中央農研 研究資料 9：1－77（2014）
Misc.Pub.NARO Agric.Res.Cent.

メッシュ農業気象データ利用マニュアル

User's manual of the Agro-Meteorological Grid Square Data, NARO/ARC

大野　宏之*

Hiroyuki Ohno*

目　次

平成25年12月 9 日受付　平成26年 2 月24日受理

* 農研機構 中央農業総合研究センター 情報利用研究領域

Ⅰ　はじめに

１　「メッシュ農業気象データ」とは

農作物の気象被害の解析や評価を迅速に実施するためには，気象データが最新の状態で準備されていることが必要です．そして，作物の成長予測や農業気象災害の早期警戒を行ううえでは，農業気象データの将来予測も必要です．そこで，（独）農業・食品産業技術総合研究機構中央農業総合研究センター情報利用研究領域では，メッシュ農業気象データシステム（The　Agro-Meteorological Grid Square Data System）を構築して，気象庁の実況データや数値予報データ，平年値データ等を基に，1980年以降の日別の農業気象データを全国についてメッシュで毎日作成し更新しています．このデータを「メッシュ農業気象データ」と呼びます．

メッシュ農業気象データシステムが作成する農業気象要素は，日平均気温，日最高気温，日最低気温，日平均相対湿度，日照時間，日射量，下向き長波放射量，日積算降水量，日平均風速（参考データ）の９種類です．ただし，1980年～2007年の期間については，日平均相対湿度，下向き長波放射量，日平均風速はありません．

メッシュ農業気象データは，北海道から西南諸島にかけて設定された６つの領域（Area１～Area６）毎に作成され，１日１回，午前６時頃に更新されます．それぞれの領域がカバーする範囲を図１に示します．

なお，メッシュ農業気象データシステムは JGD2000（新測地系）に準拠しています．

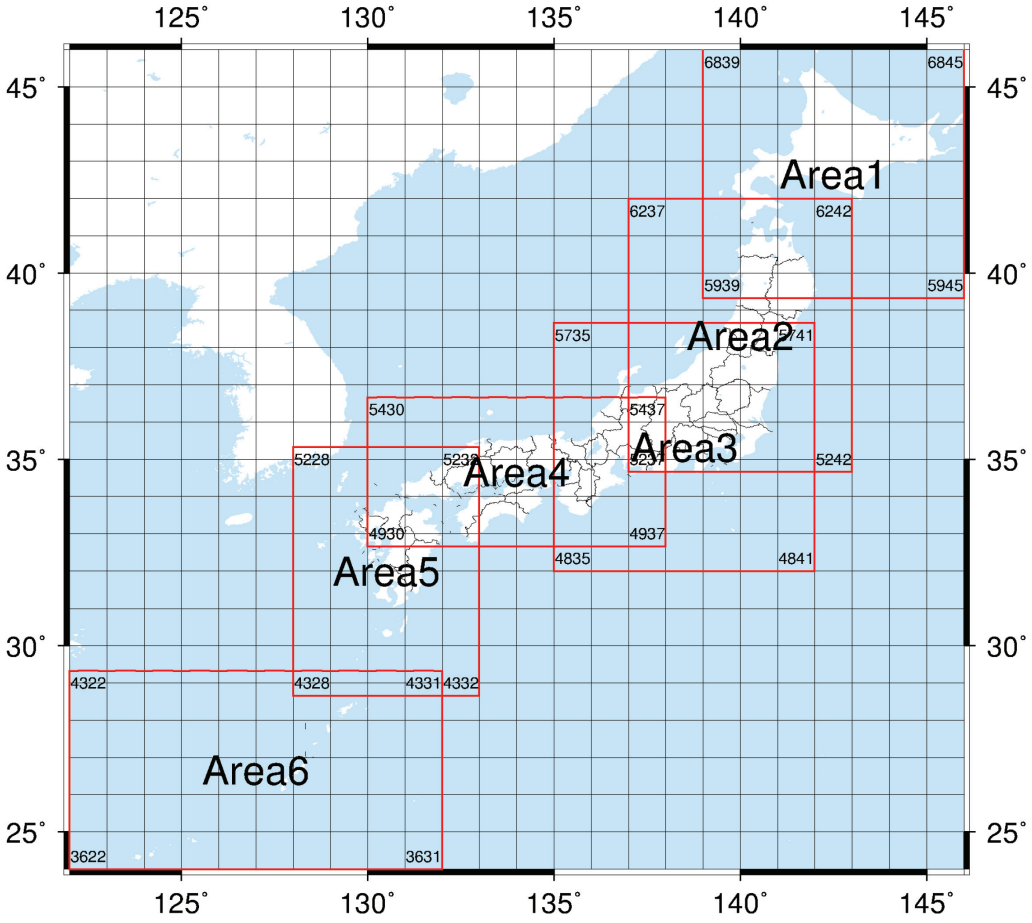

図１．メッシュ農業気象データサーバーが提供するデータの領域（赤線）

領域四隅の数字は，それぞれに位置する基準国土１次メッシュ番号を示す．

２ 「メッシュ農業気象データ」の利用について

情報利用研究領域では，「メッシュ農業気象データ」を，下に示す利用条件のもとに非営利目的の利用者に提供しています．また，利用者をメンバーとするメーリングリストと掲示板（Ｗｉｋｉ）を開設し，利用者への技術的な支援と利用者相互の情報交換の場も提供しています．
ただし，「メッシュ農業気象データ」は，農作物の管理や栽培の計画を行うために特別に推定した気象に関する農業専用の指標であって，気象庁や気象業務事業者が提供する実況・予報気象データとは性質が異なるため農業分野以外での利用はできません．

「メッシュ農業気象データ」利用条件

1．「メッシュ農業気象データ」は，気象に関する農業専用の指標なので農業分野以外での利用はできません．
2．データ提供者は，利用者が「メッシュ農業気象データ」の利用によって生じた結果，ならびに，利用できないことによって生じた結果について，一切の責任を負いません．
3．「メッシュ農業気象データ」を利用して作成した情報を販売することはできません．
4．「メッシュ農業気象データ」の利用期間は，申請受理日からその年度の３月31日までとします．ただし，くりかえし再申請することができます．
5．利用者は，利用期間の終了時に，利用状況を報告することとします．
6．中央農研が配布する「メッシュ農業気象データ」の利用に関するプログラムは “GNU General Public License” に従います．改変や再配布の際にはこれを遵守してください．

Ⅱ．メッシュ農業気象データシステム

農業気象データを栽培技術に応用するには，日々の最新データが速やかに利用者に届けられることが重要です．そこで，メッシュ農業気象データシステムでは，インターネット上にデータを配信するサーバーを設置し，表計算ソフトの利用者に対しては CSV ファイルまたは Web クエリ機能によりデータを提供し，プログラミング言語の利用者には OPeNDAP と呼ばれる通信プロトコルによりデータを提供しています．さらに，利用者向けのホームページとメーリングリストを開設し，サンプルの提供や技術相談の対応などを行っています．

１　メッシュ農業気象データ配信サーバー

メッシュ農業気象データは，インターネット上に設置された専用のサーバー（メッシュ農業気象データ配信サーバー；以降，データ配信サーバー）から全国の利用者に提供されます．データ配信サーバーのＵＲＬ（インターネットアドレス）は次の通りです．

http://mesh.dc.affrc.go.jp/opendap/

このサーバーにはアクセス制限が設定されており，登録された固定の IP アドレスからの接続だけを許可するので，メッシュ農業気象データを利用しようとする者は，予め所定の様式によりアクセスする PC の IP アドレスを申請する必要があります．

データ配信サーバーはごく簡素なホームページを持っており，上記 URL に Web ブラウザでアクセスすると，図２のようなトップページが表示されます．メッシュ農業気象データは，データ配信サーバーの中で，図３のように，領域別，年次別に整理されていて，ホームページにリス

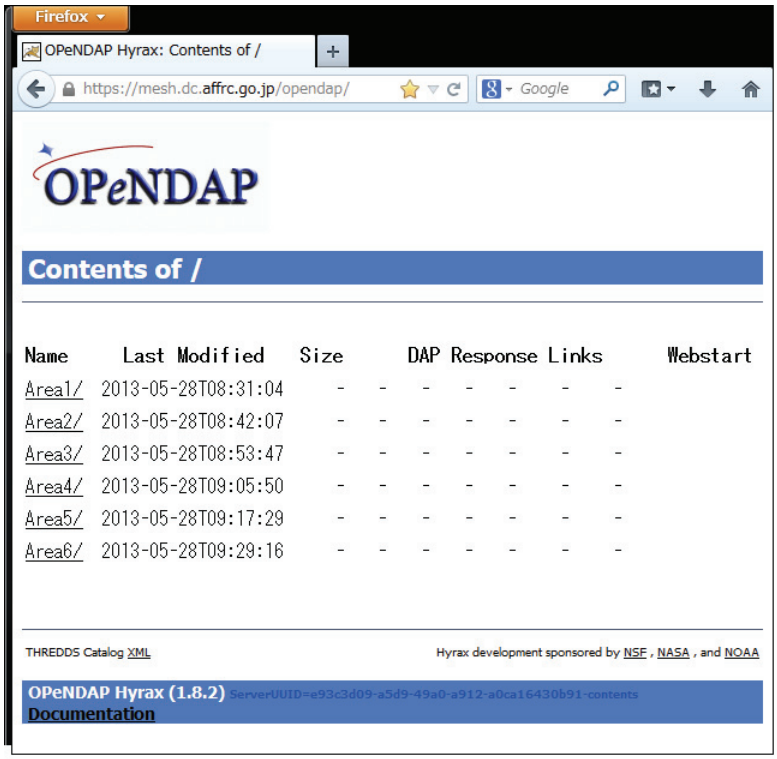

図２．メッシュ農業気象データ配信サーバーのトップページ

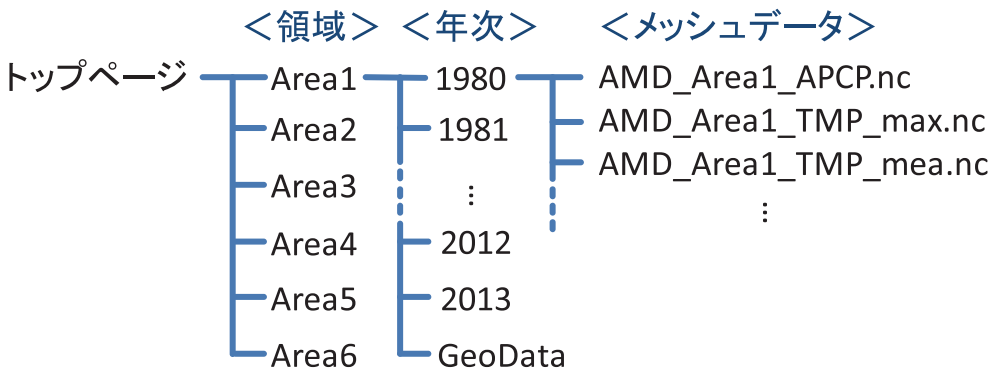

図３．データ配信サーバーにおけるメッシュ農業気象データの階層構造

データは、領域、年次、気象要素で整理されている.

トされているリンクを，領域，年次，と辿ると，「AMD_Area3_TMP_mea.nc」等のファイル名のようなリストが表示されます．これらが，当該領域，当該年，当該要素のメッシュ農業気象データセットのページへの入り口です．何の気象要素のページであるかは，“AMD_Area_”と，“.nc”との間の文字列で判断します．上の例では，“TMP_mea”であり，これは，日平均気温のデータセットであることを示しています. データへのエントリーをクリックするとフォームが開き, データのサイズや気象要素名，単位，更新日などの情報が表示されます．気象要素とそれを示す文字列との関係を表１に示します.

なお，リストには AMD_Area3_Cli_TMP_mea.nc というエントリーも表示されますが，これは平均気温の平年値データへのものです．一般に“_Cli_”は平年値データへのエントリーであることを意味します.

表１．メッシュ農業気象データシステムが提供する農業気象要素とその略号

	農業気象要素名	略　号
1	日平均気温	TMP_mea
2	日最高気温	TMP_max
3	日最低気温	TMP_min
4	日平均相対湿度	RH
5	日照時間	SSD
6	日射量	GSR
7	下向き長波放射量	DLR
8	日積算降水量	APCP
9	日平均風速	WIND

データ配信サーバーは，メッシュ農業気象データを，簡易ホームページのフォーム，GET メソッド，OPeNDAP の三通りの方法で提供します．いずれの方法においても，各データ領域（Area１～Area６）内の必要な部分だけを切り出して取得することができます．特に，OPeNDAP の方法では，複数年次にわたるデータを一つの塊として取得することができます．

２　地理情報

データ配信サーバーからは，農業気象データのほかに，メッシュの平均標高や面積，土地利用比率などの地理情報を取得することができます．地理情報を利用すると，自分の県だけの分布図の作成や，水田が分布する地域だけの平均気温の計算など，農業気象データをより高度に利用することができます．これらの地理情報は，データ配信サーバーの”年次”の階層に設けられた「GeoData」下に置かれています（図３）．データの形式を揃えてありますので，農業気象データと殆ど同じ方法で利用することができます．以下，順に地理情報について解説します．

１）平均標高

気象庁が「メッシュ平年値2010」を作成する際に使用したメッシュの平均標高データです．記号は ‘altitude’ で，単位はメートルです（図４）．

２）メッシュの面積

三次メッシュは約１km^2の広さを持ちますが，緯度・経度を基準として区切られているため北ほど面積は小さくなります．このデータは各メッシュの正確な面積を表します．記号は ‘area’ で，単位は平方メートルです（図５）．

３）土地利用比率

国土交通省の国土数値情報　土地利用細分メッシュデータの平成21年度版データから作成した，各メッシュにおける土地利用比率データです．土地利用区分毎に比率データが作成されています．単位はパーセントです（図６）．表２に，土地利用区分の説明と記号を示します．

４）都道府県範囲

国土交通省国土数値情報行政区域データ（平成24年度；世界測地系）をもとに作成した都道府

表２．土地利用比率データの記号と定義

データセット記号	種　別	定　義
landuse_H210100	田	湿田・乾田・沼田・蓮田及び田とする.
landuse_H210200	その他の農用地	麦・陸稲・野菜・草地・芝地・りんご・梨・桃・ブドウ・茶・桐・はぜ・こうぞ・しゅろ等を栽培する土地とする.
landuse_H210500	森林	多年生植物の密生している地域とする.
landuse_H210600	荒地	しの地・荒地・がけ・岩・万年雪・湿地・採鉱地等で旧土地利用データが荒地であるところとする.
landuse_H210700	建物用地	住宅地・市街地等で建物が密集しているところとする.
landuse_H210901	道路	道路などで，面的に捉えられるものとする.
landuse_H210902	鉄道	鉄道・操車場などで，面的にとらえられるものとする.
landuse_H211000	その他の用地	運動競技場，空港，競馬場・野球場・学校港湾地区・人工造成地の空地等とする.
landuse_H211100	河川地及び湖沼	人工湖・自然湖・池・養魚場等で平水時に常に水を湛えているところ及び河川・河川区域の河川敷とする.
landuse_H211400	海浜	海岸に接する砂，れき，岩の区域とする.
landuse_H211500	海水域	隠顕岩，干潟，シーパースも海に含める.
landuse_H211600	ゴルフ場	ゴルフ場のゴルフコースの集まっている部分のフェアウエイ及びラフの外側と森林の境目を境界とする.

表３．メッシュ農業気象システムが使用する都道府県の番号

番号	道（振興局）	番号	都府県	番号	都府県	番号	都府県
0100	北海道	0200	青森県	1800	福井県	3400	広島県
0101	石狩振興局	0300	岩手県	1900	山梨県	3500	山口県
0102	渡島総合振興局	0400	宮城県	2000	長野県	3600	徳島県
0103	檜山振興局	0500	秋田県	2100	岐阜県	3700	香川県
0104	後志総合振興局	0600	山形県	2200	静岡県	3800	愛媛県
0105	空知総合振興局	0700	福島県	2300	愛知県	3900	高知県
0106	上川総合振興局	0800	茨城県	2400	三重県	4000	福岡県
0107	留萌振興局	0900	栃木県	2500	滋賀県	4100	佐賀県
0108	宗谷総合振興局	1000	群馬県	2600	京都府	4200	長崎県
0109	オホーツク総合振興局	1100	埼玉県	2700	大阪府	4300	熊本県
0110	胆振総合振興局	1200	千葉県	2800	兵庫県	4400	大分県
0111	日高振興局	1300	東京都	2900	奈良県	4500	宮崎県
0112	十勝総合振興局	1400	神奈川県	3000	和歌山県	4600	鹿児島県
0113	釧路総合振興局	1500	新潟県	3100	鳥取県	4700	沖縄県
0114	根室振興局	1600	富山県	3200	島根県		
		1700	石川県	3300	岡山県		

県の範囲を示すデータです. 北海道については, 振興局毎に区分されています. 都道府県範囲データは２種類の形式で用意されています.

（1）全国一括都道府県範囲図

都道（振興局）府県に割り振られた番号を各メッシュに割り付けたもので，都道府県で色分けした日本地図のような形式です．複数の都府県に所属するメッシュには，メッシュの中心点が所属する都道府県の番号が割り付けられています．割り付けられた数と都道府県との対応は表３の

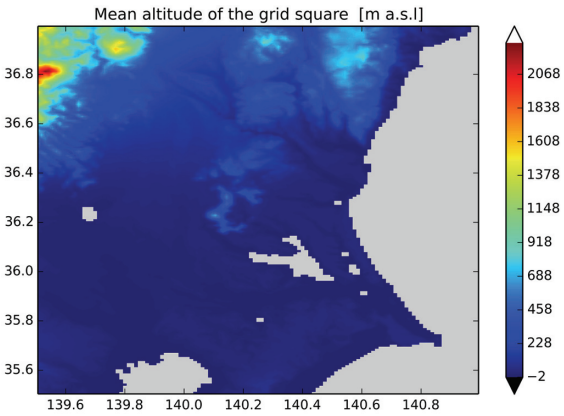

図４．平均標高データの例

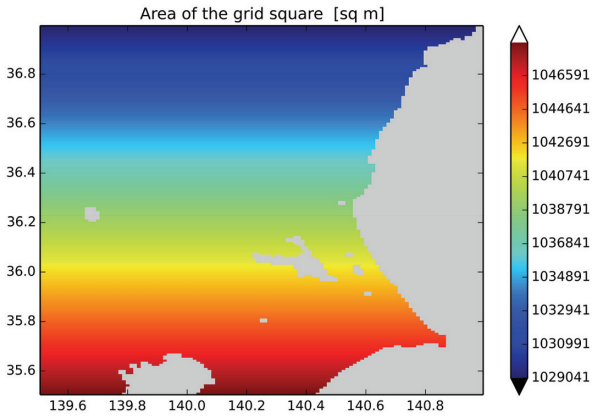

図５．メッシュ面積のデータの例

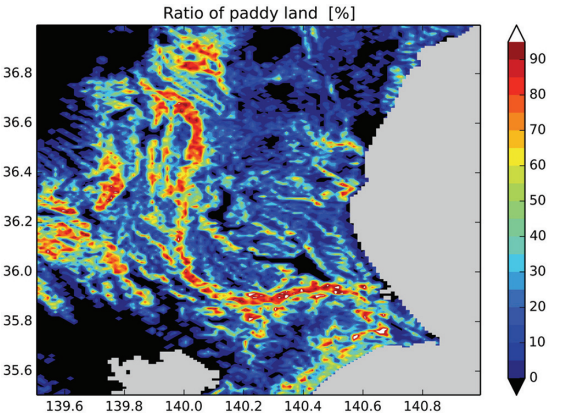

図６．土地利用比率データの例（水田比率）

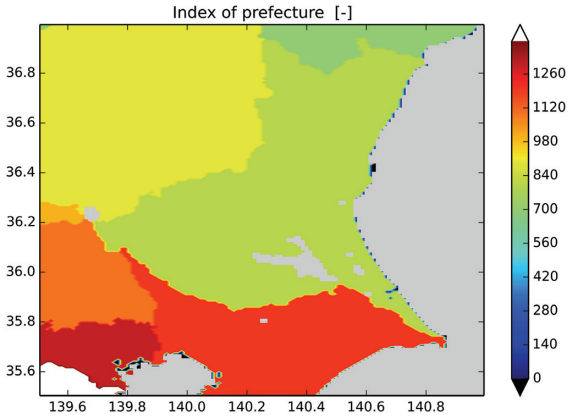

図７．全国一括都道府県範囲データの例

とおりで，記号は‘pref_all60’です（図７）．

（2）都道府県別範囲図

もう一つの形式は，都道府県毎に作成され，その都道府県に含まれるメッシュに値１，他のメッシュに無効値を与えたものです．一部でも当該都道府県に含まれていれば，そのメッシュには１が与えられています．記号は‘prefec_nnnn’で，nnnn の所には表３の４桁の数字が入ります．

例として，茨城県に含まれるメッシュだけが１，他のメッシュには無効値が入っている地理情報のデータ記号は，‘pref_0800’です（図８）．

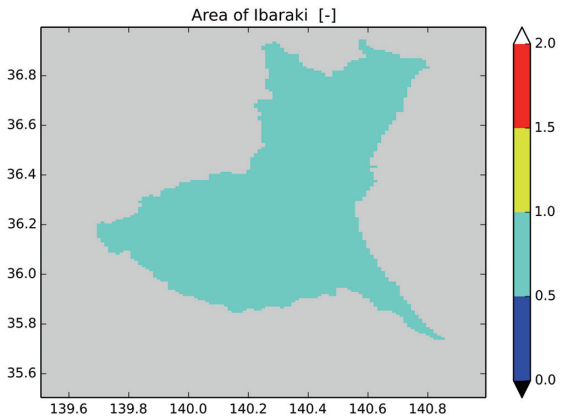

図８．都道府県別範囲データの例

３　データ処理ソフトウエア

メッシュ農業気象データシステムは，利用者がデータを処理するソフトウエアとして，Microsoft 社の表計算ソフト Excel とプログラミング言語の利用を想定しています．Excel については，第Ⅲ章において利用の具体的な方法を説明します．

メッシュ農業気象データシステムでは，これからプログラミング言語に取り組む利用者にPython（パイソン）を薦めています．第Ⅳ章において，これを利用してデータにアクセスする方法とその処理の方法を解説します．また，Ⅶ-1にPythonのインストール手順を掲載しています．

メッシュ農業気象データをPython以外のプログラミング言語で処理する利用者は，その言語におけるOPeNDAPライブラリ，および，NetCDFライブラリを入手し，実行時にインポートしてください．AMD_Toolsモジュールの GetData 関数のスクリプトに，データ配信サーバーとの具体的な通信手順が記載されていますので，それを参考に，それぞれの言語の読み込みモジュールを作成してください．

Pythonには美しい図を作成する機能がありますが，季節変化をアニメーションで表示したり，分布図上の任意の地点における季節変化を即座に表示する等のインタラクティブな可視化をすることはできません．Pythonでの作成した処理結果をインタラクティブに可視化するソフトウエアとして，メッシュ農業気象データシステムではIDVを推奨しており，第Ⅴ章においてその利用方法を解説します．また，Ⅶ-2でIDVのインストール手順を掲載しています．

４　メーリングリストとWiki

メッシュ農業気象データシステムでは，登録利用者がメンバーとなるメーリングリストを開設しています．ここでは，メッシュ農業気象データに関する様々な話題を利用者やメッシュ農業気象データ開発チームで議論するほか，技術的な質問も受け付けています．メーリングリストのアドレスは次の通りです．

MeshUser@ml.affrc.go.jp

メッシュ農業気象データシステムでは，Wikiと呼ばれる掲示板のようなホームページも開設しています（図９）．ここでは，システムの運用状況の連絡や，新しいデータセットの紹介，データ処理のためのサンプルプログラムの提供などを行っています．ホームページのURLは，次の通りです．

https://ml-wiki.sys.affrc.go.jp/MeshUser/

図９．メッシュ農業気象データ利用者用ホームページ

サンプルプログラム等をここからダウンロードすることができる．

Ⅲ　Microsoft Excel によるデータ処理

Microsoft 社の表計算ソフト Excel は，農業の試験研究普及機関でも広く使われているソフトウエアです．メッシュ農業気象データ配信サーバーは，CSV ファイルのダウンロードと，Web クエリによるデータ転送で Excel にデータを提供します．

1　CSV ファイルのダウンロード

1）簡易ホームページのフォームからのリクエスト

データ配信サーバーが持つ簡易ホームページのフォームから，メッシュ農業気象データを CSV ファイルまたは NetCDF ファイルとして取得することができます．以下に，北緯34.5～36.0度，東経139.0～140.5度（東京湾の周辺）の平均気温を2013年１月１日～10日について CSV ファイルで取り出す方法を例示します．フォームを表示させる方法は，「Ⅱ-1．メッシュ農業気象データ配信サーバー」を参照してください．東京周辺は Area３に含まれます．また，日平均気温の記号は TMP_mea です．

図10に，Area３における2013年の日平均気温データのフォームを示します．フォーム中程に表示される見出し Variables の右側の TMP_mea の左脇に小さなチェックボックスがあるので，これをチェックすると，空欄だったテキストボックスに，0:1:364，0:1:799，0:1:559という数字が表示されます．Area３は，南北方向に800メッシュ，東西方向に560メッシュ，日付方向に365層のデータの集合であり，この数字は，Area３から取得できるデータの最大範囲を示しています（中央の数字１は気にしないでください）．最初の「0:1:364」は日の範囲です．０は１月１日，364は12月31日を表します．次の「0:1:799」は緯度方向の範囲です．０は Area３の南端に並ぶメッシュ，799は北端に並ぶメッシュを示します．三番目の「0:1:559」は経度方向の範囲です．０は Area３の西端に並ぶメッシュ，559は東端に並ぶメッシュを示します．ここの数字を，データを取得する緯度経度／期間に対応する番号に書き直します．

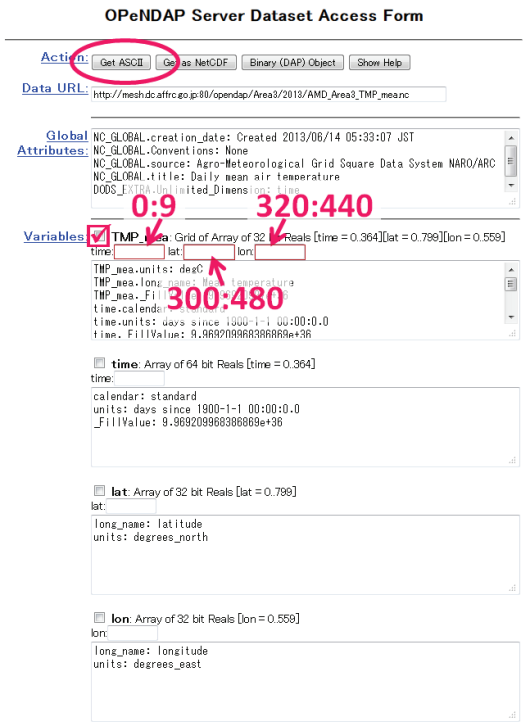

図10．データの概要表示画面

Area３地域における2013年の日平均気温の例

緯度や経度とメッシュの番号との対応は，Excel ファイル「AMGSD の領域.xls」のワークシートで調べます．このファイルは，利用者用 Wiki から入手することができます．セルB39：B40に知りたい緯度と経度を十進数表記で入力すると，対応する緯度方向のメッシュ番号（lat），経度方向のメッシュ番号（lon）の番号が計算されます（図11）．これから，北緯34.5～36.0度，東経139.0～140.5度の Area３における配列要素の範囲が，lat については300～480，lon については320～440と分かります．また，B50に日付を入力すると，time の配列要素が計算されます．これをもとに，テキストボックスに，順に0:9，300:480，320:440と記入します．正式な文法では，それぞれ，0:1:9，300:1:480，320:1:440ですが，これでも構いません．

次に，フォーム（図10）の一番上に並んでいる４つのボタンの中から［Get ASCII］を押しま

図11. ワークシート「AMGSD の領域.xls」

このワークシートを用いて緯度／経度とデータセットにおける要素番号との対応を調べる.

