中心とする立場からは、手続き型の構造の中に入出力のための GUI が入ってくることが望ましい。しかしこのような形のソフトウェアは余り無いようである。商用のデータ解析ソフトウェアである IDL には GUI 作成機能が入っているが、GUI を使用した場合には全体構造としてはevent-driven 型となっている。

GTK+はオブジェクト指向であるため、型チェックが厳しい。Compile 段階での警告で済むものが多いが、compile 段階ではじかれてしまうことも多い。型は多いため型変換をどのようにすればよいかを調べることも必ずしも簡単ではない。また多数の関数を覚える必要がある。このためか関数は、かなり読みやすくかつ系統的な名前となっている。関数名も長いが分かりやすいように作られている。少し慣れると関数名を想像し、検索して探すことができるようになる。数十年前の初期のプログラム言語では単語の長さ制限などがあり、記述は短いものの分かりやすいものではなかった。このため、フローチャートを示すなどして詳細な説明が必要であった。単語の長さの制限が緩くなったことで、プログラムは一見長くなるが、関数名が動作を示しており、プログラムを読んでいけばそのまま理解できる。例えば GTK+では

```
spinButton=gtk_spinbutton_new_with_range(0.0,
10.0,1.0);
gtk_spinbutton_set_value(spinButton, 5.0);
gtk_container_pack_start(hbox, spinButton, FALSE,
FALSE,0);
```

などとなっていれば、0 から 10 までの範囲で刻み 1 で動く spin button を新たに作り、初期値を 5.0 にセットして、しかるべき場所に置く、ということが分かる。

見栄えもある程度必要である。見栄えを良くすることは目的ではないが、作業を効率的に行うためには自然な形のボタン配置や揃えも必要である。不自然な形のGUIはパラメータ入力などで間違いを引き起こす。これはuser friendlyということとは少し異なるが、重要なことである。しかし、GUIを作ることが目的ではなく、GUIを使用してデータ解析を効率的に行うことが目的の場合には、GUIの部分の作成に時間がかかることは問題として残る。

本ソフトウェアによりデータ解析を行うと、解析の進展につれて入力パラメータの変更が入り、また入力パラメータの範囲などが限定されてくる。この状態になると、効率化のためにはパラメータの初期値などの設定変更が必要となる。これらもGUIが自作であるため、容易に変更することができる。作業での煩わしい点もある程度除去することができる。例えばFile openするとpop-up menuでファイルのリストが表示されるが、この時、元のwindowの中で表示しているファイル名が一部隠されてしまう。現在処理しているファイル名は次に処理するファイルの選択の時に必要となることが多いため、その度にいちいちwindowを動かさなくてはならず、これは煩わしい。Pop-up menuが表示される位置を変えると良いのであるが、その方法が見つからない。結局windowの中に表示されているファイル名を短くすることで対処している。このような細かいことも自作なので可能となっている。

GUIはいわゆるuser friendlyなアクセスを提供する。このためGUIによるプログラムは他人が使用することが前提となっていることが多いと思われる。しかし、データ解析をする立場からは手続き型のプログラミングの中で効率化のためGUIを使用したい。このための方向をいくつか考えてみる。

（1）GUIを前面に出したevent-driven型でプログラミングする。これはいわば通常型であり、本稿で記したソフトウェアを開発したやり方であり、既に実現されている。

（2）手続き型プログラムの中での入出力にGUI型を取り入れる。C言語ならば入出力やファイル選択にはscanfやprintf、fopenなどが使われる。このような関数で、spinButtonScanf や imagePrint、fileChooserのような名前でGUI型の入出力関数があるとよいと思われる。単純に行うと、複数の入力ボタン等がある場合には入力待ち状態が続いてしまうので、一つでも入力されたら処理が開始されるような工夫が必要であろう。なおプログラムの中ではwhile文を使うなどして入力待ちの状態を作ることになり、結局はevent-driven的なことにはなってくる。