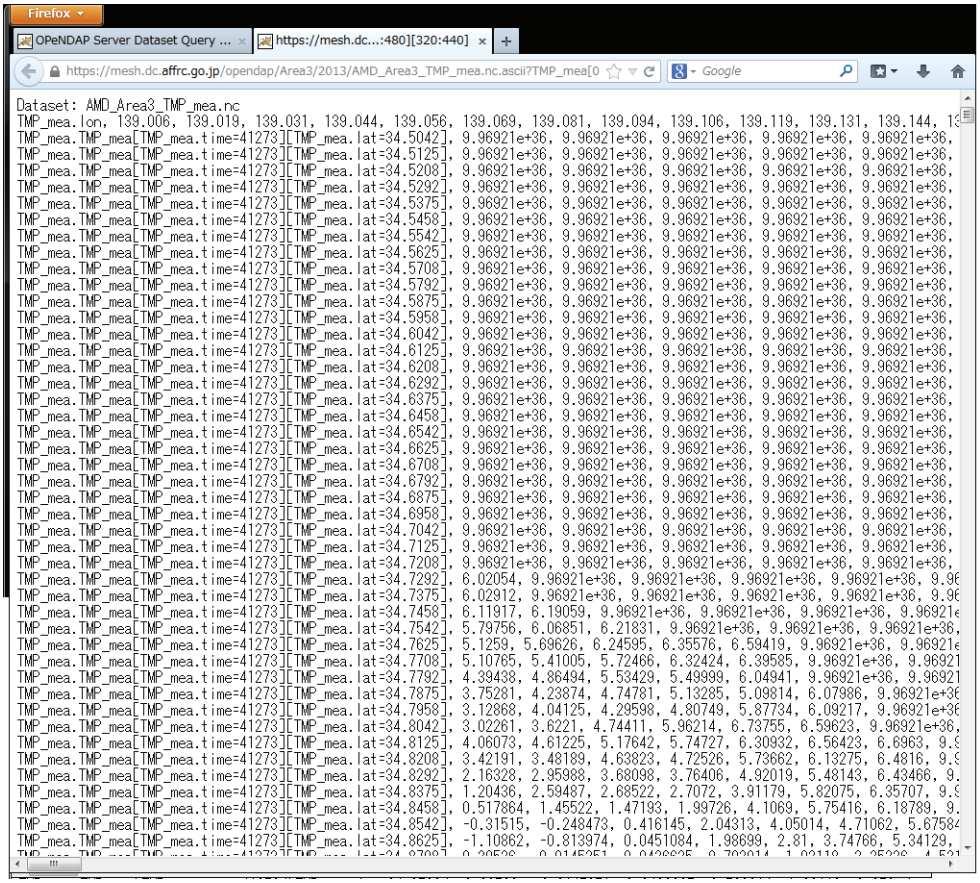

図12. データが取得されたブラウザの画面

北緯34.5～36.0度，東経139.0～140.5度における2013年１月１日～10日の日平均気温データが表示されている.

す．すると，新しいページが開いて，図12のような画面が表示されます．データをダウンロードしようと思ったのに表示されてしまいました．これは，ブラウザの設定のためです．ファイルとして保存するには，このページ上で右クリックし，「名前を付けてページを保存」を選びます.

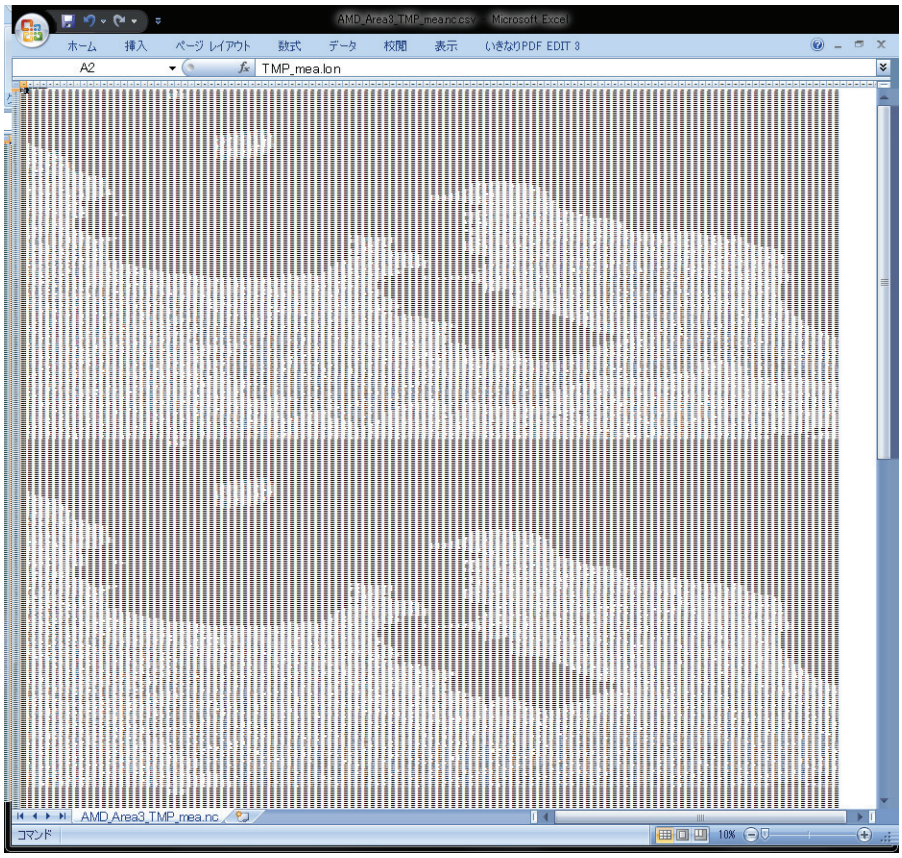

図13. 取得したデータを MS−Excel に読み込んで表示した画面

ブラウザによっては，メニュー「ファイル」から「名前付けて保存」とします．保存の際，ファイル名の拡張子を変更して AMD_Area3_TMP_mea.nc.csv として保存してください．

このファイルをエクセルから開き，ウインドウ右下の表示倍率スライドバーを左いっぱいに動かして縮小表示すると南北が逆転した房総半島〜伊豆大島が縦に10枚繋がっている様子を確認することができます（図13）．

なお，この「地図」で，海に相当するセルには “9. 96921 E + 36” が代入されます．メッシュ農業気象データでは，海や湖沼など，未定義であることを実数で示す場合にこの値を使用しています．

2）Get メソッドによるリクエスト

Get メソッドとは，インターネットでホームページの情報を通信する手順の名称です．Web ブラウザの URL 入力ボックスに次の文字列を入力すると，ブラウザはデータ配信サーバーに Get メソッドでデータをリクエストし，上の例と全く同じデータを取得します．

http://mesh.dc.affrc.go.jp/opendap/Area3/2013/AMD_Area3_TMP_mea.nc.ascii?TMP_mea[0:9][300:480][320:440]

この URL は以下のように構成されています．まず，「http://」に続く「mesh.dc.affrc.go.jp」はメッシュ農業気象データサーバーのインターネット上での名前（ホスト名）で「/opendap/」はデータセットのルートディレクトリです．そして，それに続く「Area3/2013/」が領域と年，

「AMD_Area3_TMP_mea.nc」がデータのエントリーで，その次の「.ascii」はテキスト（CSV）フォーマットでのデータ送信要求であることを示します．続けて？マークを書き，その後ろにリクエストするデータ名とその範囲を指定します．

２　Web クエリを利用した特定メッシュのデータ取得

Get メソッドを利用すると，Web クエリを設定した Excel ワークシートに最新のデータを簡単に取り込むことができます．Web クエリとは，インターネット上にあるホームページ内の（HTML で記述された）数表から数値を取り出してワークシートにとりこむ Microsoft Excel の機能です．

ここでは，埼玉県に位置する熊谷地方気象台（北緯36. 15度，東経139. 38度）における2013年の最高気温を Web クエリでワークシートに取り込んで，折れ線グラフとして表示させる例を示します．

まず，緯度経度と，データセットのメッシュ番号との対応を知るワークシート AMGSD の領域．xls を利用して，熊谷地方気象台が属するメッシュの番号を求めます．緯度経度を入力すると，この地点が Area３と Area４の領域に含まれ，Area３では，lat が498，lon が350とわかります．次に，「Ⅲ-1-2）Get メソッドによるリクエスト」に説明されている文法に従って，このメッシュにおける2013年（１年分）の最高気温データを取得するＵＲＬを作ります．それは以下のとおりです．

http://mesh.dc.affrc.go.jp/opendap/Area3/2013/AMD_Area3_TMP_max.nc.ascii?TMP_max[0:364][498][350]

ここで，［0:364］は１年分の期間（365日）で，［498］と［350］は，それぞれメッシュの緯度番号と経度番号です．

以上の準備が終わったら，Excel の操作に移ります．メニュー「データ」から，「Web クエリ」を選択し，ポップアップする「新しいクエリ」というウインドウの上部にある「アドレス（D）：」に，この URL を入力し［移動］ボタンを押します．するとこのウインドウにデータが表示される（図14）ので，右下の［取り込み（Ｉ）］ボタンを押します．データを貼り込む場所を指定するダイヤログボックスがポップアップするので，デフォルトの「既存のワークシート」「=A1」で［OK］してしばらく待つと，ワークシートにデータが取り込まれます（図15）．以上の作業で，このワークシートに Web クエリが定義されました．次回以降，このファイルを開いて，メニュー「データ」から，「全て更新」ボタンをクリックするだけで，データは最新のものに置き換えられます．

さて，Excel の Web クエリは，HTML の表をセルに変換しますが CSV をセルには変換しません．このため，図15のように，データは一切合切Ａ列に張り付いてしまいます．このままでは解析に利用できないので，Ｂ列～Ｅ列に式を書いてこの文字列を解析し，日付とデータを取り出してみます．Ｂ列には次の式を書きます．これは，日付のデータがＡ列の長い文字列中のそれぞれ何文字目から書かれているかを調べる式です．

＝FIND（“time=”, A3）　　←*セルＢ３に入力してから列全体にコピーしてください．*

同様に，気温データが何文字目から記されているかを調べる式をＣ列に入力します．

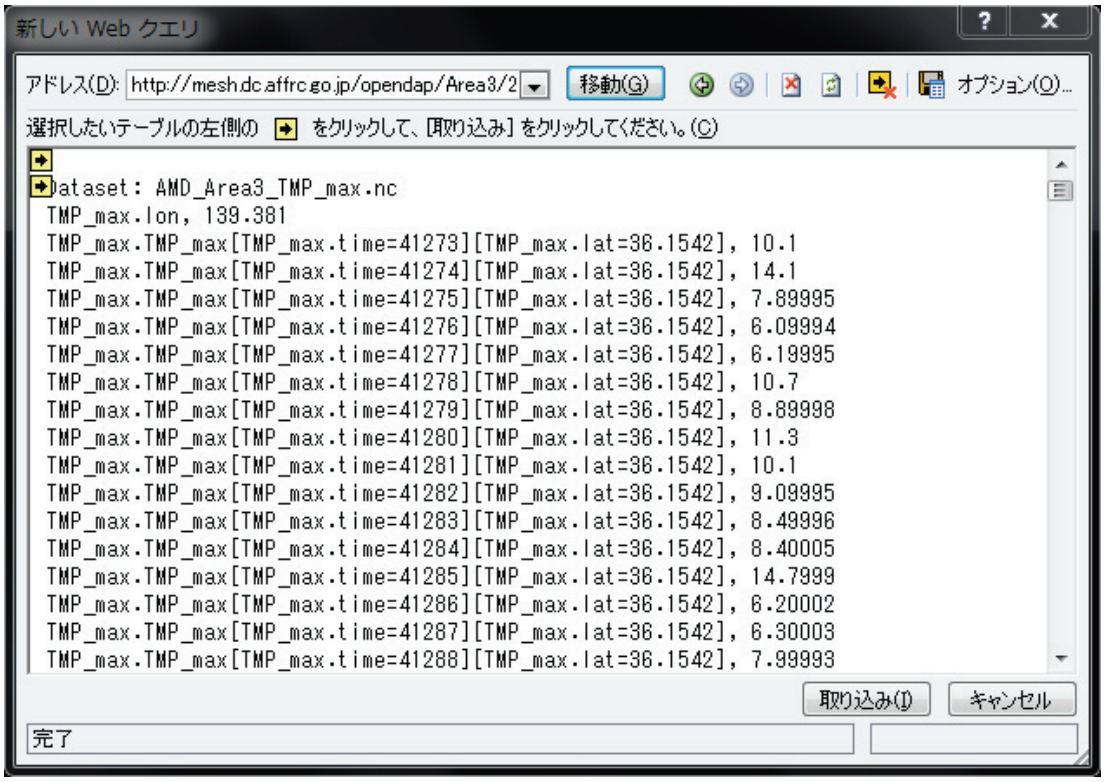

図14. 新しい Web クエリのウインドウ

熊谷の最高気温を取得する URL を指定し、［移動］ボタンをクリックした直後の様子.

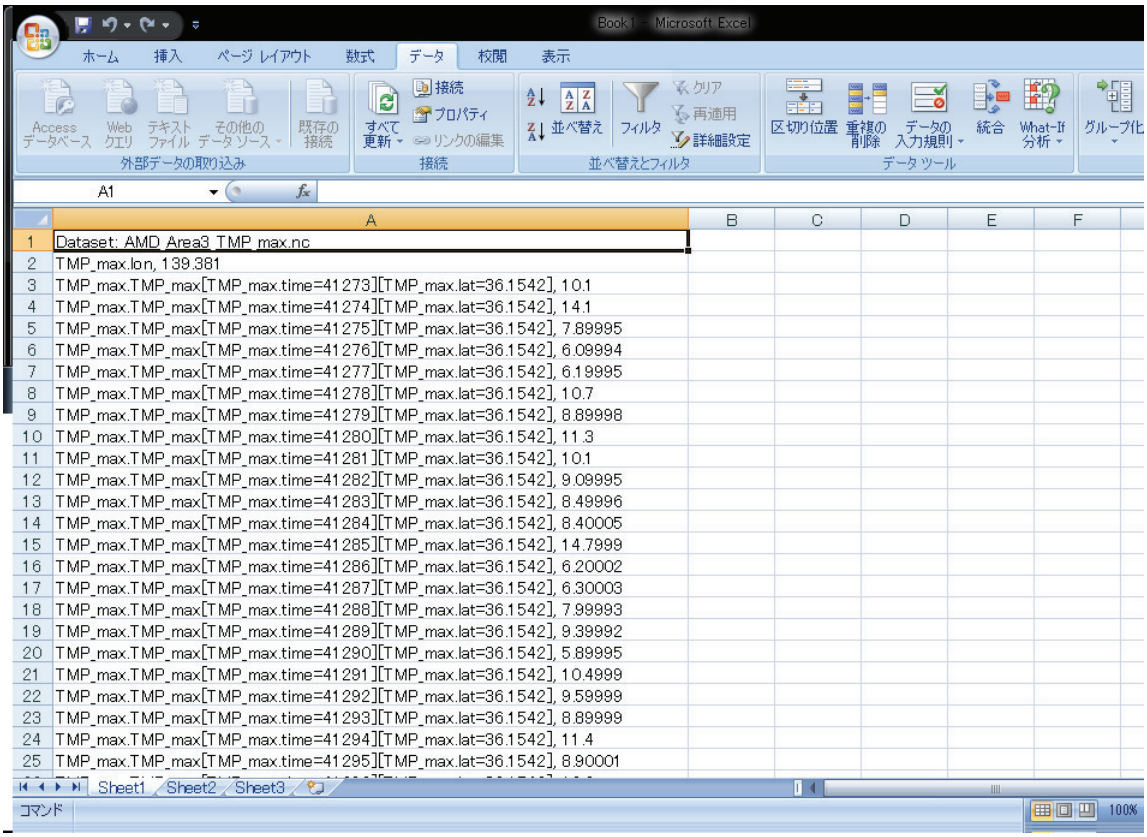

図15. Web クエリにより熊谷の日最高気温が取り込まれたところ

＝FIND（“],” , A3）　　←*セルＣ３に入力してから列全体にコピーしてください.*

日付と最高気温が書かれている場所が明らかになったところで，それを利用して文字列からデータを切り出します．Ｄ列とＥ列にそれぞれ下の式を入力します.

＝MID（A3, B3+5, 5）+2　　←*セルＤ３に入力してから列全体にコピーしてください.*

また，日付連番が返るので，このセルの表示書式を日付に設定してください

＝VALUE（MID（A3, C3+3, 10））←*セルＥ３に入力してから列全体にコピーしてください.*

これらの式を書き込むとＤ列にデータの日付，Ｅ列に日最高気温のデータが取り出されます.

このようにして取り出された日付と気温を折れ線グラフで表示するExcelファイル「WebQuery2013.xls」（図16）をwikiに掲載していますので，必要に応じダウンロードして参照してください.

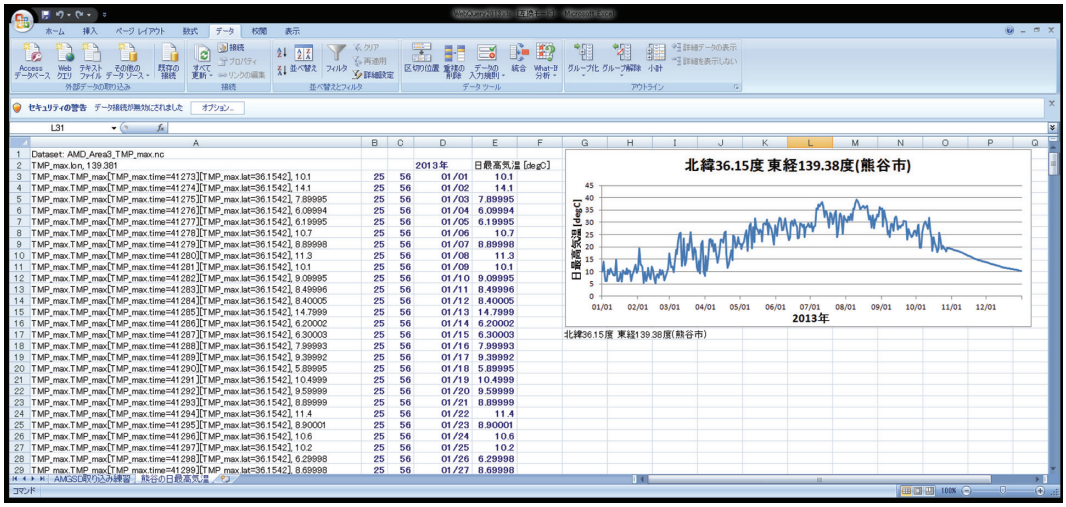

図16. 熊谷の日最高気温を取得するURLを指定し，四角ボタンをクリックしてクエリを指定したところ

３　Web クエリと VBA マクロを利用したデータの取得

先に行った作業は，Excel の VBA マクロ機能を利用すると全部自動化することができます．VBA マクロとは，Visual Basic for Applications と呼ばれるプログラミング言語を利用して Excel の様々な操作を自動化する機能です．VBA プログラムは Excel ファイルに書き込まれているので，利用者はファイルを開くだけで自動化された操作をすることができます.

Web クエリと並んでとても便利な機能ですが，反面，利用者が気づかぬままにいろいろな操作が実行されるので，大変危険な機能でもあります．そのため，通常はファイルを開いた際に「セキュリティの警告」が表示され，これらの機能が使用できない状態にされます．Excel2007の場合，図17のような表示です．これらを有効にするには，［オプション］をクリックし，表示されたウインドウで，「マクロ」と「データ接続」両方について，「このコンテンツを有効にする」を選択し，［ＯＫ］をクリックします（図18）.

メッシュ農業気象データシステムでは，以下順に説明する２種類の Excel ファイルを Wiki か

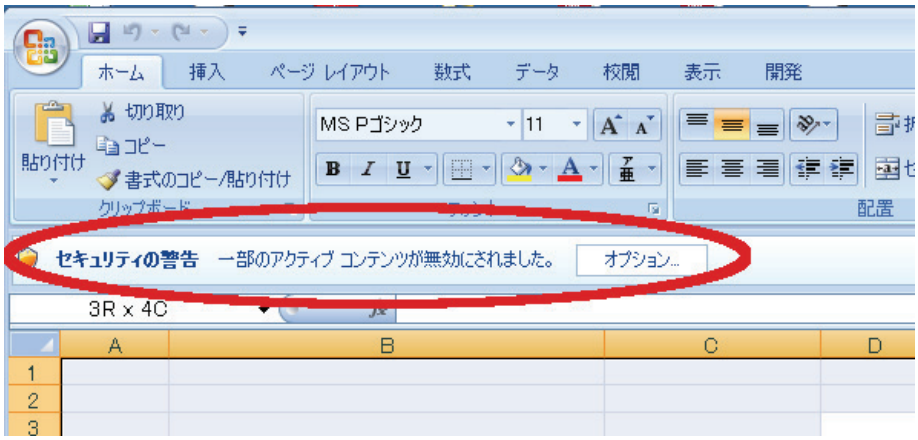

図17. Excel に表示される警告リボン

VBA マクロ等が含まれているファイルを開くときに表示される.

図18. 警告リボンの［オプション...］ボタンを押すと開く確認ウインドウ

両方のボタンを「このコンテンツを有効にする（E）」に変更する.

ら提供しています.

1）「メッシュ農業気象データ_ポイント単要素抽出_ver1.0.xlsm」

このワークシートを使用すると，特定メッシュにおける気象要素の日別値と平年値を簡単に取得することができます．マクロとデータ接続を有効にしてファイルを開くと，図19のようなワークシートが表示されます．シート上方に着色されたセルがあり，ここで取得するデータの気象要素（「データ要素」），「データ取得年」，「地点の北緯」，「地点の東経」を設定します．「データ要素」は，プルダウンメニューになっているので一覧の中から選択します．また，緯度と経度を入力すると，その位置がシート左側の地図上に菱形で表示され指定メッシュの大まかな位置が確認

できるようになっています．

指定が終了したら，［データ取得］ボタンををクリックします．しばらくするとグラフが表示され，その横に日別値と平年値が表示されます．年次や気象要素によっては，平年値が利用できないことがあります．その場合は，平年値のセルは黒色に着色されます．

2）「メッシュ農業気象データ_ポイント全要素抽出_ver1.0.xlsm」

このワークシートを利用すると，一度の操作で選択したメッシュの気象要素を全て取得し表示することが可能です（図20）．使用方法は，先のファイルとほぼ同様です．このファイルでは，データ配信サーバーデータとの通信を最多で19回繰り返すので，先のファイルでの取得よりも長い時間が必要となります．シートの動作が終わるまで，しばらく待ってください．

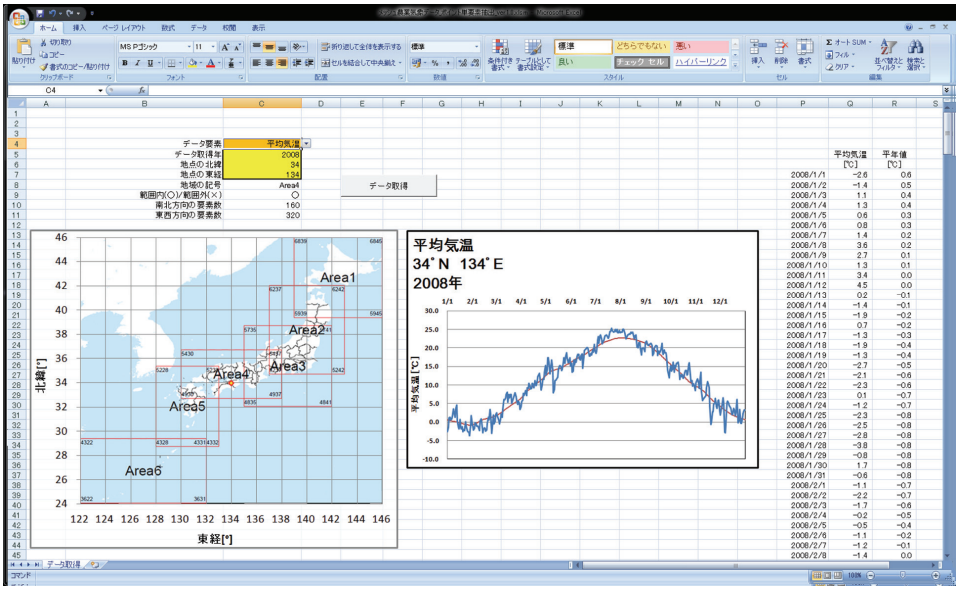

図19．ワークシート「メッシュ農業気象データ_ポイント単要素抽出_ver1.0.xlsm」

特定メッシュの特定の気象要素のデータを１年分を取得することができる．

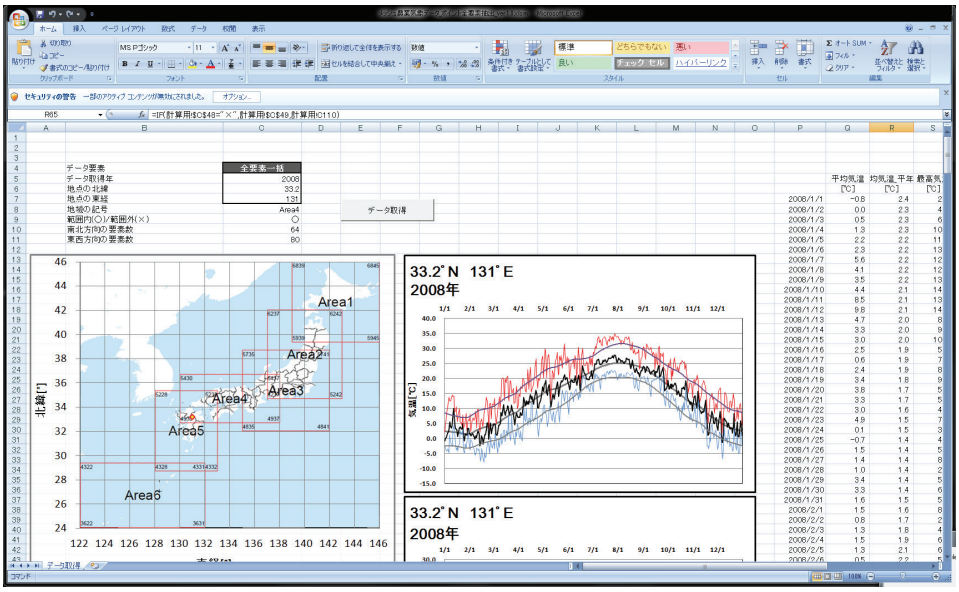

図20．ワークシート「メッシュ農業気象データ_ポイント全要素抽出_ver1.0.xlsm」

特定メッシュの全ての気象要素を一度に１年分取得することができる．

Ⅳ　Python によるデータ処理

プログラミング言語を用いると，平均値や最大値などの統計計算や特定の値を持つメッシュの抜き出しや，何らかの積算値が閾値を超えた日の計測など，データを自在に処理することができるうえ，分布図や折れ線グラフの作成までもおこなえます．そして，同じ処理を毎日更新されるデータに対して自動的に繰り返しすることも簡単にできます．

この章では，プログラミング言語の一つである Python で作成されたいくつかのプログラムを例に，データ配信サーバーからメッシュ農業気象データを取得する方法や処理の方法を解説します．

経験のない者にとってプログラミングはたいへん敷居が高いものですが，Python は初心者にも比較的わかりやすい言語なので，この機会にぜひチャレンジしてみてください．Ⅶ-1には，Python をインストールして利用できるようにするまでの手順が掲載されています．

1　Python プログラムの編集と実行

1）IPython の起動

Python プログラムは，テキストエディターで編集をして IPython コンソールで実行します．Ⅶ-1に従って作成したフォルダ PythonWorks をファイルエクスプローラーで開き，その中のショートカット「それゆけ Python！」をダブルクリックします．すると，図21のような白いウインドウが表示されます．これを IPython コンソールと呼びます．このとき，黒いウインドウが同時に開きますが，これは気にしなくて結構です．邪魔と感じる場合は，タイトルバーの右にある最小化ボタンを押して目立たなくしてください．

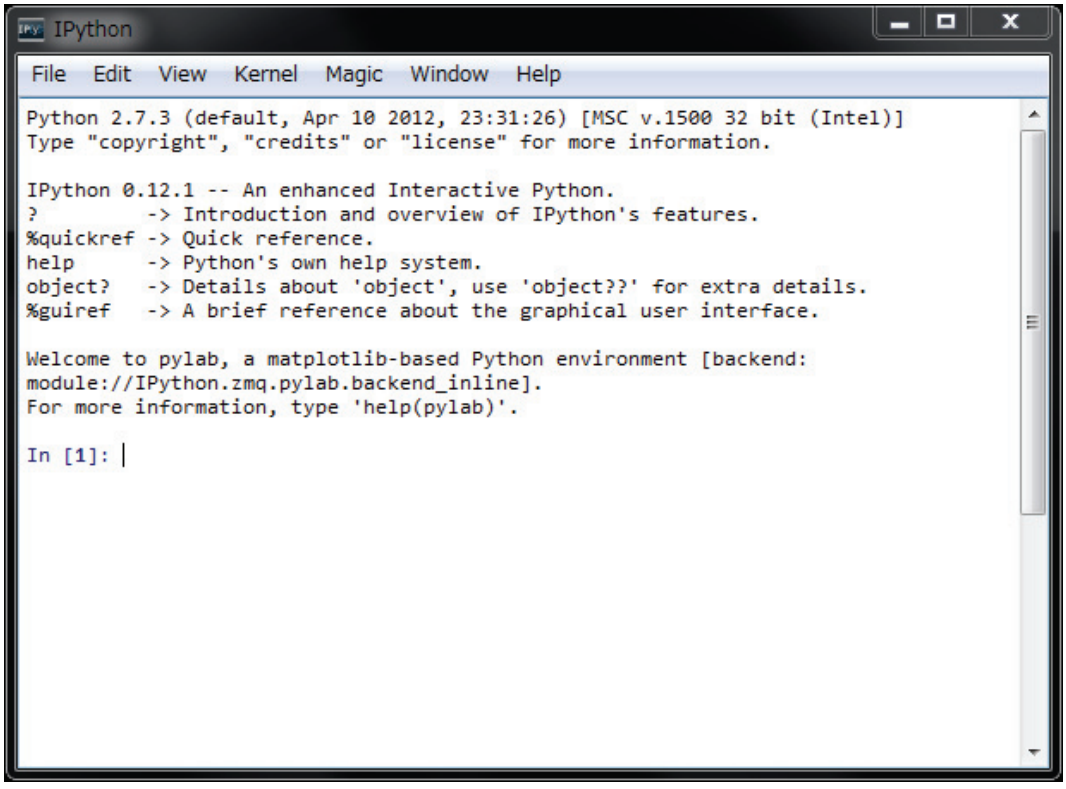

図21．IPython コンソールのスナップショット

Python プログラムをここで実行する．

２）プログラムの実行

IPython コンソールには，「In [1]:」と表示された行があります．この行をプロンプトと呼びます．ここに，「run　プログラムファイル名」と入力してエンターキーを押すと Python プログラムが実行されます（図22）.

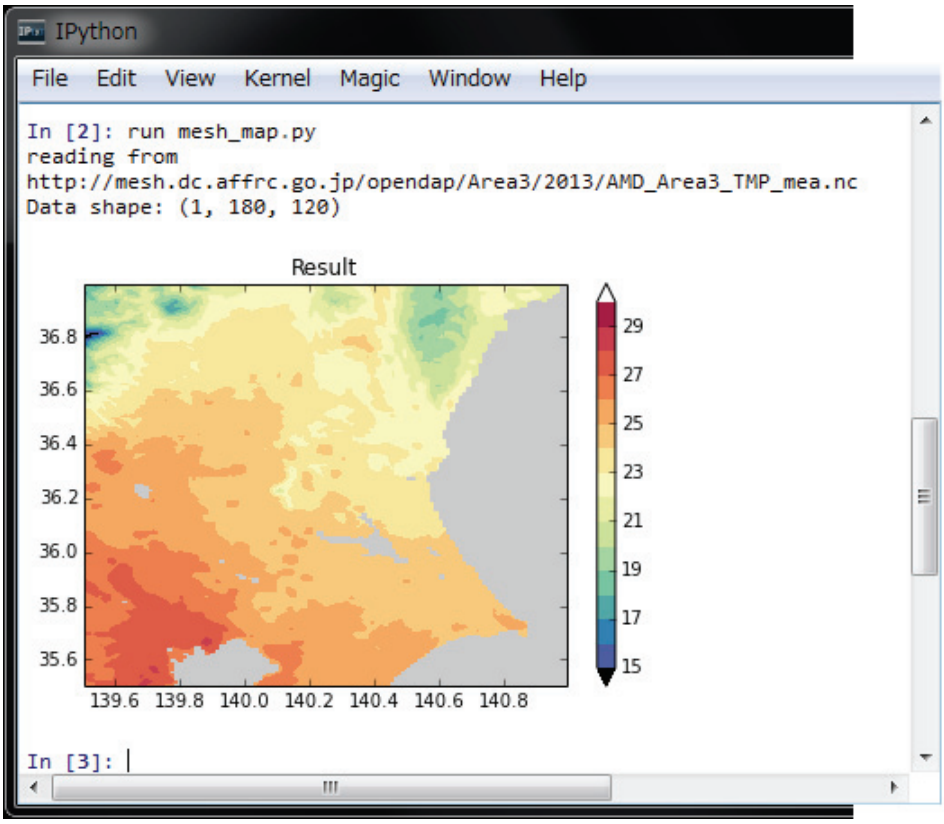

図22.　mesh_map.py の実行結果

BOX 1　メッシュデータは積み上げられた段ボール箱の中

Python プログラムのなかで，メッシュ農業気象データは配列変数に格納されますが，配列変数にデータが格納されている様子は整然と積み上げられた段ボール箱の中の品物に似ています．ここで，品物に相当するのは，特定のメッシュにおける特定の日の特定の気象要素の値　（気温で言えば「23. 0℃」など）です．データを処理する際にはこのような配列変数が複数必要になるので，Ta や Tmax などそれらに名前を付けて識別します．Python は変数名で大文字と小文字を識別します.

メッシュ農業気象データシステムでは，配列変数に個々のデータを格納する順序を決めています．段ボール箱の例えを用いると，南のメッシュのデータが入っている段ボール箱ほど手前に，西のメッシュのデータの箱ほど左に，日付が新しいデータの箱ほど上に積むことに決めています．そして，この山のなかの特定の箱を指し示すときには，必ず，「下から」「手前から」「左から」の順にその箱の場所を数えることにしています．話を配列変数に戻すと，期間初日から i 日目における，南から j 番目，西から k 番目のメッシュのデータは，配列変数の名前を Ta として Ta[i,j,k]と表記されます．i や j のことを添え字「そえじ」と呼びます.

Python では，添え字に整数のほかコロン「：」を使用することができ，「全部」を意味します．したがって，南から j 番目，西から k 番目のメッシュにおける期間全部のデータ（よって普通は複数のデータの集合）を Ta から取り出したいときは Ta[:,i,j]と指定します．なお，Python では配列の順番を０から数える決まりになっています.

さて，メッシュデータのグラフや分布図を描くときには，i や j や k が，実際何月何日で，北緯何度で，東経何度に相当するのかを知っておく必要があります．これらの情報は，GetData（）関数から戻される３つの配列変数 tim，lat，lon に格納されます．つまり，データ Ta[i,j,k]の日付，緯度，経度はそれぞれ，

tim[i]，lat[j]，lon[k]に格納されています．段ボール箱の例えでは，Ta のほかに，tim，lat，lon という一列に並んだ段ボールの塊が３つあって，それぞれの箱の列の i，j，k 番目の中に Ta[i,j,k]に対する日付，緯度，経度が書かれた紙が納められています．

逆に，日付，緯度，経度を決めてデータを取り出したいときは，箱列 tim，lat，lon の中の日付や緯度経度をそれぞれ順に照合して一致や最も近い値を見つけ，それらが入っていた箱の番号（i や j や k）を明らかにしてから，求める箱を取り出します．段ボール箱の山を相手にこの作業をするのは想像するだけでうんざりしますが，コンピュータはこのような作業が少しも苦でないので心配ありません．

３）プログラムの編集

Ⅶ-1に従って構築された環境では，プログラムファイルをダブルクリックすると，専用のテキストエディターが開きプログラムを編集することができます．それでは，2013年８月１日における茨城県周辺の日平均気温の分布図を作成するプログラム「mesh_map.py」（BOX２）を例に編集をします．このプログラムは，利用者 Wiki から入手できます．

BOX２　Python プログラム「mesh_map.py」

```
1  #!/usr/bin/env python
2  # −*− coding: utf−8 −*−
3  #
4  import numpy as np
5  import matplotlib.pylab as plt
6  import AMD_Tools as AMD
7
8  # 領域，期間，気象要素の指定.
9  element = 'TMP_mea'
10 timedomain = [ '2013−08−01', '2013−08−01' ]
11 lalodomain = [ 35.5, 37.0, 139.5, 141.0]
12 area = 'Area3'
13
14 # メッシュ農業気象データの読み込み.
15 temp, tim, lat, lon = AMD.GetData(element, area, timedomain, lalodomain)
16
17 # データの整形（３次元→２次元への変更）.
18 met = temp[0, :, :]
19
20
21
22
23 # 分布図の画面表示と png ファイル出力.
24 fig = plt.figure(num=None, figsize=(6, 4))
25 sclmin = 15.0          #最小値
26 sclmax = 30.0          #最大値
27 sclint =  1.0          #色の刻み
28 plt.axes(axisbg='0.8')   #背景を灰色に
29 levels = np.arange(sclmin, sclmax+sclint, sclint)
30 cmap = plt.cm.Spectral_r   #カラーマップを愛称で指定.「_r」を末尾に付けて反転.
31 cmap.set_over('w', 1.0)
32 cmap.set_under('k', 1.0)
33 CF = plt.contourf(lon, lat, met, levels, cmap=cmap, extend='both')
34 plt.colorbar(CF)
35 plt.title('Result')
36 plt.savefig('result'+'.png', dpi=600)
37 plt.show()
38 plt.clf()
39
40 # 分布図の保存.
41 AMD.PutCSV_Map(met, lat, lon)
42 AMD.PutNC_Map( met, lat, lon, description='None', symbol=element, unit='degC')
```

プログラムに手を加えて，2013年８月１日ではなく，８月９日の温度分布図を描くようにしてみます．それには，10行目の文；

```
timedomain = [ '2013-08-01', '2013-08-01' ]
```

を次のように書き換えます．

```
timedomain = [ '2013-08-09', '2013-08-09' ]
```

編集が完了したら，［Ctrl］＋s または，メニュー File＞Save で変更を保存します．なお，エディターを閉じる必要はありません．保存が終了したら，プログラムを実行します．先ほどと同様，プロンプトに「run mesh_map.py」と入力します．IPython コンソールには，ヒストリー機能とよばれる機能があり，過去に打ち込んだ文字列を記憶しているので，上矢印キー（↑）でこれを呼び出してエンターキーを押してもかまいません．

このように，Python でのプログラム開発は，エディターでの編集と IPython 上での実行結果の確認を繰り返して進めてゆきます．なお，IPython コンソールは，個々の文を実行することができます．また，実行が終了したプログラムで使用されていた変数を保持しているので，その内容等を表示させることもできます．

BOX３　Python について

Python（パイソン）は，使いやすさとグラフィクスの美しさから近年日本でも人気が出てきた，比較的新しいプログラミング言語です．オープンソースなので，だれでも自由に使用することができます．python（ニシキヘビ）という風変わりな名称は，この言語の開発者が，イギリスのテレビ番組「空飛ぶモンティ・パイソン」のファンだったからと言われています．

プログラミング言語は，実行の方式によりインタープリタ型とコンパイル型に大別することができ，Python は前者に属します．インタープリタ型は，プログラムに書かれた命令の一行一行を逐次解釈して実行してゆきます．このため，実行速度が遅い傾向にあり，特に，同じ計算を繰り返す様な処理が苦手です．反面，一行ずつ実行するので「とりあえずここまでプログラムを書いて実行の様子をみよう」というような場当たり的なプログラミングができ，とにかく結果を出したいというようなプログラムに向いています．

一方，コンパイル型は人間が書いたプログラムの全部をコンパイラ（翻訳機）でコンピュータが理解するコードに変換してそれを実行します．この方式は計算をとても早く実行することができる反面，プログラムがきちんとできるまでは動かすことができないので，大規模でしっかりとしたプログラムを作るのに適しています．

２　期間平均した最高気温を計算するプログラムの作成

プログラム「mesh_map.py」を編集して，８月１日から８月31日の期間について平均した日最高気温の分布を計算し，茨城県における分布図を作成する，より実用的なプログラムを作成してみます．

１）別名保存

もともとのファイルが無くなってしまわないよう，ファイルに別名をつけて保存します．mesh_map.py を専用エディターで開いた状態で，メインメニュー File＞Save As …を選択し，「mesh_

mean_map.py」に改名して保存します.

２）平均期間の設定

平均期間である８月１日から８月31日のデータをメッシュ農業気象データ配信サーバーから取得するために，10行目を次のように書き換えます.

```
timedomain = [ '2013−08−01', '2013−08−31' ]
```

３）気象要素の設定

配信サーバーから取り寄せる気象要素を，日平均気温から，日最高気温に変更します．それには，９行目，

```
element = 'TMP_mea'
```

を次のように書き換えます.

```
element = 'TMP_max'
```

平均値計算処理の追加：１月分＝31枚の日別最高気温分布データを時間方向に串刺しのように平均する処理を追加し，平均化された分布を得ます．それには，17－18行目，

```
#　データの整形（３次元から２次元へ変換）.
met = temp [0, :, :]
```

を，次のように書き換えます.

```
#　データを時間方向に（第０番目の次元について）平均する.
met = np. ma. mean (temp, axis=0)
```

「np.ma.mean( )」は，Python の数値計算モジュール numpy に定義される平均値を計算する関数で，第１引数には，計算対象の配列を，第２引数には，平均をする次元を指定します．メッシュ農業気象データは，日付，緯度，経度の３つの次元を持つデータです．このプログラムでは，時間方向の平均操作なので，「axis＝０」として最初の次元について平均することを指示しています．Python では，次元を０から数える約束なので，最初の次元を０で指示します.

４）茨城県以外の消去

メッシュ農業気象データ配信サーバーは，気象データだけでなく，各都道府県について，その都道府県が一部でも含まれる三次メッシュに１，全く含まれないメッシュには無効値が埋め込まれたメッシュデータが用意されています．そこで，このデータを利用して，最高気温の平均値分布図を茨城県だけに限定することにします．そのために，20－21行に次の文を書き入れます.

```
geo, la, lo  = AMD. GetGeoData ('pref_0800', 'Area3', lalodomain)
```

```
met = met * geo
```

初めの文は，配信サーバーから茨城県の領域が１であるメッシュデータを取り出す文です．GetGeoData( )は地理情報を呼び出す関数で，AMD_Tools に定義されています．茨城県の領域が１のデータであるということは文字列「pref_0800」で指定していて，最後の４桁の数字を変更すると，異なる県の領域データが変数 geo に代入されます．続く文で，このデータを分布図にかけ算しています．これで，変数 met の内容は，茨城県内の領域についてはそのままの値，県外の領域については無効値に変更されます．都道府県と数字の関係は表３を参照してください．

５）計算結果の利用

プログラム mesh_mean_map.py は，画面に表示するほか，計算結果を三種類のファイルで保存します．第１は，画像（.png）ファイルで，これは36行目の文で実行されます．第２は CSV ファイルで，42行目で作られます．第３は NetCDF（.nc）ファイルで，43行目で作成されます．このプログラムでは，何れも result.???の名で作成されています．

NetCDF ファイルは，Ⅴ章で解説するデータ可視化ソフト IDV やⅥ章で解説する地図ソフト GMT が直接読み込める特別な形式のファイルです．Result.nc を IDV で開くには，IDV を起動して Dashboard ウインドウの Data　Choosers のページを開き，左端のペイン（ウインドウ内の区切られた領域のこと）で Files が選択されていることを確認してから，右ペインのファイルブラウザから PythonWorks フォルダーを開いて Result.nc を選択し［Add　Source］ボタンを押します．

BOX４　数値計算モジュール numpy

numpy は，Python の機能を拡張するモジュールの１つで，ベクトルや行列などの計算を行うための関数や変数サポートが納められています．「np.ma.mean()」は，平均計算のための関数で，他に「np.ma.max()」（最大値），「np.ma.min()」（最小値），「np.ma.sum()」（積算値），なども提供されています．numpy を利用するには，プログラム冒頭でこのモジュールを使うことを宣言することが必要です．これが，４行目の文；

```
import numpy as np
```

です．この文により，Python は，冒頭が「np.」で始まる関数を，numpy というパッケージから探すようになります．なお，プログラム mesh_map.py は，numpy の他，グラフィックスモジュール matplotlib と，メッシュ農業気象データを利用するためのモジュール AMD_tools を使用しています．

３　指定地点の気象の経過グラフ

プログラム ts.py（BOX５）は，予めプログラム内に記述した地点における気温や降水量などの気象の日変化を，平年値とともに表示します（図23）．このプログラムを例に，Python プログラムの仕組みを解説します．

１）プログラム ts.py の実行

IPython コンソールのプロンプトに「run ts.py TMP_mea」と入力してエンターキーを押すと，プログラムが実行され日平均気温の日々変化が表示されます．このプログラムでは，プログラム名の後に気象要素の略号を付け加えて，プログラムに指示を与えています．このように，プログラム実行の際，追加的に付け加える情報をコマンドライン引数といいます．この場合は，文字列

BOX５　Python プログラム「ts.py」

```
#!/usr/bin/env python
# −*− coding: utf−8 −*−
#
# 特定地点における気象要素の時系列変化をグラフ表示します.
# 表示させる期間は変数"timedomain"に開始日と終了日を文字列で与えます.
# 表示させる地点は変数"lalodomain"に緯度，緯度，経度，経度で指定します.
# 特定の一地点を対象とするため，二つの緯度と経度にはそれぞれ同じ値を
# 指定します.
# IPython からの実行は次のようにします.
#
#   In [1]: run browse.py TMP_mea ＜エンター＞
#
# ここで, "TMP_mea"は，日平均気温を示す記号です．他の気象要素に対する記号は,
# 表を参照してください．また，次のように，気象要素記号の後ろに"−a"を付けると,
# 積算グラフを表示します.
#
#   In [1]: run browse.py APCP −a ＜エンター＞
#
#  気象要素略号
#  日平均気温　　TMP_mea
#  日最高気温　　TMP_max
#  日最低気温　　TMP_min
#  日平均相対湿度　RH
#  日照時間(準備中)  SSD
#  日射量　　GSR
#  下向き長波放射量  DLR
#  日積算降水量　　APCP
#  日平均風速(参考)  WIND
#   OHNO, Hiroyuki 2012.08.20
# ====================================================================
import sys # モジュール属性 argv を取得するため
import numpy as np
import datetime
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import AMD_Tools as AMD

argvs = sys.argv  # コマンドライン引数を格納したリストの取得
argc = len(argvs) # 引数の個数
if argc == 1:
    print 'Please specify a handle of the meteorological element.'
    print 'TMP_mea/TMP_max/TMP_min/RH/SSD/GSR/DLR/APCP/WIND'
    print 'なお，引数「−a」 をさらに追加すると，積算値のグラフを作成します.'
    sys.exit()

# 計算の領域と期間の指定
timedomain = [ '2013−01−01', '2013−12−31' ]
lalodomain = [ 36.0566, 36.0566, 140.125, 140.125]#つくば（舘野）
area = 'Area3'
element = argvs[1]
```

```
51
52 # データの取得
53 T1, tim, lat, lon = AMD.GetData(element, area, timedomain, lalodomain)
54 T0, tim, lat, lon, nam, uni = AMD.GetData(element, area, timedomain, lalodomain, cli=1, namuni=1)
55
56 T0 = T0[:,0,0]
57 T1 = T1[:,0,0]
58 if argc == 3 and argvs[2] == '−a':
59    for i in range(len(T0)):
60       T0[i] = T0[i] + T0[i−1]
61       T1[i] = T1[i] + T1[i−1]
62
63
64 # 表示
65 # ・領域の作成
66 fig = plt.figure(num=None, figsize=(12, 4))
67 # ・目盛の作成
68 ax = plt.axes()
69 xmajoPos = mdates.DayLocator(bymonthday=[1])
70 xmajoFmt = mdates.DateFormatter('%m/%d')
71 ax.xaxis.set_major_locator(xmajoPos)
72 ax.xaxis.set_major_formatter(xmajoFmt)
73 xminoPos = mdates.DayLocator()
74 ax.xaxis.set_minor_locator(xminoPos)
75 # ・データのプロット
76 plt.fill_between(tim,T1,T0,where=T0<T1,color='orange',alpha=0.5)    #高温部を橙色
77 plt.fill_between(tim,T0,T1,where=T1<T0,color='skyblue',alpha=0.5)   #低温部を水色
78 plt.plot(tim, T0, 'k', linewidth=0.3)                               #平年値の線
79 plt.plot(tim, T1, 'k')                                             今年の線
80 # ・「今日」印を付ける
81 p = datetime.datetime.today()                                      #「今日」の時刻オブジェクト
82 today = tim == datetime.datetime(p.year,p.month,p.day,0,0,0)       #今日の配列要素番号
83 plt.plot(tim[today], T1[today], "ro")                              #今日に赤点を打つ
84 # ・タイトルの付加
85 plt.xlabel('Date')
86 plt.ylabel(argvs[1] + ': ' + nam + ' (' + uni + ')')
87 plt.title('N'+str(lalodomain[0])+', E'+str(lalodomain[2])+' ('+str(p.year)+'/'+str(p.month)+'/'+str(p.day)+')')
88 # ・図の保存
89 plt.savefig('Fig_'+argvs[1]+'.png', dpi=300)
90 plt.show()
91
92 # 計算結果の保存
93 Table = np.array([T0,T1])
94 AMD.PutCSV_TS(Table, tim, header='Date,Normal,Obs.')               #CSV ファイル出力
```

であって数ではありませんが TMP_mea がコマンドライン引数です.