（3）もう一つの方向としてGUIをとりいれたshell scriptが考えられよう。大量のデータファイルの連続処理では、shell scriptを書いて単独のデータファイルの処理プログラムに次々にファイルを与えて処理させることが行われる。単独のプログラムは必要なパラメータの入力を持ってデータ処理を行う。Shell scriptは必要なパラメータを揃えて単独プログラムを走らす。GUIを持ちインタラクティブに入出力を操作できるようなshell scriptがあれば、本体である単独プログラムをそのまま使うことができる。

処理の流れは実質的にはどれも同じとなるが、プログラミングの効率化の面で差異があろう。データ解析の立場からは（3）が望ましい。単独の処理プログラムの作成にもっとも時間を掛けるので、それをそのまま使用できるような入出力ができることが望ましい。

9．まとめ

多数の事例からなる観測データの解析のためにLinuxシステムで標準的に使われているGTK+を使ったインタラクティブなデータ解析ソフトウェアを作成した。利用形態に合わせるように何度も改訂を施して完成したソフトウェアは、速度、使い勝手とも使えるレベルに達した。各事例毎のパラメータチューニングだけでなく、多数の事例のbrowsingにも使えるようになり、browsing途中で事例の特徴把握もできるようになり、実効が上がっている。実際、それまで疑問であったことが、データの品質によることが分かり、品質チェックを厳しくすると、結果がかなり安定してくることが分かってきた。このようなことを踏まえて、統計的な解析を行う際のパラメータの決定を行うことができるレベルに達した。

プログラミングの過程では手続き型とevent-driven型のプログラミングのやり方の差異が認識された。データ解析では手続き型が通常の形である。一方GUIはそのuser-friendlyな性格から多数の人が使用することを前提としてプロが作成しているという現状があろう。個々のデータ解析のためにGUIを実装することはプログラミングの手間を考えると必ずしも適当とは思えない。今回もGUIの実装には半年近くかかっている。何も知らない状態から始めたことを考慮しても長い時間を費やしている。通常の手続き型のプログラミングの中でGUIを入れ込む形ができることが望ましい。

今回、GUI部分を自作した一つの意味として、自分の嗜好に合った解析作業が行える、ということもある。元々の目的である解析は、限定されかつ特殊なものであり、既存のソフトウェアにはうまく乗らない。また使う場合は、データの準備などが必要である。これらを自分の嗜好に合った形で処理できることは、解析作業のストレスを大きく減らしている。これは自作のソフトウェアには一般的にいえることであろう[7]。

謝辞

本稿で述べた作業は国立研究開発法人宇宙航空研究開発機構（JAXA）地球観測研究センターの第6回、第7回、第8回PMM公募研究の一部である。システムのバックにはデータサーバを置いているがこれは学術振興会科学研究補助金基盤（B）24340110により構築した。本研究の一部は情報科学研究所研究助成によっている。また査読者からは有益なコメントを頂いた。

参考文献

(1) Nishikawa, M, K. Nakamura, Y. Fujiyoshi, K. Nakagawa, H. Hanado, H. Minda, S. Nakai, T. Kumakura, and R. Oki, Radar Attenuation and Reflectivity Measurements of Snow with Dual Ka-band Radar. IEEE Trans. Geosci. Remote Sens. 54(2), 714-722, 10.1109/TGRS.2015.2464099, (2015).

(2) 中村健治、獨協大学での個人用衛星データ解析システム、情報学研究、獨協大学情報学研究所、第5号、69-77、(2016).

(3) 竹田英二、GTK+で始めるXプログラミング、(1999).

(4) たなかひろゆき、GTK+入門、第2版、ソフトバンクパブリッシング、(2002).

(5) 菅谷保之、入門GTK+　C言語で学ぶ、はじめてのGUIプログラミング、オーム社、(2009).
(Web版はhttp://www.iim.ics.tut.ac.jp/~sugaya/public.phpからdown loadできる。)

(6) 「素人の独学 GTK+3.0」、http://uchigo.main.jp /gtk3/index.html.

(7) 中村健治、データ描画用小型ソフトウェアの開発、情報学研究、獨協大学情報学研究所、第4号、19-26、(2015).