２）Python によるデータ処理

プログラム ts.py で実行される処理について，順次解説を加えます.

１〜30行：プログラムの名称や使い方を書いた説明です．Python では，「＃」記号の後ろは無視

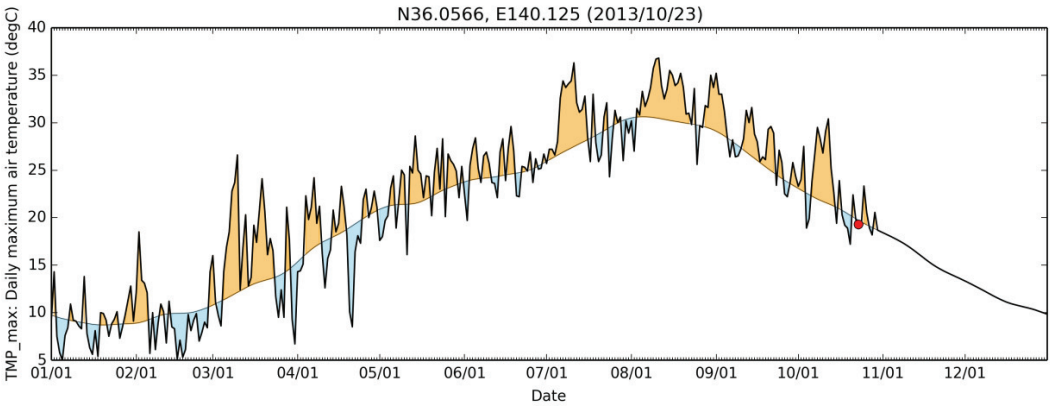

図23. ts.py の実行結果

されるので，＃を書いてからコメントを記述します．プログラムを作成しているときには，動作確認のために特定の文を一時的に無効にしたいときがあります．そのような時にもこれを挿入することがあり，そのような操作を「コメントアウトする」などと言うことがあります．

31〜36行：このプログラムでは，描画や日付計算など，素の Python に組み込まれていない機能を使用するので，import 文を使ってそれぞれの機能を提供するモジュールを組み込みます．このとき，「as」を使用して，モジュール名に別名をつけることができます．

38行：この文によって，引数をプログラム変数 argvs に格納することができます．IPython コンソールは，直前に実行したプログラムの変数の内容を保持しているので，プロンプトに続けて「argvs」と打ち込むとこの変数の中身を見ることができます．

39行：関数 len（）は，括弧の中の変数の個数を求めます．

40〜44行：条件分岐文（if 文）で，if とコロン「：」に挟まれた式が真であればインデントされている44行まで実行し，偽の場合はこれらの文をひとまとめに無視します．なお，44行はプログラムを直ちに終了する命令文です．関数 argv は，最初に入力した文字列 （この場合'ts.py'）を第一番の要素として必ず返すことになっています．そのため，argvs の要素数が１と言うことは，'ts.py'に付随する引数が０個ということです．

47行：日々変化グラフの日付範囲を「yyyy−mm−dd」の形式の文字列で指定します．年を跨る指定もできます．

48行：グラフを作成する地点の緯度と経度を数値で指定します．指定する場所が点の場合は同じ緯度と同じ経度を繰り返し指定します．このプログラムでは，特定の１メッシュにおける気象値を使用しますが，面の場合には緯度と経度の範囲を指定します．

49行：データを取得する地点が属するデータ領域名を文字列で指定します．48行で指定しているのは茨城県にあるアメダスポイント「つくば（館野）」の緯度経度で，この地点は「Area３」に含まれています．

53行：36行でインポートした AMD_Tools モジュールの中の関数 GetData（）を用いてメッシュ農業気象データを配信サーバーから読み込みます．47－50行で指定した期間や緯度経度などの情報がこの関数に与えられています．関数は，気象データ，時間の並び，緯度の並び，経度の並びの４種類のデータを左辺に用意した変数に格納します．

54行：関数 GetData（）は，引数の中に「Cli＝１」というオプション引数が含まれていると，平年値を返します．したがって，変数 T0には平年値が格納されます．また，この関数は，通常は４つのデータを左辺に返しますが，「namuni＝１」というオプション引数が含まれている

と，返されるデータに気象要素名と単位の２つが加わり合計６個になります．このため，それらを受け取るために，左辺には６個の変数が並んでいます．

56，57行：大括弧の中のコロン（：）は「すべての要素」を意味します．メッシュ農業気象データは，時間，緯度，経度からなる３次元「空間」に並べられたデータなので，一般には，何グリッド目の時間，何グリッド目の緯度，何グリッド目の経度という３種類の添え字を指定して気象データを特定します．しかし，このプログラムでは単一メッシュのデータしか扱わないので，緯度と経度についての添え字を識別する必要がありません．これは，建物が１つしかないアパートに「１号棟」という番地情報が不要なのと似ています．これらの文は，３次元の体裁で得たデータを１次元の体裁に変更するために書かれています．

58行：気象データを積算するかしないかを判断する if 文です．引数に「−a」が追加指定されたときは，引数の数が２であり，かつ，２個目（python は０から数える約束）の要素が「−a」であることを条件としています．この条件が成り立つ場合は59行以下を処理します．積算グラフは，降水量を表示する時などに便利です．

59行：気象データを積算する繰り返し計算を指示する文（for 文）で，この文よりもう一段深いインデントの部分を繰り返します．for 文にある「in」の後ろには，数字や文字の列（Python ではリストと呼びます）を置く決まりになっていて，これらのリストが順に変数 i に読み込まれてインデント部分が繰り返されます．range（）は関数で，引数の数の連番のリストを作り出します．注意しなければならないのは，関数 range（）は，引数の一つ手前の数までしか連番を作らないことです．

60行：繰り返し（for 文）の指定により，まず，日付１の気象データに日付０のデータが加えられて日付１に格納し直されます．次の繰り返しでは，日付２のデータに（日付０と日付１の和となっている）日付１のデータが加えられて日付２のデータとして格納されます（つまり日付０，日付１，日付２の和が格納される）．同様にしてその次の繰り返しでは，日付３のデータに日付０～３までの和が格納しなおされます．この繰り返しにより，積算データが作り出されます．

３）Python での作図

プログラム ts.py の64行～90行は，気象データと平年値データをグラフにする部分です．この部分では，図を書く場所を用意し，データ目盛り線や目盛り数値を決め，データの折れ線を描き，プログラム実行日の丸印を打ち，グラフの上にタイトル文字を入れ，図をファイルや画面に出力しています．Python の描画モジュール matplotlib はきわめて表現力に富み，かつ多機能ですが，その分，グラフ各要素を指定する文は複雑難解です．ここでは，大まかな機能を紹介するにとどめます．

66行：横12インチ縦４インチの作図領域を用意します．

69−74行：横軸の目盛りを日付で振ることを指定します．

76行：T0＜T1，すなわち，気象値が平年値よりも大きい部分を薄橙色に着色します．

77行：T1＜T0，すなわち，気象値が平年値よりも小さい部分を薄水色に着色します．

78，79行：T1や T0，すなわち，気象値や平年値の折れ線を黒線で引きます．

80−83行：プログラムを実行した日の気象値の場所に赤丸を打ちます．

85行：横軸の見出しを書き込みます．

86行：気象要素の略号，名称，単位を縦軸の見出に書き込みます．

87行：図のタイトルに，地点の緯度経度，作図年月日を書き込みます．

89行：図を png ファイルで出力します.

93行：365行×１列の配列である T0と T1を結合して365×２列の配列とします.

94行：平年値と気象値を日付とともに CSV ファイルとして出力します.「AMD.PutCSV_TS」は，AMD_Tools.py に書かれている PutCSV_TS（）という関数を呼び出すことを意味しています.

BOX６　Python とインデント

文の行頭を字下げ（インデント）して，プログラムを見やすくすることは，多くのプログラム言語で慣用となっていますが，Python ではループなどひとまとまりで扱うべき文をインデントそのもので表現します．たとえば，５の階乗（１×２×･･×５）を計算するプログラムは，次のように書かれます.

```
k = 1
for i in [1, 2, 3, 4, 5]:
  k = k * i
print k
```

この例では，繰り返し計算の中身は３行目だけで，これが５回繰り返されてから４行目に移ります．どこからどこまでを繰り返すかは同じインデントがどこからどこまで施されているかで決まります.「end for」など，ループの終わりを示す文はありません．従って，次のように，４行目をインデントすると，画面には，計算途中の値が５回表示されるようになります.

```
k = 1
for i in [1, 2, 3, 4, 5]:
  k = k * i
  print k
```

インデントには半角空白とタブが使用できますが，両者は絶対に混用しないようにして下さい．両者が混ざっていると，人の目からは同じインデントでありながら Python には異なるインデントと見なされエラーとなります.

一方で，python は括弧の途中では自由に改行やインデントができます．たとえば，関数にたくさんの引数があり，一行で書くととても長くなるようなとき，見やすくなるよう，括弧のなかの適当な場所で改行し字下げをすることができます.

```
T0, tim, lat, lon, nam, uni = AMD.GetData(element, area,
                timedomain, lalodomain, cli=1, namuni=1)
```

４．茨城県における水稲の発育を推定するプログラム

１）DVI/DVR 法

DVI/DVR 法とは，作物の発育段階を，出芽を０，成熟を２とする発育指数（DVI）で表現する方法で，DVI は日々の気象条件から計算される発育速度（DVR）を出芽日以降積算したものです．この方法は，水稲の出穂日や収穫日を推定するのにとても便利なので，DVR の式を栽培試験から独自に作成したり，DVI の起点を出芽ではなく移植日として定義するなど，地域の実情にあわせた様々なアレンジが加えられて普及や指導の現場で広く使われています.

２）プログラム RiceDevelopment.py の概要

プログラム RiceDevelopment.py（BOX７）は，DVI/DVR 法に基づいて，県下の水稲の発育

を予測し，出穂日の分布図，成熟日の分布図を作成するとともに，日々のDVIの値をNetCDFファイルとして出力します．このプログラムでは，DVIの定義を，出芽で０，出穂で１，成熟で２としています．そして，気象条件からDVRを計算する際，移植～出穂の期間と出穂～成熟の期間とで異なる計算式を用いています．また，実際には，水稲の出芽日や移植日が県下で同一ということはありませんが，「苗は同一日に移植される」，「移植時の苗のDVIは県下で同一である」という二つの仮定をおいて分布図を作成しています．

プログラムの実行は，IPythonのプロンプトに「run　RiceDevelopment.py」と打ち込み，エンターキーを押します．すると，しばらく計算した後に，出穂日の分布図と成熟日の分布図が図24のように表示されます．あわせてこれらの画像ファイルと，DVI分布の日々の変化を記録したNetCDFファイルが出力されます．

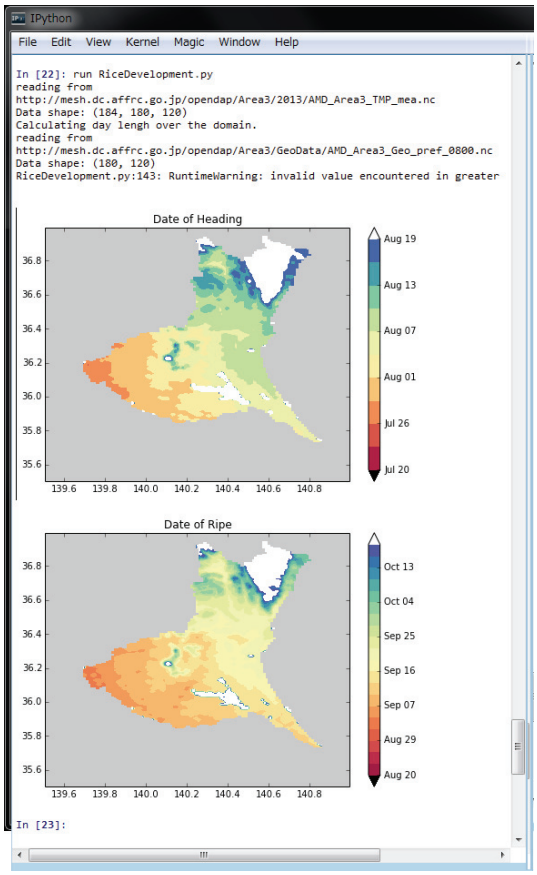

図24.　RiceDevelopment.pyの実行結果

IPythonコンソール上に分布図が描画される．

実際の利用あたっては，移植日と移植時のDVI値，それに，DVRの式を適切に与える必要があります．移植日はプログラム55行で設定します．この文は計算期間を設定する文で，計算開始日を移植日に設定します．計算終了日については，大まかに予想される収穫日の数週間後を目処に設定します．移植時のDVI値は，プログラム52行で指定します．DVRの式の与え方については，次の節で詳しく説明します．

３）独自のDVR関数を使用する方法

DVI/DVR法では，作物の発育の進行を気象条件から計算したDVRという量で表現します．DVRは，地域や品種に応じて様々なものが考案され使用されています．したがって，DVI/DVR法で作物の生長を推定するプログラムでは，DVRを計算する式を柔軟に入れ替えられるようにしておくと便利です．これを実現するプログラミングの方法の一つに，関数の定義という方法があります．これは，特定の計算や処理を行うプログラムを本体とは切り離してプログラミングして名前をつけておき，プログラム本体でそれを呼んで所定の計算や処理をさせるものです．

関数は下のような書式で定義します；

```
def　関数名（引数１，引数２，…）：
  計算文
  計算文
    :
  return　戻り値が入っている変数
```

ここで，引数（ひきすう）とは，呼ぶ側のプログラムからの数値を受け取る変数のことで，２

通りの表記法があります．一つは，引数となる変数だけが書かれているもの，もう一つは引数となる変数に等号と値が付随するものです．前者のように引数を書くと，それは，関数を呼ぶ際に必ず与えなければならない変数値となります．後者のように書くと，関数を呼ぶ際にこの引数は値を指定してもしなくてもよい引数（キーワード引数）となり，省略されて関数が呼び出された場合には等号の右側の値がセットされます．

プログラム RiceDevelopment.py では，７行～30行で出芽から出穂までを受け持つ DVR 関数を DVR01 () という名前で定義し，32行～45行で出穂から成熟までを受け持つ DVR 関数を DVR 12 () という名前で定義しています．７行目を見てわかるとおり，関数 DVR01 () は DVI，Ta，Ld という３個の通常の引数と Para という１個のオプション引数を持ちます．この関数は，49行目から始まるプログラム本体の中の93行目で呼び出されています．呼び出される際，プログラム本体で使用している DVI，Tmea，Ld という変数の値がこの順で関数 DVR01 () の DVI，Ta，Ld に引き渡されます．キーワード引数 para は指定されていないので，７行に記されているデフォルト値が用いられます．ここで，プログラム本体で使用している変数の名前と，通常の引数の名前は一致している必要がないことに注意してください．関数への値の引き継ぎには，数と順序だけが意味を持ちます．これに対し，キーワード引数は省略ができるものの，使用するときは定義で使われている名前を使用する必要があります．

７行に記されている para のデフォルト値は，この DVR 関数のコシヒカリに対するパラメータです．したがって，この関数を使用して他の水稲品種の発育を推定する場合は，次のようにしてキーワード引数 para にその品種に対するパラメータ値を明示的に与える必要があります．

```
DVR = DVR01(DVI[t−1, i, j], Tmea[t, i, j], Ld[t, i, j], Para=[51.3, 17.8, 0.365, 16.0, 0.566, 0.23])
```

プログラム RiceDevelopment.py において，気象データを取得したり DVR を積算したり，グラフを描いたりするプログラムの本体は，48行以降にあります．プログラムの本体も一塊のプログラムですから，プログラミング言語によっては，main という名で定義することがありますが，Python では，def 文を使用せず，if 文を使います．これは Python の少々風変わりなところですが，このような扱いにすることで，RiceDevelopment.py をそのままモジュールとして DVR 関数を使用する他のプログラムにインポートして使うことができます．

５．モジュール AMD_tools

モジュール AMD_tools は，メッシュ気象データの利用に必要な関数（計算で使う道具）のコレクションで，ファイル AMD_tools.py を作業ディレクトリに置き，プログラムでこれをインポートすることで使用可能となります．モジュールには，メッシュ農業気象データを処理するのに便利な９個の関数が定義されていています．

１）GetData（）

概要：メッシュ農業気象データをデータ配信サーバーまたはローカルファイルから取得する関数．

書式１：ret1, ret2, ret3, ret4 = GetData (element, area, timedomain, lalodomain, cli=0, url='http://mesh.dc.affrc.go.jp/opendap')

書式２：ret1, ret2, ret3, ret4, ret5, ret6 = GetData (element, area, timedomain, lalodomain, namuni=1, url='http://mesh.dc.affrc.go.jp/opendap')

引数：

BOX７　Python プログラム「RiceDevelopment.py」

（誌面の都合上，折り返している行がありますが，実際は１行で記述します.）

```
#! c:/Python27/python.exe
# −*− coding: utf−8 −*−
#
import numpy as np

# 独自定義の DVR 関数−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
def DVR01(DVI, Ta, Ld, Para=[51.3,17.8,0.365,16.0,0.566,0.23] ):
#          Para=[ Gv0, Th ,  A , Lc ,  B ,DVI*]
#   Function name:    'Horie et al. (1995)'
#   Varid phase: emergence to heading
#   Description:
#     Gv0 [days]: the minimum number of days required for heading (GV)
#     Th [C]: the temperature at which DVR is half the maximun rate at the optimum temperature (TH)
#     A [1/C]: empilocal Parameter on air temperature (ALF)
#     Lc [hour]: critical day length (LC)
#     B [1/hour]: empilocal Parameter on day length (BDL)
#     DVI* [ ]: DVI at which the crop becomes photosensitive (DVSAS).
#     For KoshHikari;
#       Gv0 ,   Th ,    A ,    Lc ,     B ,  DVI*
#       51.30, 17.80, 0.365, 16.00, 0.566, 0.230
#
#                      Gv0                 A          Th
    FT = np.max( 1.0 / (Para[0] * (1.0 + np.exp(−Para[2] * (Ta − Para[1])))), 0.0)
#                           B             Lc
    FL = np.max((1.0 − np.exp(Para[4] * (Ld − Para[3]))), 0.0)
    DVR = FT
#           DVIs
    if DVI > Para[5] :
        DVR = FT * FL
        return DVR

def DVR12(Ta, Para=[0.0, 1000.0]):
#           Para=[ T0,  EDDd ]
#    Function name:    'Normalized Effective Degree Days'
#   Varid phase: not specified
#    Description:
#     T0 [degC]: threshold temperature of development
#     EDDd [degC day]: desired effective degree days
#     For the symple cumlative temperature of 1000[degC day];
#       T0,  EDDt
#       0.0, 1000.0
#
#              T0       EDDd
    DVR = (Ta − Para[0]) / Para[1]
    return DVR


# プログラムのメイン関数−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
if __name__ == "__main__":

```

```
51    #移植時 DVI の指定
52    DVI0 = 0.2
53
54    #計算の領域と期間の指定
55    timedomain = [ '2013-05-01', '2013-10-31' ] #計算の開始日を移植日と見なす.
56    lalodomain = [  35.5,  37.0,  139.5, 141.0]
57    area = 'Area3'
58    pref = 'pref_0800'  #茨城県
59
60    #各種データの準備
61    import AMD_Tools as AMD        #メッシュデータ用モジュールをインポート.
62    import DayLength as DL         #日長を計算するモジュールをインポート.
63  #   Tmea, tim, lat, lon = AMD.GetData('TMP_mea', area, timedomain, lalodomain, url='./AMD') #USB
    メモリに保存されている気象データを使用するときはこちらを使い，下の行は消去します.
64    Tmea, tim, lat, lon = AMD.GetData('TMP_mea', area, timedomain, lalodomain)  #配信サーバーの気
    象データを使用するときはこちらを使い，上の行は消去します.
65    Tmea[Tmea.mask == True] = np.nan
66    Ld = DL.daylength(tim, lat, lon)   #モジュール daylengthd で対象時空間全部の日長を計算する.
67    Pref, lat, lon = AMD.GetGeoData(pref, area, lalodomain, url='./AMD')   #USB メモリに保存されて
    いる県域データを使用するときはこちらを使い，下の行は消去します.
68  #   Pref, lat, lon = AMD.GetGeoData(pref, area, lalodomain)  #配信サーバーかの県域データを使用す
    るときはこちらを使い，上の行は消去します.
69
70    #計算結果を保存する配列の定義
71    frst = np.datetime64(timedomain[0])   #期間の初日を数値化したもの
72    last = np.datetime64(timedomain[1])   #期間の最終日を数値化したもの
73    ntim = Tmea.shape[0]   #日数
74    nlat = Tmea.shape[1]   #メッシュの行数
75    nlon = Tmea.shape[2]   #メッシュの列数
76    DVI = np.ma.zeros((ntim, nlat, nlon))   #DVI の時空間分布を記録する配列（気象データと同じ時空
    間サイズ）を確保する.
77    DVI[:,Pref==0.0] = np.ma.masked   #県外にマスクをかける.
78    DVI[:,:,:] = DVI0              #全領域に初期値を代入.
79    DOH = np.ma.zeros((nlat, nlon), dtype='datetime64[D]')   #出穂日（期間の初日からの日数）をしま
    う配列.
80    DOH[Pref==0.0] = np.ma.masked   #県外にマスクをかける.
81    np.ma.harden_mask(DOH)         #マスクを"固く"して以降の計算でも変化しないようにする.
82    DOH[:,:] = last                #全領域に期間の最終日を入れておく.
83    DOR = np.ma.zeros((nlat, nlon), dtype='datetime64[D]')   #収穫適日（期間の初日からの日数）をし
    まう配列.
84    DOR[Pref==0.0] = np.ma.masked   #県外にマスクをかける.
85    np.ma.harden_mask(DOR)
86    DOR[:,:] = last
87
88    #各メッシュにおける DVI の計算
89    for i in range(nlat):   #メッシュ行（緯度方向）
90      for j in range(nlon):   #メッシュ列（経度方向）
91        for t in range(1, ntim):   #日数
92          if DVI[t-1,i,j] < 1.0:   #移植から出穂までは…
93            DVR = DVR01(DVI[t-1,i,j], Tmea[t,i,j], Ld[t,i,j])
94            DVI[t,i,j] = DVI[t-1,i,j] + DVR
95            if DVI[t,i,j] >= 1.0:   #出穂日ならば…
96              DOH[i,j] =  frst + np.timedelta64(t,'D')   #日付を記録する.
```

```
97              else:                   #出穂から成熟までは…
98                DVR = DVR12(Tmea[t−1,i,j])
99                DVI[t,i,j] = DVI[t−1,i,j] + DVR
100               if DVI[t−1,i,j] < 2.0 and DVI[t,i,j] >= 2.0:   #収穫適日ならば…
101                 DOR[i,j] = frst + np.timedelta64(t,'D')      #日付を記録する.
102
103     DVI[np.ma.where(DVI > 2.0)] = 2.0    #全体を見直してみて，DVI が2より大きいメッシュがあればそれを2.0に置きかえてしまう.
104   #   DVI[DVI>2.0] = 2.0 でも同じ.
105     # 計算結果の描画
106     import matplotlib.pyplot as plt
107     from matplotlib.dates import DateFormatter,DayLocator
108     import matplotlib.pylab as plt
109     import matplotlib.colors as clr
110     aspect = (lalodomain[3] − lalodomain[2]) / (lalodomain[1] − lalodomain[0]) + 0.5
111   #   一枚目の図_____
112     fig = plt.figure(num=None, figsize=(5*aspect, 5))  #図のオブジェクト（入れ物）を定義
113     sclint = 3                          #カラーバーの刻み
114     sclmin = np.datetime64('2013−07−21')       #カラーバーの最小値
115     sclmax = np.datetime64('2013−08−20')       #カラーバーの最大値
116     levels = np.arange(sclmin, sclmax+np.timedelta64(sclint,'D'), sclint)
117     plt.axes(axisbg='0.8')          #背景を灰色に
118     cmap = plt.cm.Spectral       #カラーマップを愛称で指定.「_r」を末尾に付けて反転.
119     cmap.set_over('w', 1.0)       #上限を超えたときの色
120     cmap.set_under('k', 1.0)      #下限を超えたときの色
121     CF = plt.contourf(lon, lat, DOH, levels, cmap=cmap, extend='both')   #分布図を描く
122     CB = plt.colorbar(CF, format=DateFormatter('%b %d'))        #カラーバーを描く
123     plt.title('Date of Heading')                    #タイトルを書く
124     plt.savefig('Date_of_Heading.png', dpi=600)        #図をビットマップ画像にする
125     plt.show(fig)                       #図を表示する
126     plt.clf()
127   #   二枚目の図_____
128     fig = plt.figure(num=None, figsize=(5*aspect, 5))
129     sclint = 3
130     sclmin = np.datetime64('2013−08−21')
131     sclmax = np.datetime64('2013−10−20')
132     plt.axes(axisbg='0.8')
133     levels = np.arange(sclmin, sclmax+np.timedelta64(sclint,'D'), sclint)
134     cmap = plt.cm.Spectral
135     cmap.set_over('w', 1.0)
136     cmap.set_under('k', 1.0)
137     CF = plt.contourf(lon, lat, DOR, levels, cmap=cmap, extend='both')
138     CB = plt.colorbar(CF, format=DateFormatter('%b %d'))
139     plt.title('Date of Ripe')
140     plt.savefig('Date_of_Ripen.png', dpi=600)
141     plt.show(fig)
142     plt.clf()
143
144     # 計算結果をファイルに保存します.
145     AMD.PutNC_3D( DVI, tim, lat, lon, description='Rice Development Index',
146                   symbol='DVI', unit='−', filename='DVI.nc')
```

element：気象要素記号で，'TMP_mea'などの文字列で与える
area：データの領域で，'Area3'などの文字列で与える
timedomain：取得するデータの時間範囲で，［'2008−05−05'，'2008−05−05'］のような文字列の２要素リストで与える．特定の日のデータを取得するときは，二カ所に同じ日付を与える．
lalodomain：取得するデータの緯度と経度の範囲で，［36.0，40.0，130.0，135.0］のように緯度経度の順で指定する．特定地点のデータを取得するときは，緯度と経度にそれぞれ同じ値を与える．
cli：平年値のデータを取得するとき cli＝１として与える．１以外の数値であったり，このパラメータそのものが省略されている場合は，観測値が返される．
namuni：変数の名前と単位を取得するとき namuni＝１として与える．このとき，関数の戻り値の数は２つ増えて６つになる．１以外の数値であったり，このパラメータそのものが省略されている場合は，戻り値は４つ（変数，時刻，緯度，経度）である．
url：データファイルの場所を指定する．省略した場合はデータ配信サーバーに読みに行く．ローカルにあるファイルを指定するときはディレクトリー構造をデータ配信サーバーと同一（図３）とし，AreaN（N＝１～６）の上までの場所（通常は"…/AMD"）を指定する．
戻り値：
ret１：指定した気象要素のマスク付きの三次元データ．［時刻，緯度，経度］の次元を持つ．なお，マスクとは対象とする／しないを示す配列変数の付随情報のことで，メッシュ農業気象データシステムでは水域などデータのある／なしを表すのに用いています．
ret２：切り出した気象データの時刻の並び．Python の時刻オブジェクトの一次元配列である．時刻オブジェクトとは，Python で時刻を表現するために使用される特別な形式のデータで実数ではない．このため，Excel の時刻連番のように，１を足して翌日を表現するというようなことはできない．
ret３：切り出した気象データの緯度の並び．実数の一次元配列である．
ret４：切り出した気象データの経度の並び．実数の一次元配列である．
ret５：オプション引数「namuni＝１」が指定されたときに限り，気象要素の名称の文字列が返される．
ret６：オプション引数「namuni＝１」が指定されたときに限り，気象要素の単位の文字列が返される．
使用例１：以下により，北緯35度，東経135度の地点の2008年１月１日～2012年12月31日の日最高気温が一次元の配列変数 Tm に格納される．関数 GetData（）は，特定単一メッシュにおける複数日の日別値を取得する場合でも三次元配列を返すので，これを一次元配列に変換するために「Tm = Tm3D [:, 0, 0]」が実行されている．

```
import numpy as np
import AMD_Tools as AMD
timedomain = ['2008−01−01', '2012−12−31']
lalodomain = [35.0, 35.0, 135.0, 135.0]
Tm3D, tim, lat, lon = AMD.GetData('TMP_max', 'Area4', timedomain, lalodomain)
Tm = Tm3D[:,0,0]
```

使用例２：以下により，北緯35～36度，東経135～136度の範囲地点における日最高気温の日別平

年値の分布が三次元配列変数 Tmo に格納される．平年値であるから2011年の10月１日～12月31日と2012年10月１日～12月31のデータについては互いに等しい値が格納される一方，時刻オブジェクト配列 tim は，2011年10月１日から2012年12月31日までのすべて異なる値の日付オブジェクトが格納される．

```
import numpy as np
import AMD_Tools as AMD
timedomain = ['2011-10-01', '2012-12-31']
lalodomain = [35.0, 36.0, 135.0, 136.0]
Tmo, tim, lat, lon = AMD.GetData('TMP_max', 'Area4', timedomain, lalodomain, cli=1)
```

使用例３：以下により，北緯35～36度，東経135～136度の範囲地点における日最高気温の日別分布が，データ配信サーバーではなく，コンピュータの D:￥data￥以下に置かれたデータファイルから取得される．この際，D:￥data￥以下のディレクトリ構造は，データ配信サーバーと同じでなければならない．

```
import numpy as np
import AMD_Tools as AMD
td = ['2011-10-01', '2012-12-31']
lalo = [35.0, 36.0, 135.0, 136.0]
ele = 'TMP_max'
Tmo, tim, lat, lon = AMD.GetData(ele, 'Area4', td, lalo, url='D:￥data￥')
```

２）GetGeoData（）

概要：土地利用や都道府県などの地理情報をデータ配信サーバーまたはローカルファイルから取得する関数.

書式１：ret1, ret2, ret3 = GetGeoData(element, area, lalodomain, url='http://mesh.dc.affrc.go.jp/opendap')

書式３：ret1, ret2, ret3, ret4, ret5 = GetGeoData(element, area, timedomain, lalodomain, namuni=1, url='http://mesh.dc.affrc.go.jp/opendap')

引数：

element：地理情報の記号で, 'landuse_H210100'などの文字列で与える.

area：データの領域で, 'Area３'などの文字列で与える

lalodomain：取得するデータの緯度と経度の範囲で，［36.0，40.0，130.0，135.0］のように緯度経度の順で指定する．特定地点のデータを取得するときは，緯度と経度にそれぞれ同じ値を与える.

namuni：変数の名前と単位を取得するとき namuni＝１として与える．このとき，関数の戻り値の数は２つ増えて６つになる．１以外の数値であったり，このパラメータそのものが省略されている場合は，戻り値は４つ（変数，時刻，緯度，経度）である.

url：データファイルの場所を指定する．省略した場合はデータ配信サーバーに読みに行く．ローカルにあるファイルを指定するときは，AreaN（N＝１～６）の上までの場所を指定する.

戻り値：

ret１：指定した地理情報のマスク付きの二次元データ．［緯度，経度］の次元を持つ．
ret２：切り出した気象データの緯度の並び．実数の一次元配列である．
ret３：切り出した気象データの経度の並び．実数の一次元配列である．
ret４：オプション引数「namuni＝１」が指定されたときに限り，地理情報の名称の文字列が返される．
ret５：オプション引数「namuni＝１」が指定されたときに限り，地理情報の単位の文字列が返される．
使用例１：以下により，北緯35～36，東経135～136度の範囲にある各メッシュの水田面積比率の分布が取得される．

```
import numpy as np
import AMD_Tools as AMD
lalodomain = [35.0, 36.0, 135.0, 136.0]
Pad, lat, lon = AMD.GetGeoData(’landuse_H210100’’, ’Area4’, lalodomain)
```

使用例２：以下により，北緯35～36度，東経135～136度の範囲にある各メッシュの水田面積比率の分布が取得される．データは，データ配信サーバーではなく，コンピュータの D:￥data￥以下に置かれたデータファイルから取得される．この際，D:￥data￥以下のディレクトリ構造は，データ配信サーバーのそれ（図３）と同じでなければならない．地理情報は日々更新されないので，よく使うものをローカルストレージに保存しておくとネットワークの負荷を軽減できる．

```
import numpy as np
import AMD_Tools as AMD
lalo = [35.0, 36.0, 135.0, 136.0]
handle = ’’landuse_H210100’
Pad, tim, lat, lon = AMD.GetGeoData(handle, ’Area4’, lalo, url=’D:¥data¥’)
```

３）GetCSV（）

概要：CSV 形式のテキストファイルを配列変数に読み込む関数．配列の列数は取り込み範囲の先頭行で判別され，行数は EOF までの行数から判別する．文字列“nan”は，numpy.nan（数値ではないと理解される特別な数値）として理解し，これ以外の文字列が検出されると警告文を表示する．
書式：ret1 = GetCSV(filename, skiprow=0, fill=9.96921e+36)
引数：
filename：読み込むべき CSV ファイルの名前．
skiprow：余白や見出しなどに使用されていて読み込み対象としない行の数．指定を省略した場合は０に設定される（デフォルト値）．
fill：無効値として使用する数値．この値の配列要素にはマスクがかけられる．指定を省略した場合は9. 96921 e ＋36に設定される．
戻り値：ret１：指定した fill で指定した値のメッシュにマスクがかけられた二次元データ．
使用例１： CSV ファイル data.csv が BOX８のように与えられているとき，下を実行すると，結果は図25のように出力される．大括弧 ’[’ の付き方から，CSV ファイルの内容は，５行４

列の配列 arr に代入されていることがわかる（小数点以下８位以降に変換に伴う誤差が発生している）.

```
import numpy as np
import AMD_Tools as AMD
fn = 'data.csv'
arr = AMD.GetCSV(fn, skiprow=1, fill=−999.9)
print arr
```

BOX８　テキストファイル data.csv

```
lat, lon, dat1, dat2
35.0, 135.0, 1.5, 230.0
35.0, 135.0, 1.7, 253.6
35.0, 135.0, 2.2, nan
35.0, 135.0, 1.4, −999.9
35.0, 135.0, 0.9, 260.3
```

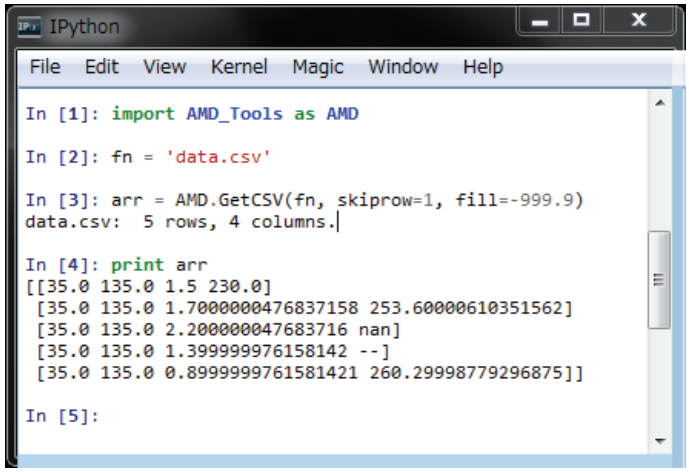

図25.　関数 Get_CSV () の使用例

文字から数値への変換に際し若干の誤差が発生している.

４）PutNC_Map（）

概要：２次元の気象変量（平面上に分布する気象要素）を NetCDF 形式のファイルで出力する関数.

書式：PutNC_Map(Var, lat, lon, description='Variable', symbol='Var', unit='−−', fill=9.96921e+36, filename='result.nc')

引数：

Var：気象変量として書き出す２次元配列変数．Var［緯度の次元，経度の次元］でデータが並んでいること.

lat：気象変量の各要素が並ぶ緯度を示す１次元配列．Var の最初の次元の要素数と一致していなくてはならない.

lon：気象変量の各要素が並ぶ経度を示す１次元配列．Var の２番目の次元の要素数と一致していなくてはならない.

description：気象変量の名前等を description = '名前等' として指定する．指定を省略した場合は 'Variable' という名で出力される.

symbol：気象変量の記号を symbol = '記号' として指定する．指定を省略した場合は 'Var' が使用される.

unit：気象変量の記号を unit = '単位の記号' として指定する．指定を省略した場合は '−−' が使用される.

fill：無効値として使用する数値を fill＝数値で指定する．指定を省略した場合は9. 96921e＋36に設定される．

filename：出力される NetCDF ファイルのファイル名を filename＝'ファイル名'として指定する．指定を省略した場合は 'result.nc' という名で出力される．

戻り値：戻り値はない．

使用例：以下により，データ配信サーバーから北緯35～36度，東経135～136度の範囲における水田占有率データを取得し，PaddyMap.nc という名の NetCDF ファイルとして出力する．

```
import AMD_Tools as AMD
import numpy as np
lalodomain = [35.0, 36.0, 135.0, 136.0]
Pad, lat, lon = AMD.GetGeoData('landuse_H210100'', 'Area4', lalodomain)
AMD.PutNC_Map( Pad, lat, lon, description='Ratio of paddy land', symbol='Rpad',
        unit='%', fill=9.96921e+36, filename='PaddyMap.nc')
```

5）PutNC_3D（）

概要：３次元の気象変量（時空間上に分布する気象要素）を NetCDF 形式のファイルで出力する関数．

書式：PutNC_3D(Var, tim, lat, lon, description='None', symbol='Var',
unit='−−', fill=9.96921e+36, filename='result.nc')

引数：

Var：気象変量として書き出す３次元配列変数．Var［時刻の次元，緯度の次元，経度の次元］でデータが並んでいること．

tim：気象変量の各要素が並ぶ時刻を示す時刻オブジェクトの１次元配列．Var の最初の次元の要素数と一致していなくてはならない．

lat：気象変量の各要素が並ぶ緯度を示す１次元配列．Var の２番目の次元の要素数と一致していなくてはならない．

lon：気象変量の各要素が並ぶ経度を示す１次元配列．Var の３番目の次元の要素数と一致していなくてはならない．

description：気象変量の名前等を description＝'名前等' として指定する．指定を省略した場合は' Variable' という名で出力される．

symbol：気象変量の記号を symbol＝ '記号' として指定する．指定を省略した場合は 'Var' が使用される．

unit：気象変量の記号を unit＝ '単位の記号' として指定する．指定を省略した場合は '−−' が使用される．

fill：無効値として使用する数値を fill＝数値で指定する．指定を省略した場合は9. 96921 e ＋36に設定される．

filename：出力される NetCDF ファイルのファイル名を filename ＝ 'ファイル名' として指定する．指定を省略した場合は 'result.nc' という名で出力される．

戻り値：戻り値はない．

使用例：　以下により，データ配信サーバーから北緯35～36度，東経135～136度の範囲における2008年～2012年の日最高気温データを取得し，MaxTemp.nc という名の NetCDF ファイルと

して出力する．無効値にはデフォルトが用いられる．

```
import AMD_Tools as AMD
import numpy as np
timedomain = ['2008−01−01', '2012−12−31']
lalodomain = [35.0, 36.0, 135.0, 136.0]
Tm, tim, lat, lon = AMD.GetData('TMP_max', 'Area4', timedomain, lalodomain)
AMD.PutNC_3D( Tm, tim, lat, lon, description='Maxmun air temperature', symbol='Tmax',
        unit='degC', filename='MaxTemp.nc')
```

６）PutCSV_TS（）

概要：時刻順に並ぶ配列を，行方向にデータが並ぶ CSV ファイルとして出力する関数.

書式：PutCSV_TS(Var, tim, header=None, filename='result.csv')

引数：

Var：CSV ファイルに書き出す１次元配列変数．Var の第０次元の要素数は，tim の要素数と一致していること.

tim：気象変量の各要素が並ぶ時刻を示す時刻オブジェクトの１次元配列．Var の第０次元の要素数と一致していなくてはならない.

header：CSV ファイルに行見出しを与える時に，header = '列見出し' として指定する．この際，CSV の書式に基づいて，文字列を与える．指定を省略した場合は見出しは付けられない.

filename：出力される CSV ファイルのファイル名を filename＝ 'ファイル名' として指定する．指定を省略した場合は 'result.csv' という名で出力される.

戻り値：戻り値はない.

使用例：以下により，データ配信サーバーから，北緯35度，東経135度の地点における2008年の日平均気温と降水量を取得し，日付，気温，降水量，の順に３列に並ぶ CSV ファイルとしてデフォルトのファイル名で出力する.

```
import AMD_Tools as AMD
import numpy as np
timedomain = ['2008−01−01', '2008−12−31']
lalodomain = [35.0, 35.0, 135.0, 135.0]
Ta3D, tim, lat, lon = AMD.GetData('TMP_mea', 'Area4', timedomain, lalodomain)
Ta = Ta3D[:,0,0]
Pr3D, tim, lat, lon = AMD.GetData('APCP', 'Area4', timedomain, lalodomain)
Pr = Pr3D[:,0,0]
tapr = np.array([Ta, Pr])
AMD.PutCSV_TS(tapr, tim, header='Date, Ta, Pr')
```

７）PutCSV_MT（）

概要：３次元の配列を，３次メッシュコードをキーとするテーブルの形式の CSV ファイルで出力する関数．第１次元（緯度），第２次元（経度）が同じ第０次元の内容を添え字の順に記号で区切って出力する．３次メッシュコードを属性に持つメッシュのポリゴンデータを GIS に

整備しておくと，GIS 上でこのファイルとポリゴンをリンクするにより，データの分布図を GIS 上で簡単に表示することができる．

書式：PutCSV_MT(Dat, lat, lon, addlalo=False, header=None, filename='result.csv', removenan=True, delimiter=',')

戻り値：なし

引数：

Dat：書き出すべき３次元配列変数.

lat：配列 Dat の各行が位置する緯度値が格納されている配列.Dat の第１次元の要素数と一致していなくてはならない.

lon：配列 Dat の各列が位置する経度値が格納されている配列.Dat の第２次元の要素数と一致していなくてはならない.

addlalo：これを True にすると，３次メッシュ中心点の緯度と経度が出力ファイルの第２フィールドと第３フィールド追加挿入される．デフォルトは False であり挿入されない.

header：一行目に見出しやタイトルなど何か書き出すときはここに「header＝ '文字列'」として指定する.

filename：出力されるファイルの名前．デフォルト値は 'result.csv' .

delimiter：フィールドの区切り文字．デフォルト値は ',' ．すなわち，CSV ファイルとなる.

removenan：無効値だけのレコードを削除するかを指定するキーワード．値は 'True'.しないときは 'removenan=False'とする.

使用例：以下により，データ配信サーバーから，新潟県における2013年８月１日～10の日平均気温を取得し，メッシュ別に「メッシュコード，緯度，経度，８月１日の気温，８月２日の気温，…，８月10日の気温」の順で並ぶテキストファイルを作成することができる.

```
import AMD_Tools as AMD
import numpy as np
timedomain = [ '2013-08-01', '2013-08-10' ]
lalodomain = [36.7, 38.6, 137.6, 140.0]
dat, tim, lat, lon = AMD.GetData('TMP_mea','Area2',timedomain, lalodomain)
pref,    lat, lon = AMD.GetGeoData('pref_1500', 'Area2', lalodomain)
dat = dat * pref
hd = 'MeshID, latitude, longitude'
for t in range(len(tim)):
  hd = hd + ',' + str(tim[t])
AMD.PutCSV_MT(dat, lat, lon, addlalo=True, header=hd )
```

８）PutCSV_Map（）

概要：２次元の配列変数を，緯度を行方向に，経度を列方向に配置する CSV ファイルとして出力する関数．第１行には経度の数値，第１列には緯度の数値が見出しとして出力される．この際，緯度は北が上になるように出力する．また，無効値には文字列 nan が代入される.

書式：PutCSV_Map(Var, lat, lon, filename='result.csv')

引数：

Var：CSV ファイルに書き出す１次元配列変数．Var の最初の次元の要素数は lat の要素数と一

致していること.

lat：配列の各要素が並ぶ緯度を示す１次元配列．Var の最初の次元の要素数と一致していなくてはならない.

lon：気象変量の各要素が並ぶ経度を示す１次元配列．Var の２番目の次元の要素数と一致していなくてはならない.

filename：出力される CSV ファイルのファイル名を filename ＝ 'ファイル名' として指定する．指定を省略した場合は 'result.csv' という名で出力される.

戻り値：戻り値はない.

使用例：以下により，データ配信サーバーから，2008年１月１日の佐渡島周辺おける日平均気温の分布を取得し，CSV ファイルとしてデフォルトのファイル名で出力する（図26）.

```
import AMD_Tools as AMD
import numpy as np
timedomain = ['2008−01−01', '2008−01−01']
lalodomain = [37.7, 38.4, 138.2, 138.6]
Ta3D, tim, lat, lon = AMD.GetData('TMP_mea', 'Area2', timedomain, lalodomain)
Ta2D = Ta3D[0,:,:]
AMD.PutCSV_Map(Ta2D, lat, lon)
```

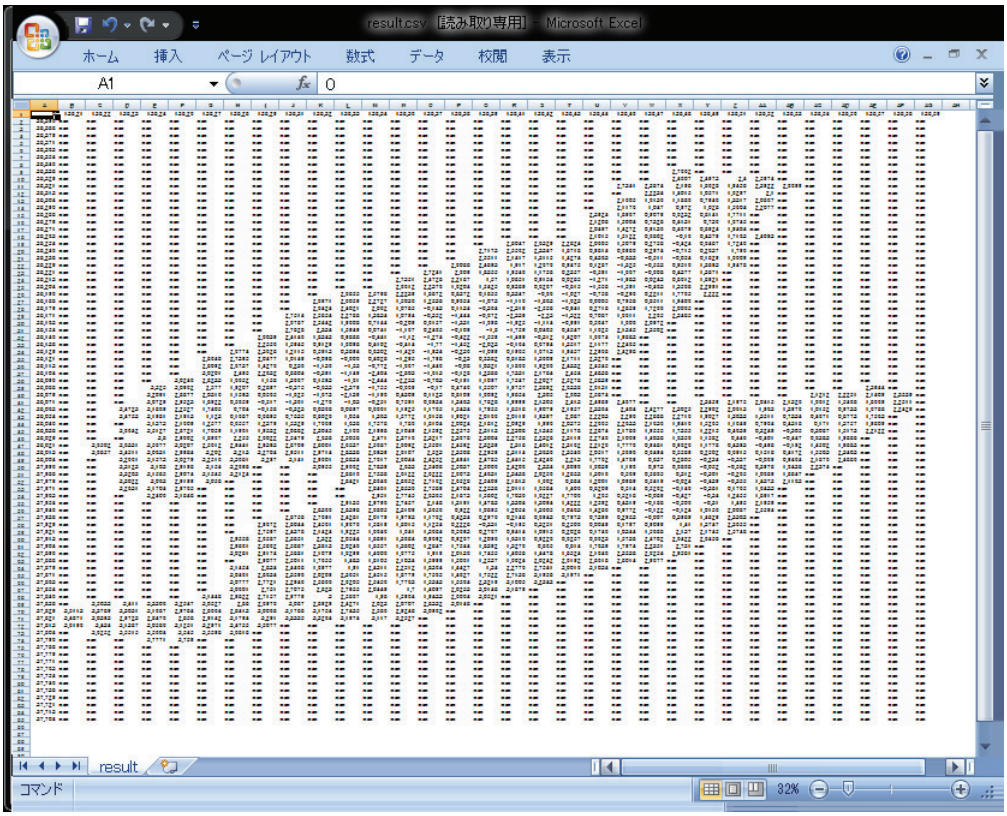

図26.　関数 PutCSV_Map () で出力した CSV ファイルを Excel で表示した画面

佐渡島周辺の平均気温分布が取得されている.