(8) 伊藤ただし、IDLプログラミング、講談社、(2013).

(9) マイケル・ヒルツィック、未来を作った人々、毎日コミュニケーションズ、(2001).

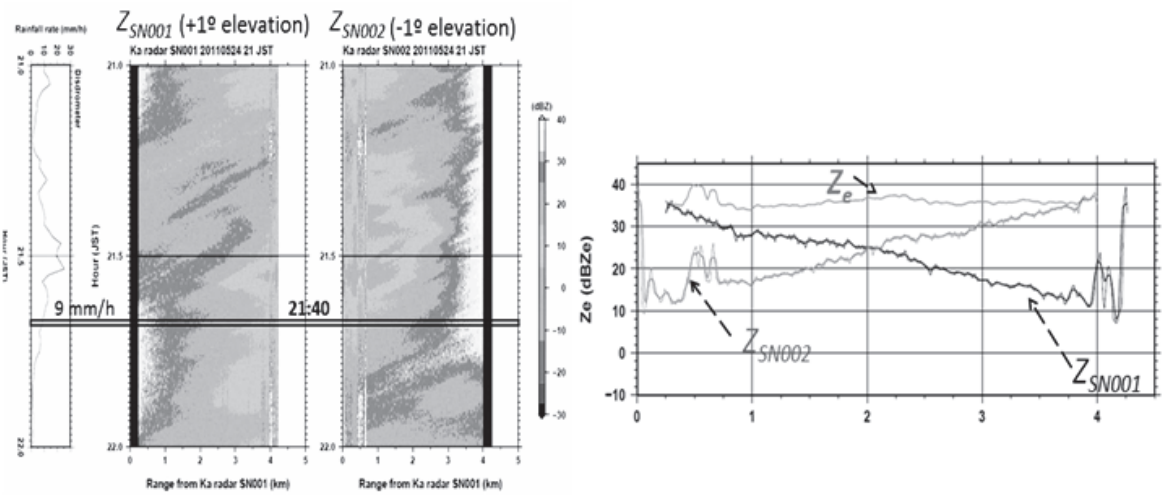

図2　NICT沖縄とOIST間の対向実験の事例。横軸は距離、縦軸は時間である。左：中間点の地上降雨強度、二つのKaレーダのエコーの変化。右:21:40の時の二つのレーダの降水エコー強度のプロファイルと推定された等価レーダ反射因子のプロファイル。

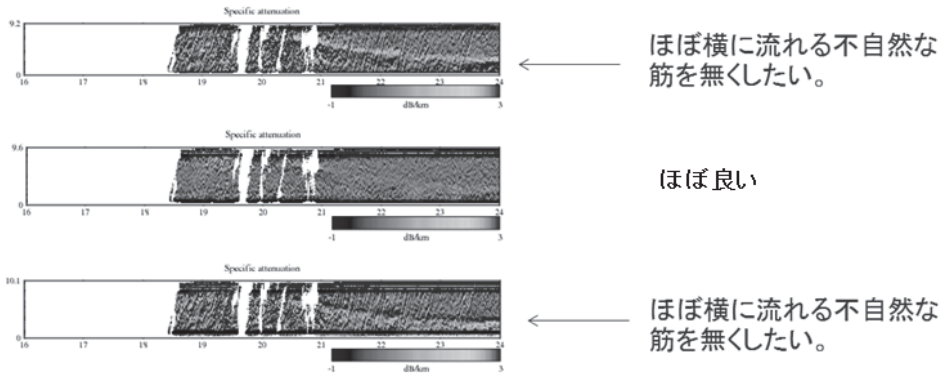

図3　降雨減衰係数の時間（横軸、8時間分）距離（縦軸、9.6km）断面のパラメータによる変化。上と下の図には不自然な線が現れているが、中の図のようにパラメータのチューニングにより消える。