BOX９　NetCDF ファイルについて

NetCDF ファイルは，気象データなどの地球科学的なデータを納めることを目的として Unidata というプロジェクトが策定したファイル形式です．とても複雑なのでこのファイルの読み書きをするには，言語毎に作られているツールを利用するのが普通です．データの各要素が持つ時刻，緯度，経度，高度をはじめとして，データの名前，単位，無効値，数値型，作者などまでもが規約に基づいて整然と格納されているので，これを前提にプログラムを組むと，データのサイズや無効値を予め調べてプログラムの定義文に書き込んだり，月の大小や閏年によるややこしい条件分岐をプログラミングしたりする必要がなくなり，プログラムをとてもシンプルにすることができます．以上の理由から，この手引きでは，農業気象データの処理結果を NetCDF ファイルの形で保存しています．

９）accumulation_of_effective_temperature（）

概要：引数に気温の三次元配列を取り，これをもとに気温有効積算温度を計算する関数．引数として入力する配列と，戻り値として出力される配列のサイズは同一である．戻り値の配列における時間方向に n 番目の（２次元）要素には，１日目から n 日目までの積算値が格納されている．例えば，ret[6, :, :] は，期間の７日目における有効積算温度分布を意味する．

書式：ret = accumulation_of_effective_temperature(Var, To=0.0)

引数：

Var：有効積算温度を計算するもととなる気温の時空間分布データの配列．

To：基準温度を To= ’温度（℃）’ として指定する．指定を省略した場合は０℃が与えられる．

戻り値：

ret：有効積算温度を計算するもととなる気温と同じサイズの３次元配列．戻り値の配列における時間方向に n 番目の（２次元）要素には，１日目から n 日目までの有効積算温度が格納されている．

使用例：以下により，北緯35～36度，東経135～136度の範囲における2013年８月１日を起日とし，基準温度を５℃とする有効積算気温の一ヶ月間の推移を計算し，NetCDF 形式のファイルとして出力する．この結果は，IDV で可視化することができる．

```
import AMD_Tools as AMD
import numpy as np
timedomain = [’2013−08−01’, ’2013−08−31’]
lalodomain = [35.0, 36.0, 135.0, 136.0]
Ta, tim, lat, lon = AMD.GetData(’TMP_mea’, ’Area4’, timedomain, lalodomain)
Tacc = AMD.accumulation_of_effective_temperature( Ta, To=5.0 )
AMD.PutNC_3D( Tacc, tim, lat, lon, description=’Effective Degree Day Temperature’,
symbol=’DDT’,  unit=’degC day’, filename=’DDT_Aug.nc’)
```

Ⅴ　IDV を用いたデータの可視化

第Ⅳ章で見たとおり，プログラミング言語 Python は品質の高いグラフィクスを作成することができますが，そのためには難解な書式設定の文をたくさん書かなければなりません．定番の図として繰り返し使用するものはそれでもいいのですが，数枚しか作成しない図の作成にプログラ

ミングの労力を費やすのは効率的ではありません．このような時には，計算結果を数値としてファイルに保存し，専用のソフトウエアでこれをインタラクティブに図化した方が効率的です．

この章では，IDV というデータ可視化ソフトウエアを使って，メッシュ農業気象データや Python によるその処理結果を図化する方法を解説します．Ⅶ-2に，IDV のインストールと設定の手順が説明されています．

１　IDV によるメッシュ農業気象データの可視化

１）IDV の起動と配信サーバーへの接続

IDV は，インストールによりデスクトップに作成される IDV のアイコンをダブルクリックして起動します．３つ開くウインドウの中の，タイトルバーに Dashboard と表示されているウインドウを一番上にします．このウインドウは，タブがついた４枚のページを持っています．

この中の Data Choosers のページを開き，左端のペインに表示されているリストから Catalogs を選択します．次に，右ペインの上部にあるテキストボックスに，次の文字列を入力します．「http://mesh.dc.affrc.go.jp/opendap/catalog.xml」すると，その下に Area１から Area６までのフォルダがアイコン表示されます．これで，メッシュ農業気象データ配信サーバーへの接続が完了しました（図27）．

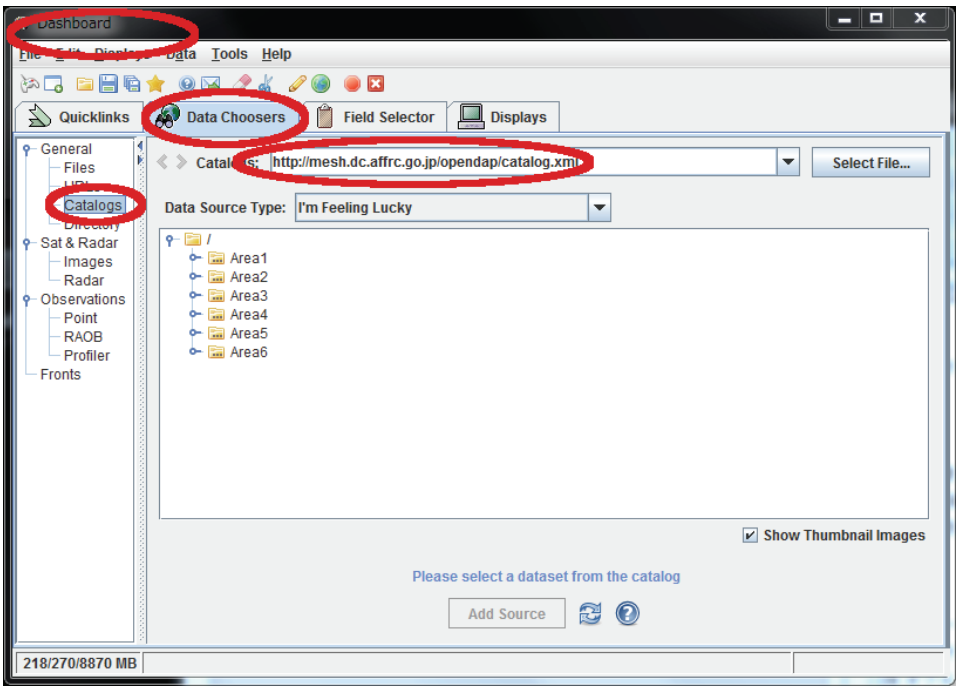

図27．メッシュ農業気象データ配信サーバーへの接続が完了したところ

データセットのフォルダがアイコンで表示されている．

２）データの選択

2013年８月の茨城県周辺の日平均気温を例に，分布図を作成します．茨城県は，Area３に含まれるので，アイコン表示されているフォルダを順に展開して，Area3/2013/AMD_Area3 TMPmea.nc を表示させ，これをハイライトしたうえでさらに右ペイン下部の［Add Source］ボタンを押します．すると，選択されているタブが Data Choosers から Field Selector に移ります．

このタブのページは，左，右上，右下の３つのペインで構成され，右下のペインには，さらに，３つのタブが設けられ，Times が選択されています（図28）．

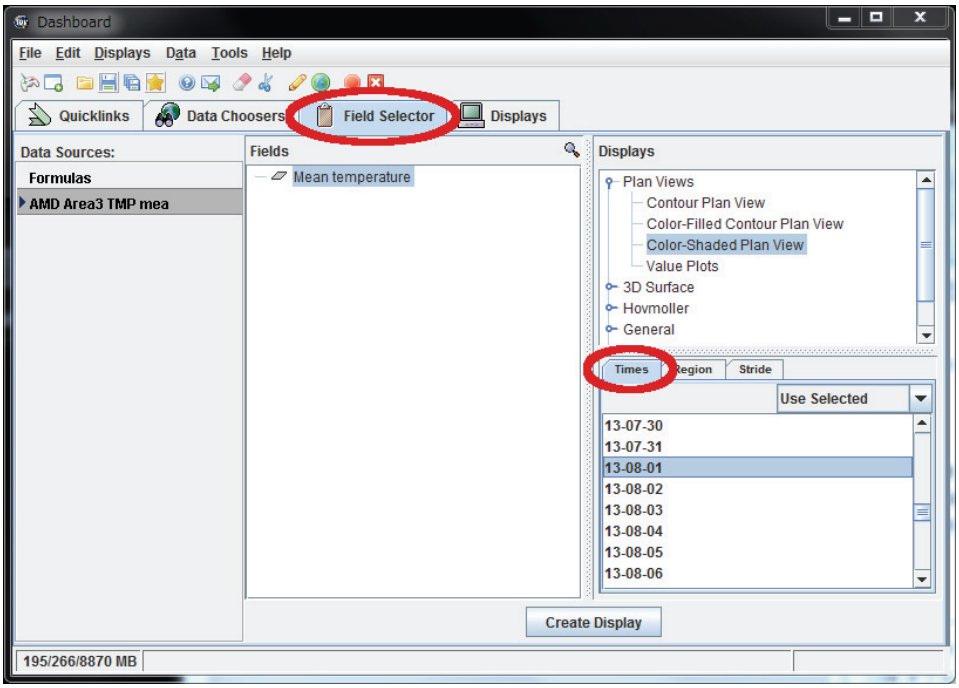

図28. 可視化する期間の設定法

Dashboard ウインドウ，Field Selector ページの右下ペインの Times で設定する．

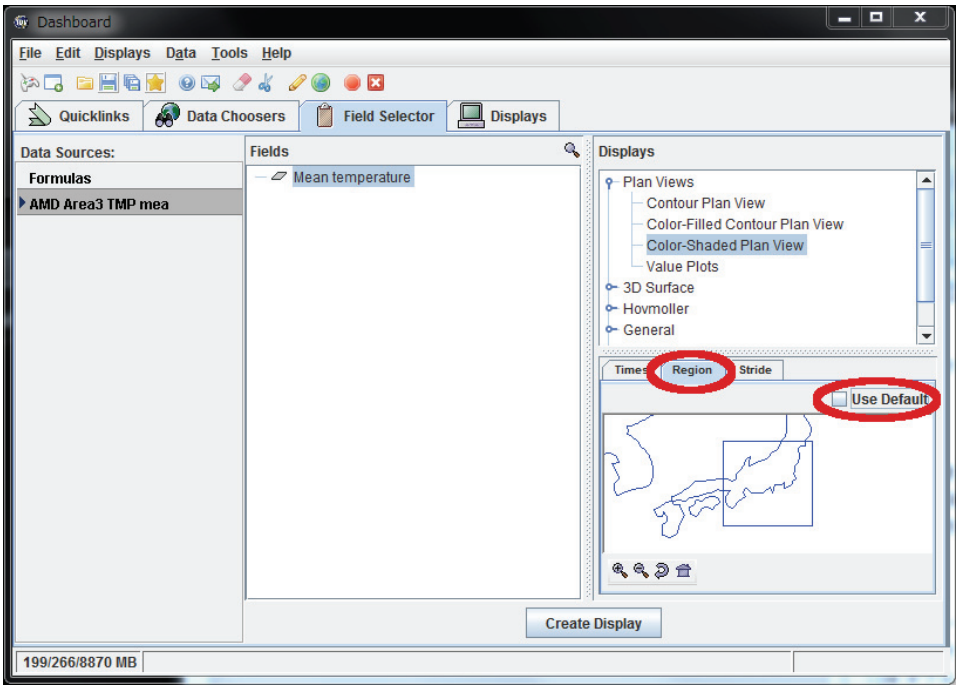

図29. 可視化する地理的領域の設定法

Dashboard ウインドウ，Field Selector ページの右下ペインの Region の地図上にマウスで矩形を設定する．

データセットを選択したら，次に期間を設定します．タブの右下にあるプルダウンボタンを押して，表示を「Use Default」から「Use Selected」に変更します．その上でスクロールバーを操作して日付リストを動かし，「13－08－01」から「13－08－31」の範囲を選択します．

期間を設定したら，次に領域を設定します．これは Region タブで行います．このタブをクリックすると，Area 3 がカバーする領域が四角で示されます(図29)．地図の右上にある Use Default

ボックスのチェックを外すと，背景が灰色から白に変わり，マウスにより可視化範囲を変更することができるようになるので，茨城県のあたりに可視化範囲を設定します．この際，範囲を大きく取ると，データ取得に長い時間を必要とするので，注意してください．

３）図の種類の設定

データの時空間範囲が定まったら，図の種類（等高線図にするか，色分け図にするか等）を指定します．これは，右上のペインで指定します．ここでは，「Color-Shaded Plan View」をハイライトしてください．その上で，ウインドウの一番下にある［Create Display］ボタンを押します（図30上段）．すると，データが読み込まれ分布図が表示されます（図30下段）．分布図が表示されているウインドウの左端にあるツールボタンで，画面を拡大，縮小，移動することができます．

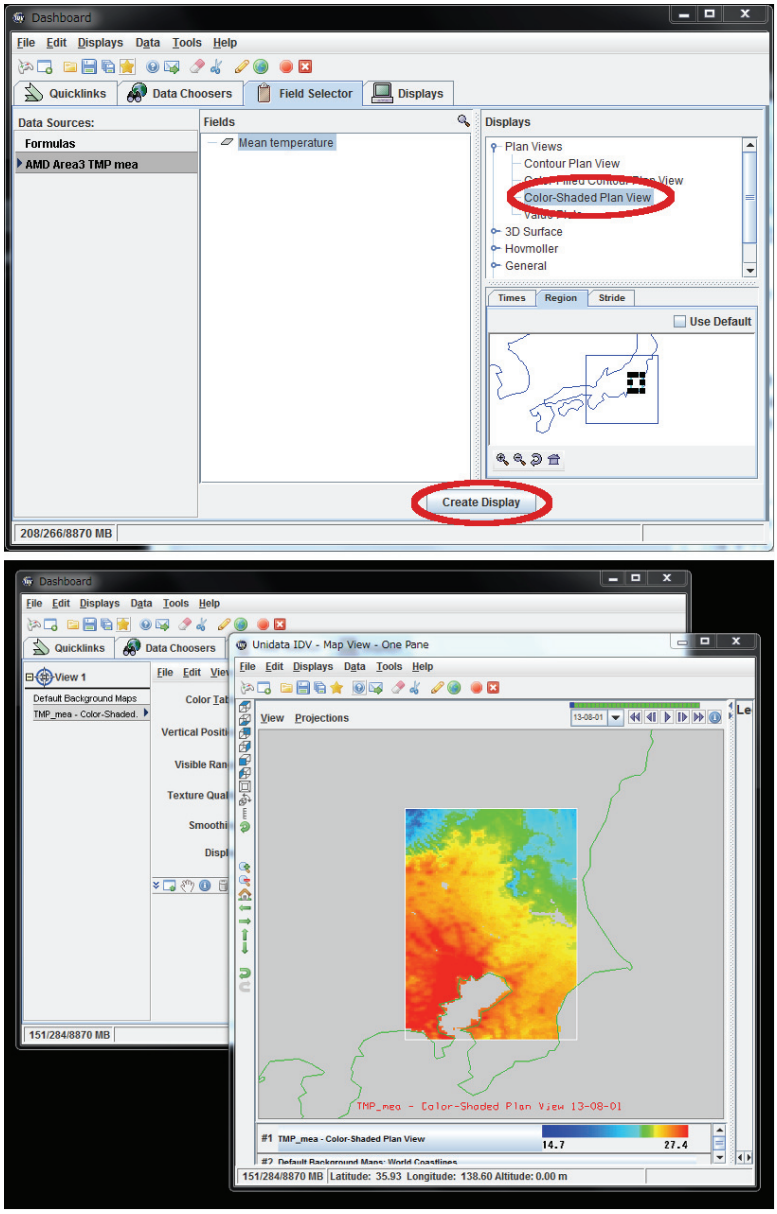

図30．作図方法の設定と可視化の実行

Dashboard ウインドウ，Field Selector ページの右上ペインの一覧から作図方法を選択し［Create Display］ボタンを押す（上段）と，2013年８月１日の平均気温分布図が表示される（下段）．

（1）色範囲の変更

Dashboard ウインドウ，Displays ページのカラーサンプルの左側にある［default］ボタンを押すと，メニューがプルダウンします．ここの中の Change Range を選択すると，表示する色の上限と下限を設定することができます（図31）．

（2）可視範囲の設定

指定した値の範囲だけを可視化し，範囲外の領域には彩色しない表示をすることができます．Dashboard ウインドウ，Displays ページの，Visible　Range：の右側にあるチェックボックスをクリックしてからその隣にある［Change］ボタンを押すと，入力するウインドウが開くので，ここで範囲を設定します（図32）．

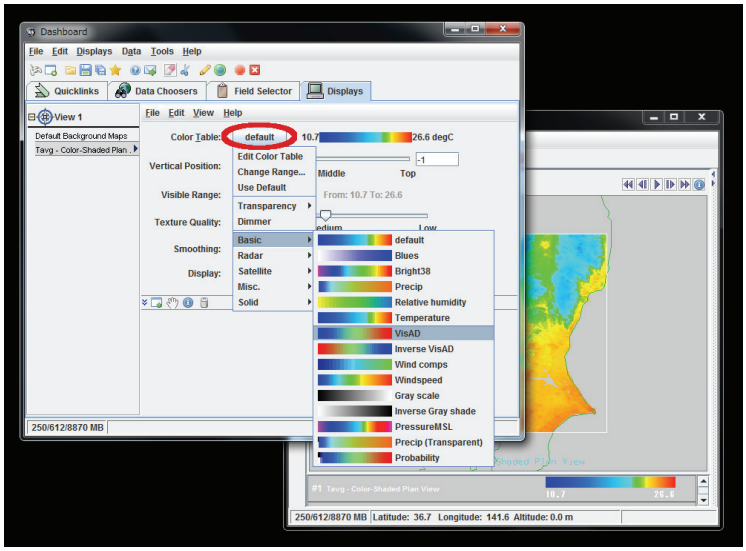

図31．色範囲と色の並びの変更

Dashboard ウインドウ，Displays ページの，「Color Table」の横のボタンから設定する．

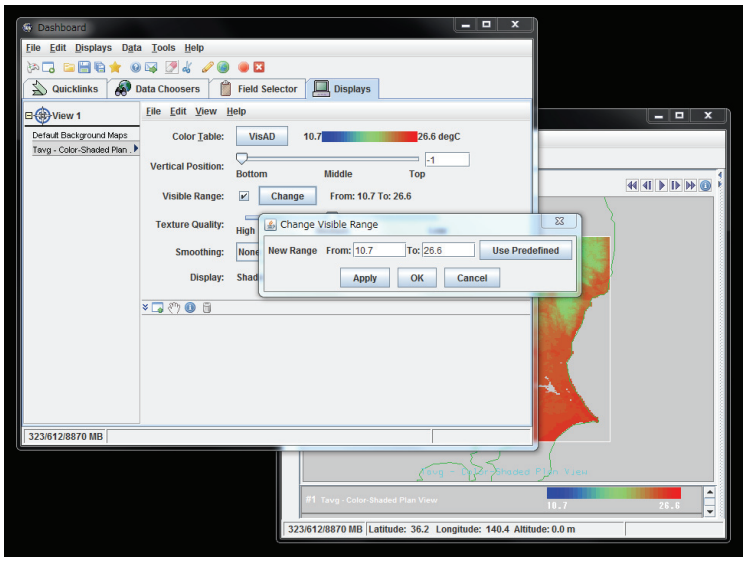

図32．可視する値の範囲の設定

Dashboard ウインドウ，Displays ページの，Visible Range から設定する．

（3）カラーテーブル（色の並び）の変更

Dashboard ウインドウ，Displays ページのカラーサンプルの左側にある［default］ボタンを押してメニューをプルダウンし，この中の Basic から Solid にかけて項目にマウスカーソルを合わせると，様々なカラーテーブルが表示され，それらに変更することができます（図31）.

（4）描画日の選択・アニメーション

View に表示されている分布図の日付は，描画エリア右上のプルダウンボックスに表示されています．ここを切り替えると，異なる日付の分布図を表示させることができます．また，その右にある三角印をクリックすると，取得した期間の分布図を連続して表示します.

４）任意メッシュの日々変化の可視化

IDV は，分布図上の任意のメッシュにおける気象データの日々の変化を折れ線グラフとして表示することができます．分布図が表示されている View ウインドウから離れて，先ほどの Dashboard ウィンドウに戻り，Field Selectoer ページを開きます．そして，右上のペインに示される図の種類をスクロールして Data　Probe/TimeSeries をハイライトし，ウインドウの一番下にある［Create　Display］ボタンを押します．すると，表示されているページが Display に切り替わり，日変化グラフが表示されます（図33左）.

Dashboard ウィンドウに折れ線グラフが表示されるのと同時に，View ウインドウの中央には，マーカーが周囲と色の違う四角で表示されます（図33右）．折れ線グラフは，この地点における気象要素の日々変化を示しています．このマーカーはマウスで任意の場所に移動することができます.

マーカーが見にくいときは：マーカーの色が周囲の色に紛れて見にくいときは，Dashboard ウインドウの Displays タブのサブメニューから Edit＞Probe Color を選択して色を選びます. なお，緯度経度を指定して正確にプローブの位置を定めるときには，Displays タブの右下にある緯度経度ボックスに数値を直接入力します.

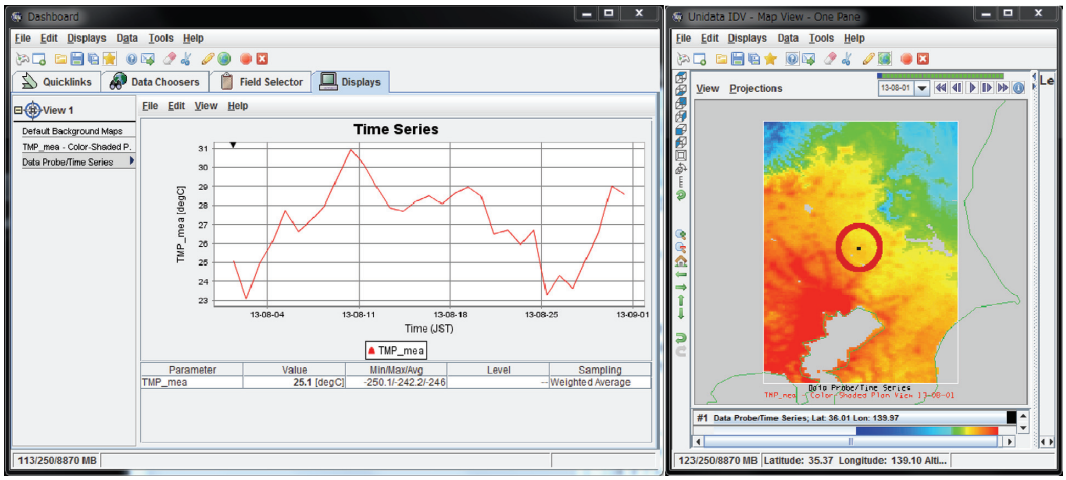

図33．任意地点における値の日変化の表示

Dashboard ウィンドウの Field Selector ページから Data Probe／TimeSeries を指定すると，日変化グラフが得られる（左図）．View ウインドウのカーソルを移動するとグラフ化する地点を変えることができる（右図）.

5）市町村界のオーバーレイ

IDV は，GIS でよく使われる ShapeFile と呼ばれるフォーマットの地理情報を表示することができます．この機能を利用すると，気象要素の分布図に市町村界を重ね書きすることができます．

Dashboard ウインドウの Data Choosers ページを開き，左端のペインのハイライトを Files にします．すると右ペインはファイルブラウザになるので，「参照：」の所に並ぶボタンを適宜利用して，行政界の ShapFile を選択し［Add Source］ボタンを押します（図34）．Field Selector のページに遷移するので［Create Display］ボタンを押して描画させます（図35）．

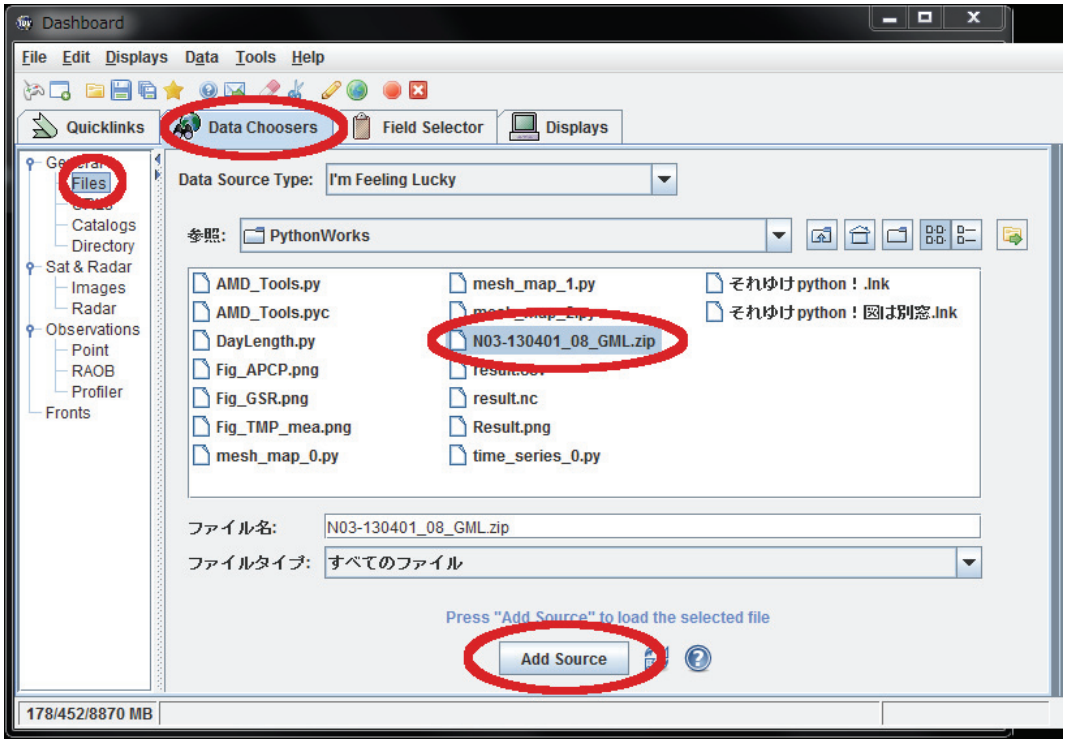

図34. 行政界の表示法

Dashboard ウィンドウ Data Choosers ページの左ペインでデータ種別を「Files」とし，ファイルブラウザから行政界の Shape ファイルを選択してデータソースに加える．

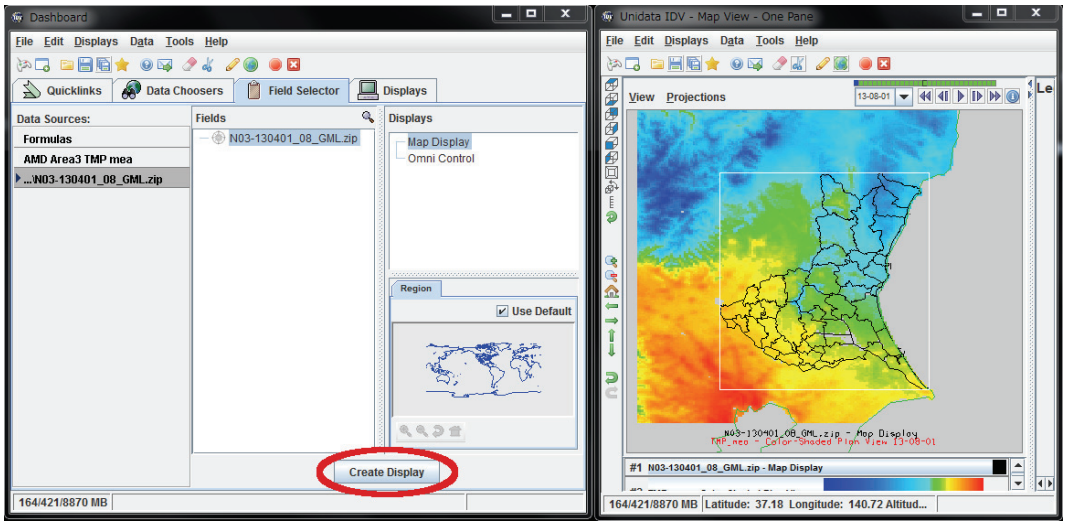

図35. 行政界の表示法（つづき）

遷移した Field Selector ページの下部にある［Create Display］ボタンを押すと行政界がオーバーレイされる．

ShapeFile は本来は３つ以上のファイルで構成されるため，ZIP ファイルに圧縮されて流通することがしばしばあります．IDV は ZIP ファイルをそのまま指定することができます．

なお，日本の行政界の ShapFile は，国土地理院の国土数値情報ダウンロードサービス（http://nlftp.mlit.go.jp/ksj/gml/gml_datalist.html）より入手することができます．

線が見にくいときは：境界線の色が見にくいときは，Dashboard ウインドウの Displays ページのサブメニューから Edit＞Line Color を選択して色を選びます．

６）表示の画像ファイル化

IDV で可視化した画像は，画像ファイルとして保存することができます．分布図が表示されている View ウインドウのサブメニュー，View から,Capture＞Image と選択します（図36）．画質やファイル形式，凡例を付ける付けない，など幾つかの選択肢があるので，目的に応じ設定して保存します．

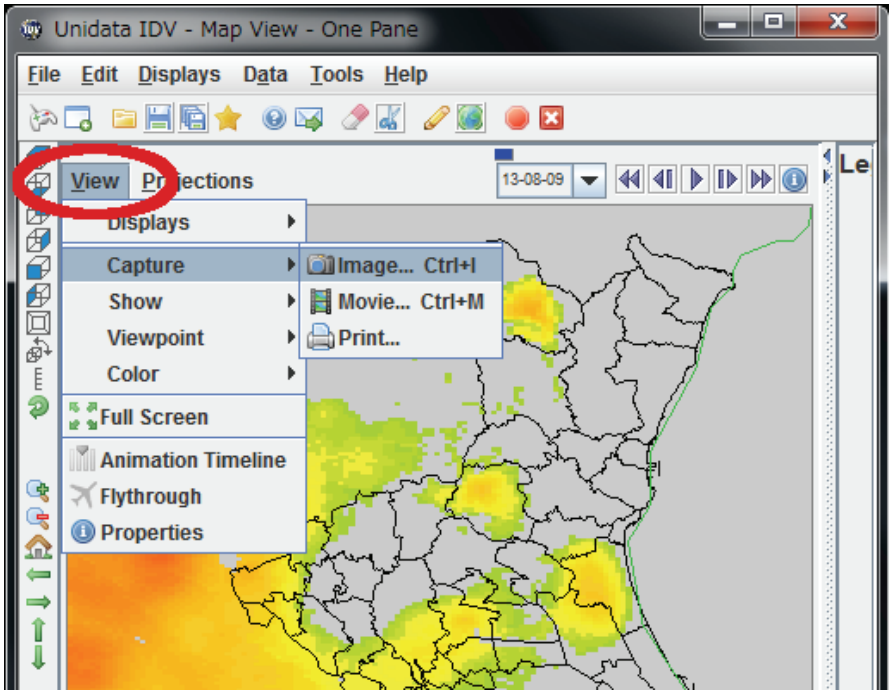

図36. 可視化した画像の保存

View ウインドウのサブメニュー View から，Capture>Image...を選択して保存する．

７）表示設定の保存

IDV は可視化に際して施した様々な表示設定を保存することができます．

Dashboard ウインドウのメインメニューから Displays＞Favorite Bundles＞Save As Favorite ...を選択し，設定ウインドウを開きます（図37上段）．ここの Name：に「Ibaraki」等適当な名前を入力します．他は図37（下段）のとおりとします．設定がすべて終了したら［了解］ボタンを押します．

プログラムの終了はメインメニューから File＞Exit とします．ツールバーの赤四角でも構いません．さて，先ほどの画面はうまく戻るでしょうか．IDV のデスクトップアイコンをダブルクリックして再び起動し，Dashboard ウインドウのメインメニューから Displays＞Favorite Bundles＞Genaral を選択してください．そのさらに右に Ibaraki という選択肢が表示されるのでこれを選択します．

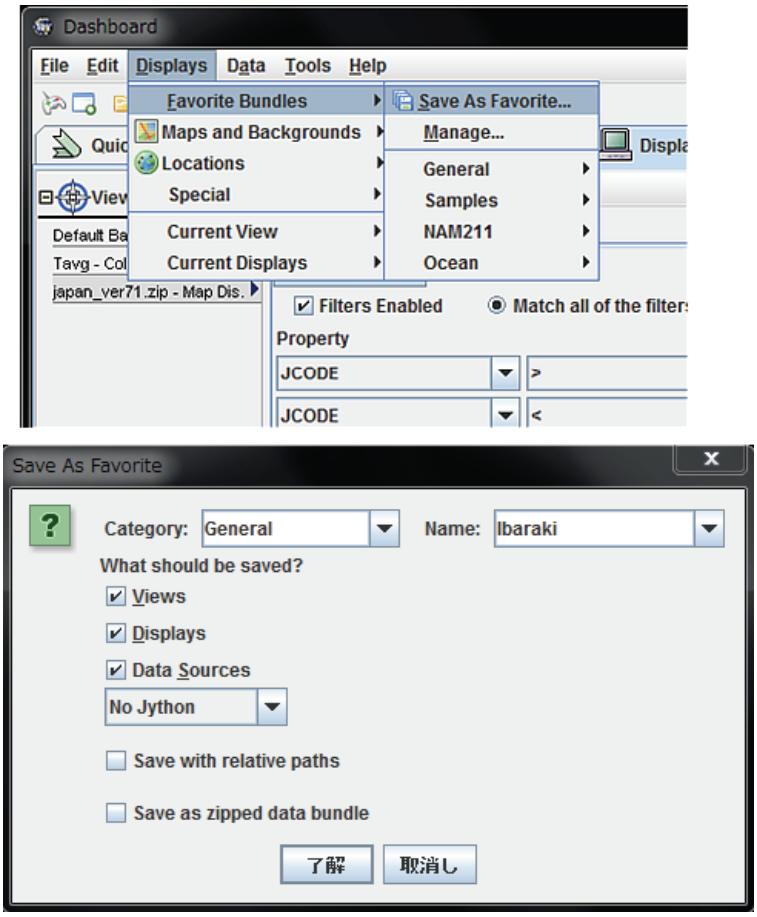

図37．表示に関する諸設定の保存

Dashboard ウインドウのメインメニューから Display を選択する（上段）．ポップアップする設定ウインドウ（下段）で設定に名称を与えて［了解］ボタンを押す．

２　IDV による NetCDF ファイルの可視化

Python モジュール AMD_Tools の関数，PutNC_Map（）や，PutNC_3D（）を使用すると，Python プログラムによる処理結果を簡単に NetCDF ファイルとして出力することができます（Ⅳ-5参照のこと）．IDV は，NetCDF ファイルのデータを様々に可視化できるので，Python プログラムの処理結果の確認やプレゼンテーションに使用することができます．

ここでは，Ⅳ-4で説明した Python プログラム「RiceDevelopment.py」で計算した，茨城県における2013年の水稲の発育指数（DVI）の時空間分布が保存されているファイル「DVI.nc」を例に可視化手順を説明します．なお，「DVI.nc」は利用者 Wiki からダウンロードすることができます．

デスクトップにあるアイコンをダブルクリックして IDV を起動し，タイトルバーに Dashboard と表示されているウインドウを一番上にします．このウインドウには４つのタブが設けられています．その中の Data Choosers のページを開きます（図38）．まず，左端のペインに表示されているリストの中で「Files」が選択されていることを確認します．次に，右側のペイン上部にある「参照：」に並ぶアイコンを適宜利用して DVI.nc を探して選択し，右ペインの一番下にある［Add Source］ボタンを押します．

すると，選択されているタブが Data Choosers から Field Selector に移ります．次に，右上のペインで作図の方法を決めます．ここでは，「Color-Shaded Plan View」をハイライトしてくだ

さい（図39）．その上で，ウインドウの一番下にある［Create　Display］ボタンを押します．すると，データが読み込みこまれ分布図が表示されます．

ここから先の操作は，Ｖ-1と全く同じです．カラースケールの変更や行政界のオーバーレイなど適宜設定してください（図40）．

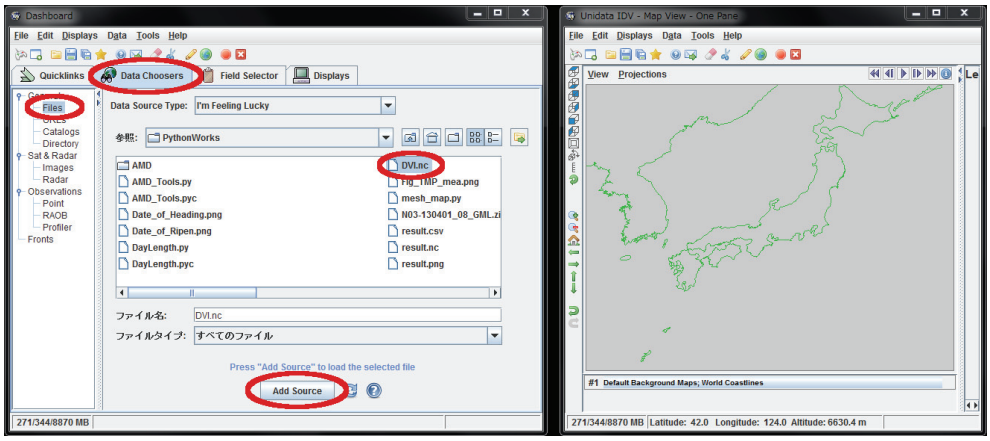

図38.　NetCDF ファイルの可視化

Dashboard ウィンドウ Data　Choosers ページの左ペインでデータ種別を「Files」とし，ファイルブラウザから可視化する NetCDF ファイルを選択してデータソースに加える．

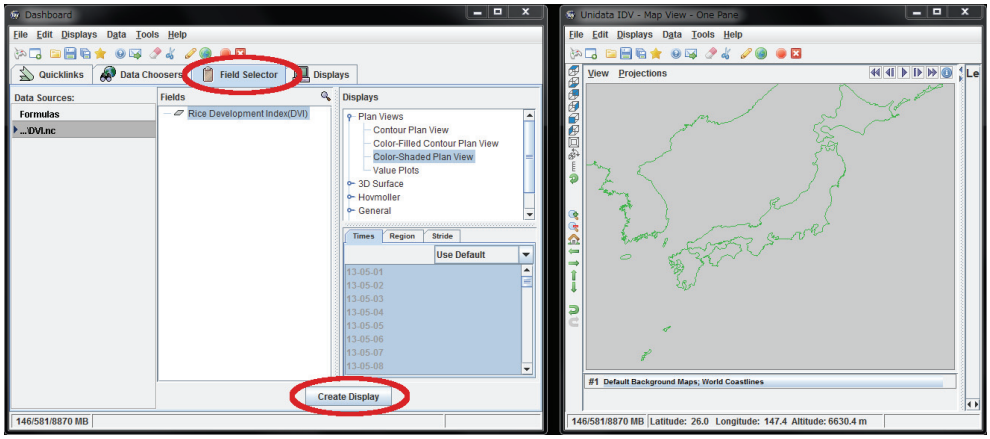

図39.　NetCDF ファイルの可視化（つづき）

ページが Field Selector に遷移するので，作図方法や可視化領域を設定し，表示させる．

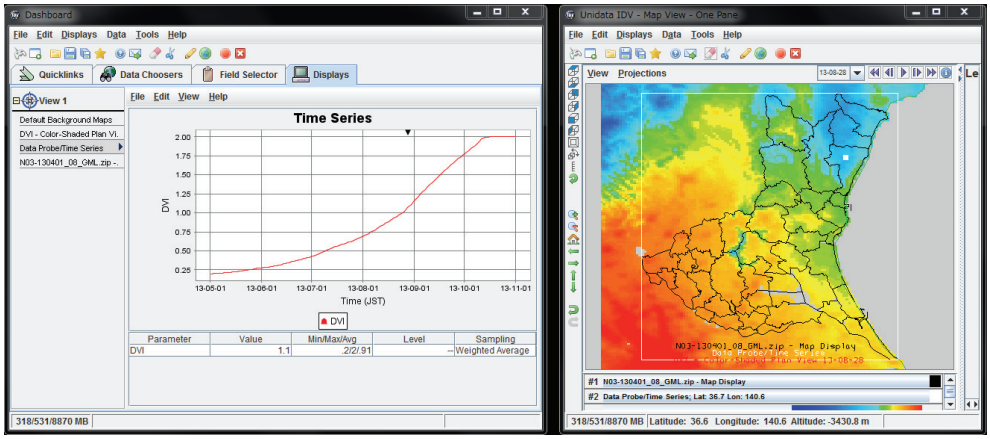

図40.　可視化された発育指数（DVI）の分布

View ウインドウ（右側）上のカーソル地点における DVI の日々変化が Dashboard ウインドウ（左側）の Display ページに折れ線グラフで表示される．

Ⅵ　GMT を用いた高品質な図の作成

GMT はハワイ大学の海洋地球科学技術教室が開発する，地図やメッシュデータ，地点観測データ，さらに，一般的なプロットグラフまでもたいへん綺麗に描画するフリーソフトウエアです．Ⅶ-3に，GMT のインストールと設定の手順が説明されています．

1　GMT での作画方法

IDV がマウスを操作しながらインタラクティブに図を作ってゆくのと対照的に，GMT は作図のためのコマンドをテキストファイルに書き込み，それを手順書（スクリプト）として実行させて図を作ります．どちらかというと，プログラムを書いて実行する Python に似ています．ですから，繰り返し使う定番の図や学術論文に投稿する図をしっかり描くのに向きます．

例えば，図41に示す北関東の地図画像は，海岸線を描画するコマンド pscoast を用いて次のようにして作成します．

```
pscoast −G220/220/200 −R139.0/141.0/35.5/37.5 −Jm8 −Ba0.5f0.5/a0.5f0.5:.”
Sample Figure”: esWN  −P −Dh −X2.5 −Y6.0 −K > figure.eps
```

（実際には１行で書きます）

コマンドの後ろに延々続いているのは，全てオプションです．オプションは，ハイフンで始まり，何を設定するのかを示すアルファベットと，どのようにするのかを示すパラメータからなります．上の例で現れるオプションは，順に，以下を意味します．

−G220/220/200：陸地を赤220／緑220／青200の色で塗る．
−R139.0/141.0/35.5/37.5：作図範囲を東経139. 0～141. 0／北緯35. 5～37. 5とする．
−Jm8：投影法をメルカトル図法とし縮尺を８とする．
−Ba0.5f0.5/a0.5f0.5:.” Sample Figure”:esWN：0. 5度刻みで地図に枠を付け，緯度／経度の数値を西と北にだけ印字し，上に「Sample Figure」を印字する．
−P：紙を縦置きとする．
−Dh：海岸線を精細に描く．
−X2.5：紙の左端から2. 5cm 空けて地図を描く．
−Y6.0：紙の下端から６cm 空けて地図を描く．
−K：このあとまだ作図が予定されている．

なお，「>figure.eps」はコマンドオプションではありませんが，ファイル figure.eps を新規に作成して図を保存することを意味します．

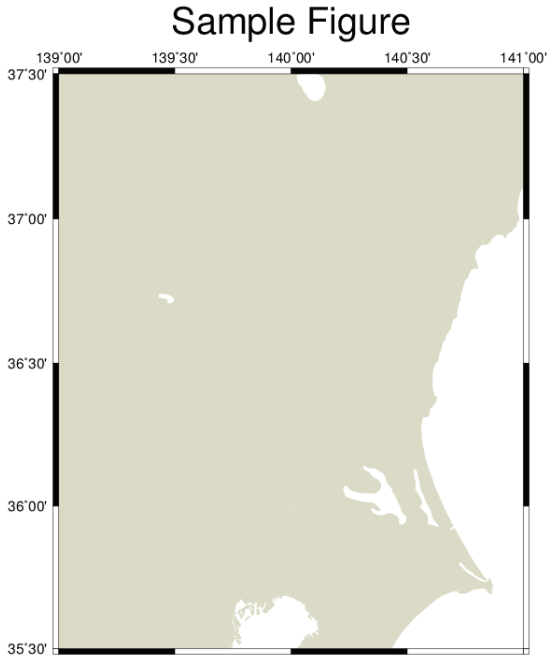

図41．GMT による地図の作成

コマンドの一つ一つにとても沢山のオプションがあり，しかもその書き方が一見しただけではよく分かりません．GMT は使い始めにちょっと苦労するソフトウエアです．けれど，幸いインターネット上には GMT に関するページが沢山あるので，コマンドやオプションを暗記しておく必要は無く，例えば海岸線を描くときにどのようなオプ

ションを付ければいいか？といったことは検索サイトで「GMT　海岸線」などと検索すればすぐに答えが得られます.

図42に示す気温分布図は，図41の地図の上に，さらに，気温やカラースケール，等値線などを重ね書きして作成されています．これを実行するスクリプト「Draw.sh」を BOX10に示します．このスクリプトを実行するには，Cygwin ターミナルを開いて，そこに，「sh ./Draw.sh＜エンター＞」と打ち込みます.

Draw.sh や県境データ，気温データなど，口絵を作図するのに必要なファイル一式が利用者 Wiki からダウンロードできるので利用してください.

２　気温分布図の作成

BOX10を見て分かるとおり，この図は７つのコマンドによって作図されています．使用されているコマンドとそのオプションについて以下，順に概略を説明します.

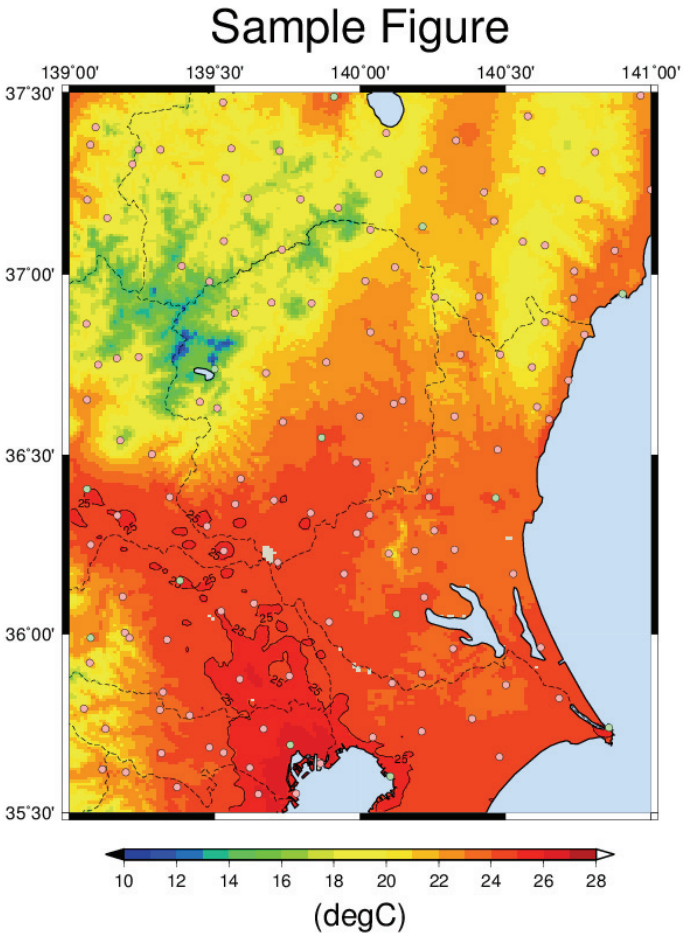

図42.　GMT で作成した2012年９月の平均気温分布図

BOX10　GMT で2013年9月の平均気温分布図を作成するスクリプト

（誌面の都合上，折り返している行がありますが，実際は１行で記述します.）

```
1  #!/bin/bash
2
3  #北関東の陸地を描きます.
4  pscoast −G220/220/200 −R139.0/141.0/35.5/37.5 −Jm8 −Ba0.5f0.5/a0.5f0.5:." Sample Figure":esWN −
   P −Dh −X2.5 −Y6.0 −K > figure.eps
5
6  #分布図を重ね書きします.
7  grdimage ../../PythonWorks/Ta_2012Sep.nc −CColPale.cpt −R −J −Sn −Q −O −K >> figure.eps
8
9  #カラーチャートを描きます.
10 psscale −D8.0/−1.0/13.0/0.3h −CColPale.cpt −B2.0:"(degC)": −E0.5 −I0 −O −K >> figure.eps
11
12 #25度 C の等値線を引きます.
13 grdcontour ../../PythonWorks/Ta_2012Sep −CContLeve.txt −G5c −W0 −R −J −Sn −Q −O −K >>
   figure.eps
14
15 #水域を上書きします.
16 pscoast −R −J −W0.5p/0/0/0 −S200/225/255 −Dh −O −K >> figure.eps
17
18 #アメダス/気象台の位置を重ね書きします.
19 psxy PointList.txt −CPointList.cpt −R −J −W0.1p −Sc0.2 −O −K >> figure.eps
20
21 #県境を重ね書きします.
22 psxy ../common/PrefecBound.txt −W0.5pta/0/0/0 −R −J −M −O >> figure.eps
```

```
grdimage Ta_2012Sep.nc -CColPale.cpt -R -J -Sn -Q -O -K >> figure.eps
```

この文は，データファイル Ta_2012Sep.nc から気温分布を取り出して先の図に重ね書きすることを指示しています（図43）．コマンド grdimage は，NetCDF ファイルのメッシュデータを地図上に表示するためのコマンドです．付加されているオプションの意味は次の通りです．
（第１引数）：データは Ta_2012Sep.nc である．
-CColPale.cpt：ファイル ColPale.cpt で定義されるカラーテーブルを使用する．
-R：緯度経度範囲はさっき決めたとおり（R オプションの後ろに何も指定しない）．
-J：図法はさっき決めたとおり　（J オプションの後ろに何も指定しない），
-Sn：作図に際しメッシュを間引かない．
-Q：データがないところは色をつけない．
-O：一つ前に書いた地図に重ね書きをする．
-K：このあとまだ作図が予定されている．
なお，「>>figure.eps」はコマンドオプションではありませんが，コマンドが作成した図をファイル figure.eps に追加保存することを意味します．

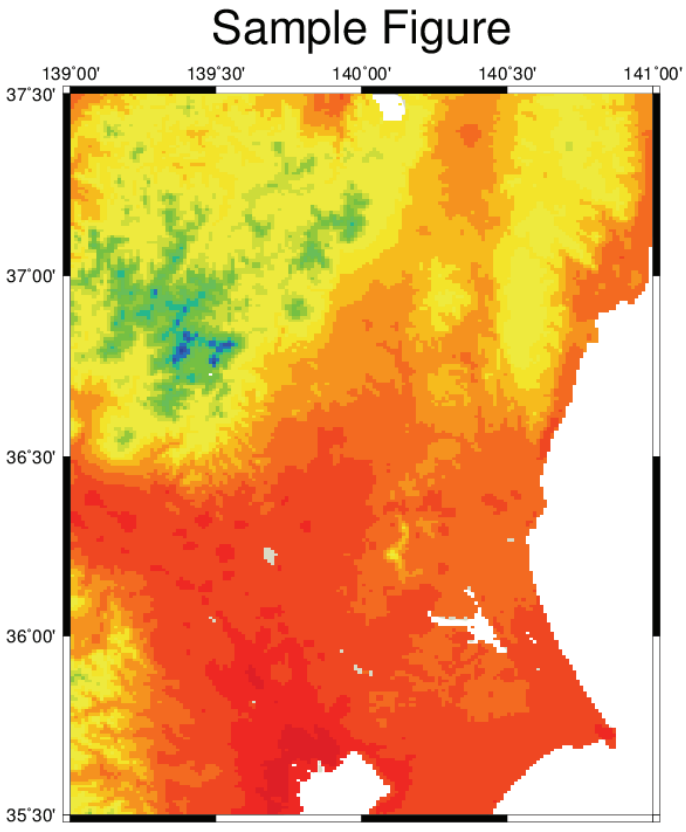

図43. 平均気温分布図の作成過程（1）
grdimage コマンドでデータファイルから分布図を作成する．

```
psscale -D8.0/-1.0/13.0/0.3h -CColPale.cpt -B2.0:"(degC)": -E0.5 -I0 -O -K >> figure.eps
```

この文は，図の下にカラースケールを描くことを指示しています（図44）．psscale はカラースケールを描画させるコマンドです．オプションの意味は次の通りです．

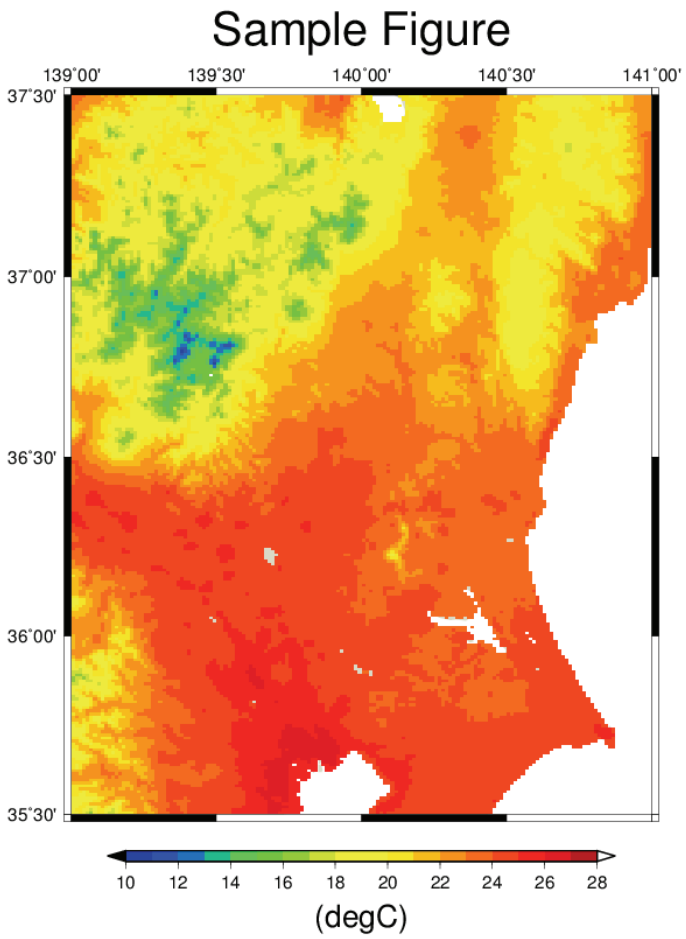

図44.　平均気温分布図の作成過程（2）

psscale コマンドでカラースケールバーを描画する.