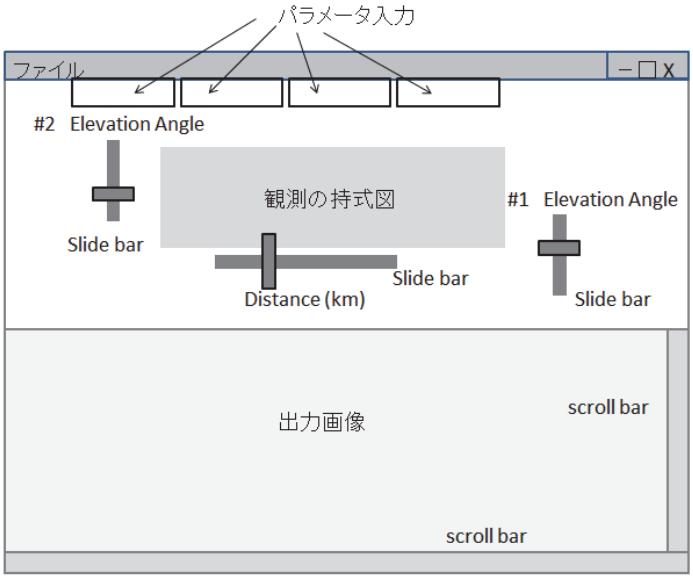

図4 制作開始時に想定したGUI の画面。パラメータの入力にはslide bar を使う。観測の模式図を説明図として持つ。上部にはファイル名とともに数値でパラメータを表示する。下側にはscroll bar 付きで処理画像を表示する。

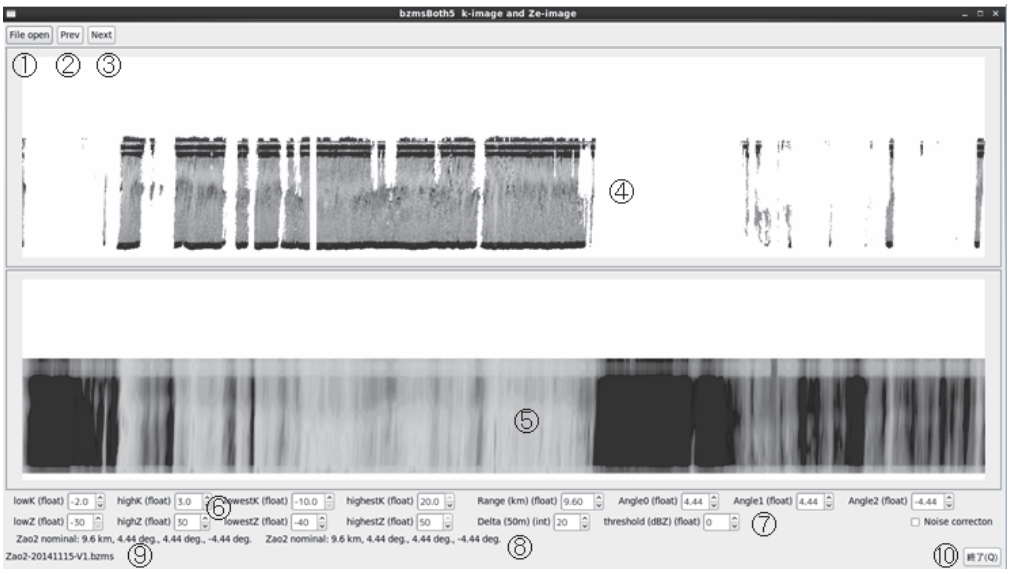

図5 2014 年11 月14 日の観測についてのGUI 画像の例。横軸は時間（0-24 時）、縦軸は距離である。①：File open ボタン、②：一つ前の降雨日の図を出力するボタン、③：一つ後の降雨日の図を出力するボタン、④：降雨減衰係数の時間距離断面、パラメータにより、パターンが変化する、⑤：等価レーダ反射因子の時間距離断面。⑥：画像のカラースケールの調節ボタン。上が降雨減衰係数の図に対して、下は等価レーダ反射因子の図に対するもの。レインボーカラーの上限と下限、また表示する範囲の上限、下限を示している。⑦：7 個の調節パラメータ用のボタン。⑧：現在表示しているデータの調節パラメータの標準値。⑨：現在表示しているデータのファイル名。⑩：終了ボタン。