−D8.0/−1.0/13.0/0.3h：枠の右８cm／枠の下１cm／長さ13cm／太さ３mm で横向きに作図する.
−CColPale.cpt：ファイル ColPale.cpt で定義されるカラーテーブルを使用する.
−B2.0:"(degC)":：目盛は２間隔とし，文字列「(degC)」を付加する.
−E0.5：スケールの両側に三角付レンジアウトの部分を0. 5cm の長さで設ける.
−I0：立体的には描かない.
−O：一つ前に書いた地図に重ね書きをする.
−K：このあとまだ作図が予定されている.

grdcontour Ta_2012Sep.nc −CContLeve.txt −G5c −W0p −R −J −Sn −Q −O −K >> figure.eps

この文は，分布図に25度Ｃの等値線を引き加えることを指示しています（図45）. grdcontour は，NetCDF ファイルのメッシュデータから等値線図を作成するコマンドです．オプションの意味は次の通りです.
（第１引数）：データは Ta_2012Sep.nc である.
−CContLeve.txt：ファイル ContLeve.txt に定義された値について等値線を引く.
−G5c：長い等値線には５cm 間隔で値を印字する.
−W0p：太さの線を０ポイント（最も細く）引く.
−R：緯度経度範囲はさっき決めたとおり（Ｒオプションの後ろに何も指定しない）.

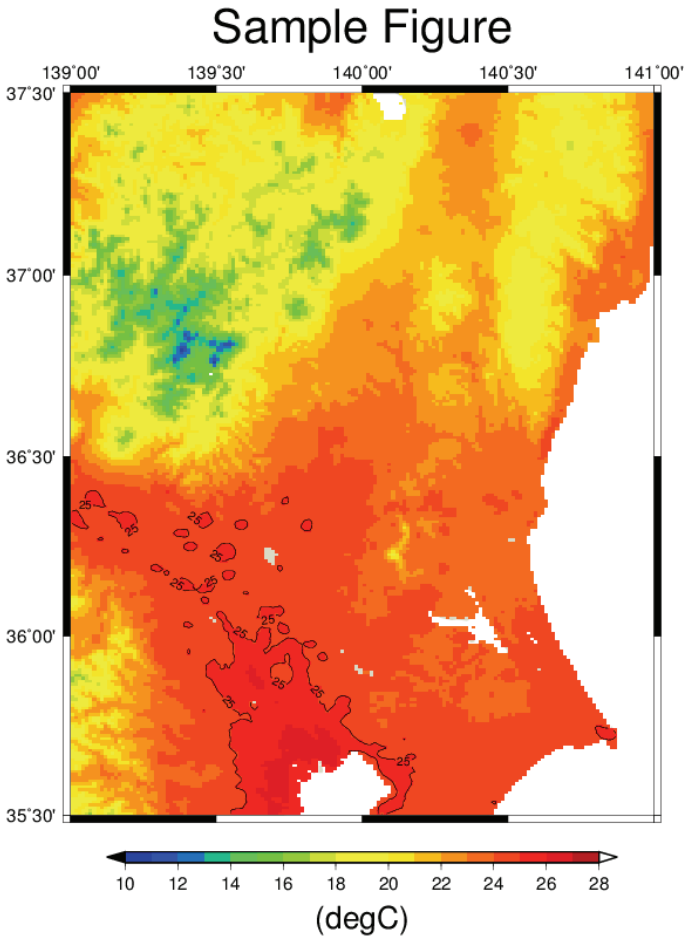

図45. 平均気温分布図の作成過程(3)

grdcontour コマンドで25℃の等値線を描画する.

−J：図法はさっき決めたとおり（J オプションの後ろに何も指定しない).
−Sn：スムージングせず忠実に等値線を描く.
−Q：どんなに小さい領域であっても等値線を描く.
−O：一つ前に書いた地図に重ね書きをする.
−K：このあとまだ作図が予定されている.

```
pscoast −R −J −W0.5p/0/0/0 −S200/225/255 −Dh −O −K >> figure.eps
```

図の見栄えを良くするために，この文で水域を上塗りし海岸線を引きます（図46). オプションの意味は次の通りです.
−R：緯度経度範囲はさっき決めたとおり（R オプションの後ろに何も指定しない).
−J：図法はさっき決めたとおり（J オプションの後ろに何も指定しない).
−W0.5p/0/0/0：海岸線は幅0.5ポイント，赤0／緑0／青0色の実線で描く.
−S200/225/255：水域を赤200／緑225／青255の色で塗る.
−Dh：海岸線を精細に描画する.
−O：一つ前に書いた地図に重ね書きをする.
−K：このあとまだ作図が予定されている.

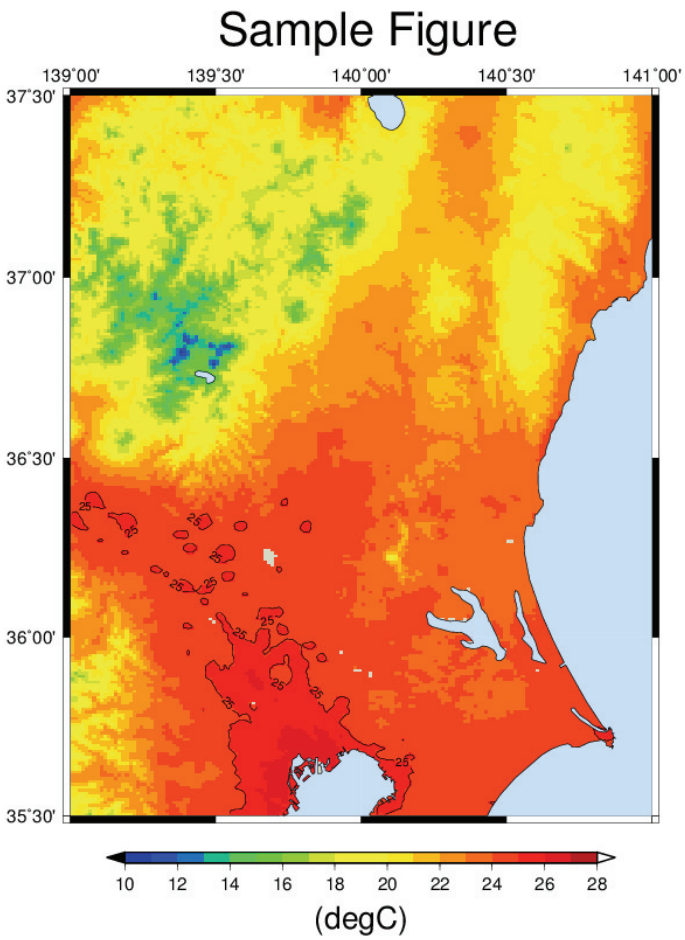

図46. 平均気温分布図の作成過程（4）

pscoast コマンドで海域を着色し海岸線を描く.

```
psxy PointList.txt -CPointList.cpt -R -J -W0.1p -Sc0.2 -O -K >> figure.eps
```

この文は，アメダス／気象台の位置に印をつける指示をしています（図47）．psxy は，テキストファイルに記述された緯度経度の地点に点や線を引くコマンドです．オプションの意味は次の通りです.

（第１引数）：PointList.txt に定義された地点情報に基づいて印を付ける.
−CPointList.cpt：印の色は PointList.cpt に定義されている（－Ｃオプション），
−R：緯度経度範囲はさっき決めたとおり（Ｒオプションの後ろに何も指定しない）.
−J：図法はさっき決めたとおり　（Ｊオプションの後ろに何も指定しない）.
−W0.1p：幅0. 1ポイントの線で図形を描く.
−Sc0.2：図形の形は0. 2cm の円とする.
−O：一つ前に書いた地図に重ね書きをする.
−K：このあとまだ作図が予定されている.

```
psxy ../common/PrefecBound.txt -W0.5pta/0/0/0 -R -J -M -O >> figure.eps
```

この文は，県境を重ね書きすることを指示しています（図42）．オプションの意味は次の通りです.

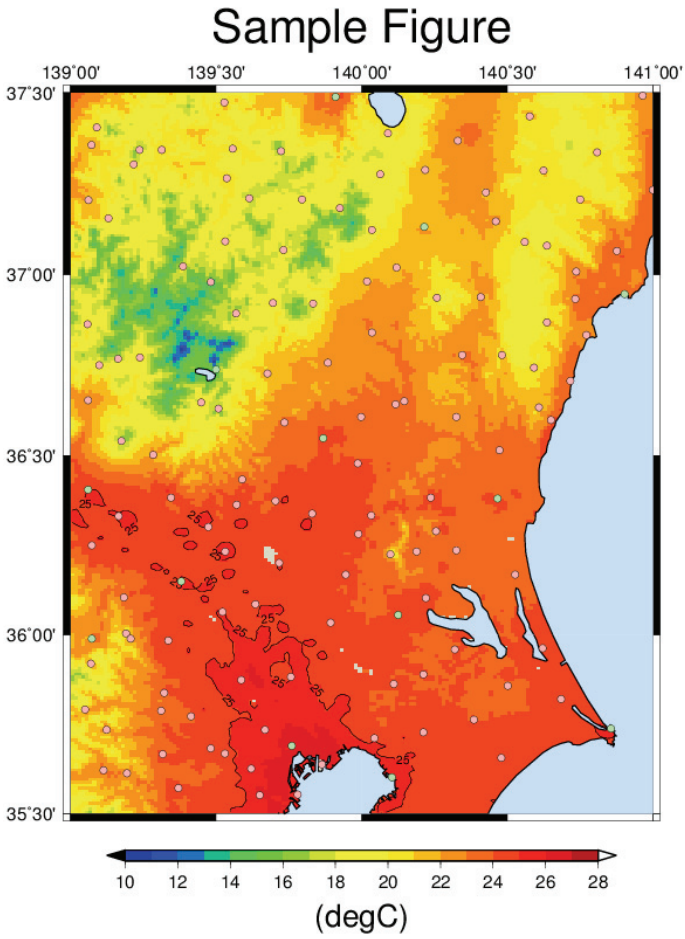

図47. 平均気温分布図の作成過程(5)

psxy コマンドでアメダス地点の位置に丸印を描く.

（第１引数）：../common/PrefecBound.txt に定義された地点情報に基づいて印を付ける.
−W0.5pta/0/0/0：幅0. 5ポイント，破線，黒色（赤０／緑０／青０）で線を引く.
−R：緯度経度範囲はさっき決めたとおり（Ｒオプションの後ろに何も指定しない).
−J：図法はさっき決めたとおり（Ｊオプションの後ろに何も指定しない).
−M：一筆書きとしない.
−O：一つ前に書いた地図に重ね書きをする.

さて，折角描いた手順書ですから，似た図を書くときに使い回したいものです．しかし，このスクリプトは使い回すのには少々不便です．例えば月を変えて何枚かの図を作る場合，出力するファイル名はそれぞれ別なものに変更しなければなりませんが，このためには，各コマンドの後ろに必ずあるファイル名をいちいち書き直さなければならずとても厄介です．そこで，これに手を加えて，スクリプトの１カ所を書き換えれば，それが全部に反映されるようにしてみます．このスクリプトはBOX11ようになります.

変数 output に実際のファイル名を覚えさせて，${output}でこれを呼び出し使い回します．このように，スクリプトでは変数（シェル変数）を使うことができます．これは GMT の機能ではなく Cygwin が持つシェルと呼ばれる機能です．実行されるときにシェル変数に値が代入されて GMT に渡されています．GMT から見ると BOX10と BOX11は全く同じ手順書です.

さて，シェルの機能を使うともっといろいろなことができます．筆者が口絵を描くときに実際

BOX11 シェルの機能を使い出力ファイル名を簡単に変えられるようにしたGMTスクリプト

（誌面の都合上，折り返している行がありますが，実際は1行で記述します.）

```
#!/bin/bash
output="figure.eps"                    #イコールの前後に空白を入れてはいけません.
#北関東の陸地を描きます.
pscoast -G220/220/200 -R139.0/141.0/35.5/37.5 -Jm8 -Ba0.5f0.5/a0.5f0.5:." Sample Figure":esWN -P -Dh -X2.5 -Y6.0 -K > ${output}

#分布図を重ね書きします.
grdimage ../../PythonWorks/ Ta_2012Sep.nc -CColPale.cpt -R -J -Sn -Q -O -K >> ${output}

#カラーチャートを描きます.
psscale -D8.0/-1.0/13.0/0.3h -CColPale.cpt -B2.0:"(degC)": -E0.5 -I0 -O -K >> ${output}

#25度Cの等値線を引きます.
grdcontour ../../PythonWorks/ Ta_2012Sep.nc -CContLeve.txt -G5c -W0 -R -J -Sn -Q -O -K >> ${output}

#水域を上書きします.
pscoast -R -J -W0.5p/0/0/0 -S200/225/255 -Dh -O -K >> ${output}

#アメダス/気象台の位置を重ね書きします.
psxy PointList.txt -CPointList.cpt -R -J -W0.1p -Sc0.2 -O -K >> ${output}

#県境を重ね書きします.
psxy ../common/PrefecBound.txt -W0.5pta/0/0/0 -R -J -M -O >> ${output}
```

BOX12 GMTで2013年9月の平均気温分布図を作成するための実用的なスクリプト

（誌面の都合上，折り返している行がありますが，実際は1行で記述します.）

```
#!/bin/bash
input='../../PythonWorks/ Ta_2012Sep.nc '
output='figure.eps'

#カラーパレットの作成
#  ベースとなるパレットを作成する.
makecpt -CGMT_seis -I -T10.0/28.0/1.0 > ColPal.cpt
#  超上限，超下限，Nullの色を上書き指定する.
sed -e 's/^B.*/B   0   0   0/g' ColPal.cpt -e 's/^F.*/F   255   255   255/g' -e 's/^N.*/N   -   -   -/g' > ColPale.cpt

#構図を定めるために，とりあえず簡単な地図を書いてしまう
title='Sample Figure'
region=139.0/141.0/35.5/37.5
size=8.0      # 地図のサイズ日本全国だと0.95ぐらい，関東だと8ぐらい
xanot=a0.5f0.5   # 横軸の目盛りの設定
yanot=a0.5f0.5   # 縦軸の目盛りの設定
pscoast -G220/220/200 -R${region} -Jm${size} -B${xanot}/${yanot}:."${title}":esWN -P -Dh -X2.5 -Y6.0 -K > ${output}

#分布図の描画：
grdimage ${input} -CColPale.cpt -R -J -Sn -Q -O -K >> ${output}

#カラーチャートを描画
```

```
23 psscale -D8.0/-1.0/13.0/0.3h -CColPale.cpt -B2.0:"(degC)": -E0.5 -I0 -O -K >> ${output}
24
25 #等高線の描画：
26 grdcontour ${input} -CContLeve.txt -R -J -G5c -W0p -Sn -Q -O -K >> ${output}
27
28 #水域や海岸線などの上塗り
29 pscoast -R -J -W0.5p/0/0/0 -S200/225/255 -Dh -O -K >> ${output}
30
31 #アメダス地点の描画
32 cat ../Common/LoLa-JMA_Station.txt | awk '{print $1, $2, $5}' > PointList.txt  #観測地点情報リストから表示する情報を選ぶ．(3列目：アメダスか測候所かの情報)
33 psxy PointList.txt -CPointList.cpt -R -J -W0.1p -Sc0.2 -O -K >> ${output}
34
35 #県境の描画
36 psxy ../common/PrefecBound.txt -R -J -W0.5pta/0/0/0 -M -O >> ${output}
```

に使用したスクリプトを BOX12に示します．これには，シェル変数の他，Cygwin が提供するコマンドの sed や cat，awk なども使われていますが，残念ながらそれらについて説明する紙面がありません．インターネット上にはこれらの解説が豊富に存在するので，GMT を利用される方は，それらの解説で勉強してみて下さい．

Ⅶ　ソフトウエアのインストールと設定手順

1　Python のインストールと設定

Python のプログラムを作成して実行するには，ほとんどの場合，Python 本体以外に，オプションで提供されている数値計算やグラフ描画といったモジュールも目的に応じてインストールするのが普通であり，ネット上の様々なサイトから関連ファイルを集めて順次インストールするスタイルが一般的ですが，Python 本体と科学技術計算に必要な関連モジュール，それに専用エディタまでもが一気にインストールできる WinPython と呼ばれる Windows 用のディストリビューションが作られたので，これを利用することにします．

なお，Python には，バージョン２系とバージョン３系があり両方が使用されていますが，ここでは，関連モジュールが充実している２系の最新版であるバージョン2.7をインストールします．また，それぞれに，32-bit 版と64-bit 版がありますが，後述する理由から32-bit 版をインストールします．

1）すでにインストールされている Python のアンインストール

すでに Python がインストールされている PC に WinPython をインストールする場合には，以下の作業を実施して，まず，既存の Python をアンインストールしてください．

(1) Python と関連モジュールのアンインストール

これらには，アンインストーラーがあるので，これを利用します．Windows の「コントロールパネル」＞「プログラムのアンインストール」を選択して「プログラムのアンインストールまたは変更」ウインドウを開き，一覧の中から Python##を選択します．［アンインストールと変更］を押し，プログラムをアンインストールします．このとき，Matplotlib や，SciPyhton など

の関連モジュールを先にアンインストールし，Python 本体は一番最後にアンインストールします．

（2）個人設定フォルダーの削除

Python の関連モジュールのいくつかは，C：￥ユーザー￥user 名フォルダ内にフォルダを作って個人設定を保存していますが，アンインストーラーはこれらのファイルは消去しません．これらのファイルが残っていると，新しくインストールされた関連モジュールに対しても既存の個人設定が反映されて便利ですが，時としてエラーの原因になることがあるので，念のため消去しておきます．

ところが，これらのフォルダには隠しファイルの設定がされているので，デフォルト状態の Windows エクスプローラーでは表示されません．そこで，まず，この設定を変更し Windows エクスプローラーが隠しファイルを表示するようにします．Windows エクスプローラーを開き，「整理」ニューをクリックし，「フォルダーと検索のオプション」をクリックします（図48）．「フォルダーオプション」画面が表示されるので，「表示」タブをクリックします．「詳細設定」の「ファイルとフォルダーの表示」で，「隠しファイル，隠しフォルダー，および隠しドライブを表示する」にチェックをつけ，［OK］ボタンをクリックします（図49）．これで隠しフォルダーも表示されるようになります．あとは，C：￥ユーザー￥user 名フォルダー内にある「.matplotlib」と「.ipython」フォルダーを削除します．

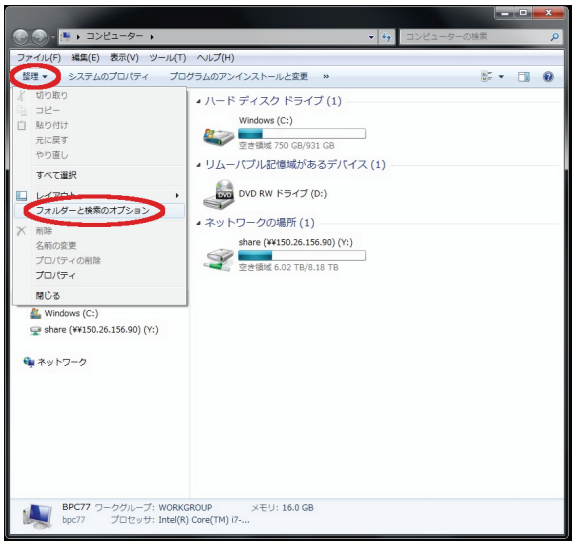

図48. Windows のファイルエクスプローラーの設定変更

「整理」メニューの中の「フォルダーと検索のオプション」をクリックする．

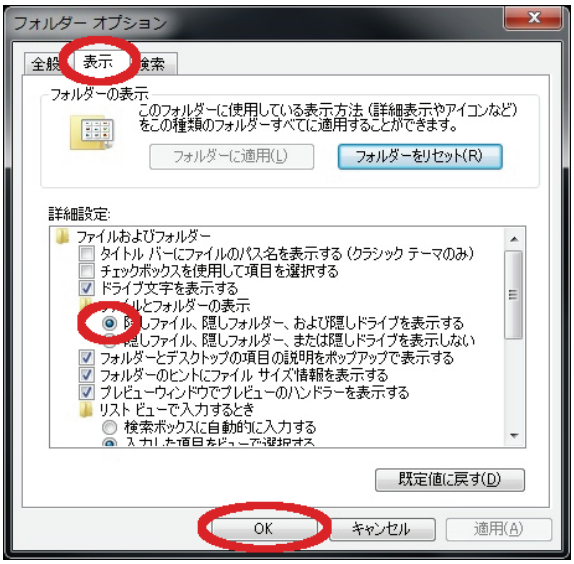

図49. Windows のファイルエクスプローラーの設定変更（つづき）

フォルダーオプション画面. 表示タブをクリックし，「詳細設定」の「ファイルとフォルダーの表示」で，「隠しファイル…表示する」にチェックを入れる．

2）WinPython のインストール

インストーラ：WinPython−32bit−2.7.5.3.exe

入手先：http://code.google.com/p/winpython/の左ペイン

(1) ファイルのコピー

インストーラを起動して利用条件に同意すると，図50のようなウインドウが表示されます．この状態では［install］ボタンは押せません．中央のテキストボックスにデフォルトで表示されている文字列の先頭に，「C：」を追記してください．［install］ボタンが有効になるので，押してインストールを実行します．この段階では，単にファイルがコピーされているだけの状態なので，次の手順で Windows にプログラムとして登録します．

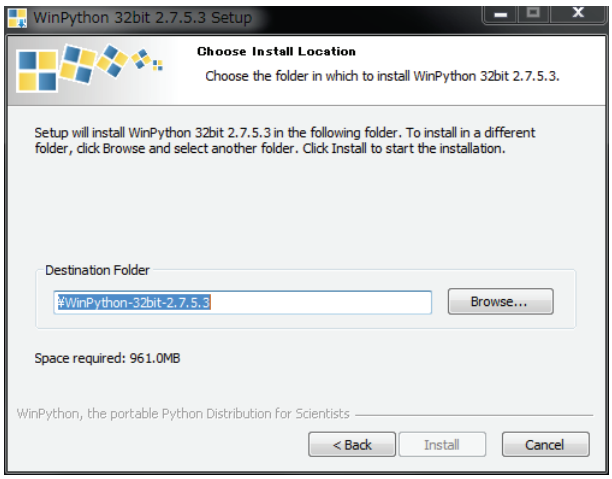

図50. WinPython インストーラ画面

利用条件に同意した後に表示される．予め入力されている「￥WinPython−32bit−2.7.5.3」の左に「C：」を追記してインストールを進める．

(2) Windows への登録

Windows エクスプローラーを用いて，インストールディレクトリの中を表示し，「WinPython Control Panel.exe」をダブルクリックして実行します．表示されるウインドウのメインメニュー「Advanced」から，「Register　distribution…」をプルダウンしてクリックします（図51）．黒い窓が数秒間だけ表示され消えたら完了です．

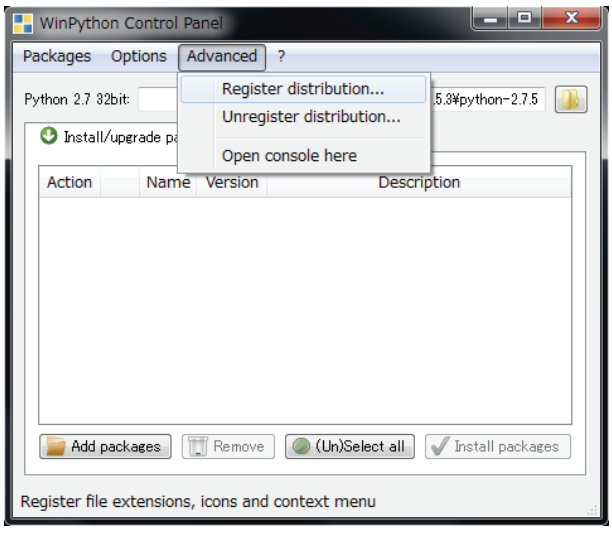

図51. WinPython のコントロールパネル画面

メインメニュー「Advanced」から「Register　distribution…」を選び，Windows にプログラムとして登録する．

（3）システム環境変数 Path の設定

モニタ左下のスタートボタンから，「コントロールパネル」＞「システム」と開き，さらに「システムの詳細設定」を選択して「システムのプロパティ」ウインドウを開きます．そして，右下にある［環境変数］ボタンを押します（図52）．すると図53のような環境変数ウインドウが開くので「システム環境変数（S）」のテキストボックスをスクロールして「Path」と言う変数を見つけてハイライトし，［編集］ボタンを押します．システム変数の編集ウインドウが表示されるので，変数値の末尾に，続けて「；C：￥WinPython−32bit−2.7.5.3￥python−2.7.5」を追記します．この際，最初のセミコロンを忘れないよう注意してください．追記したら，ウインドウ右下の［OK］ボタンを順に３回押して終了します．

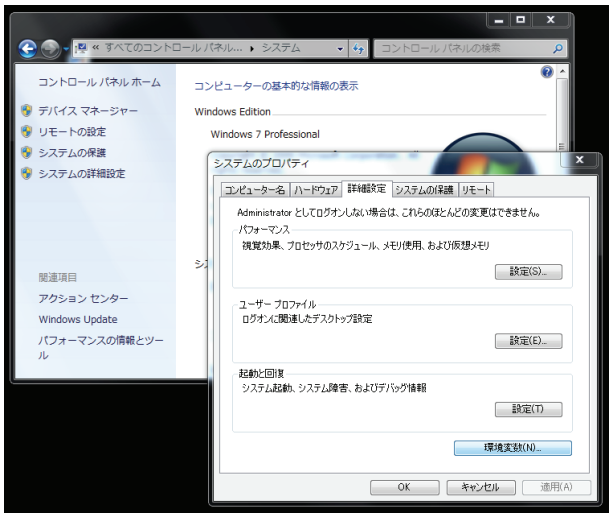

図52. 環境変数 Path の編集

「システムの詳細設定」で「システムのプロパティ」ウインドウを開き．環境変数ボタンを押す．

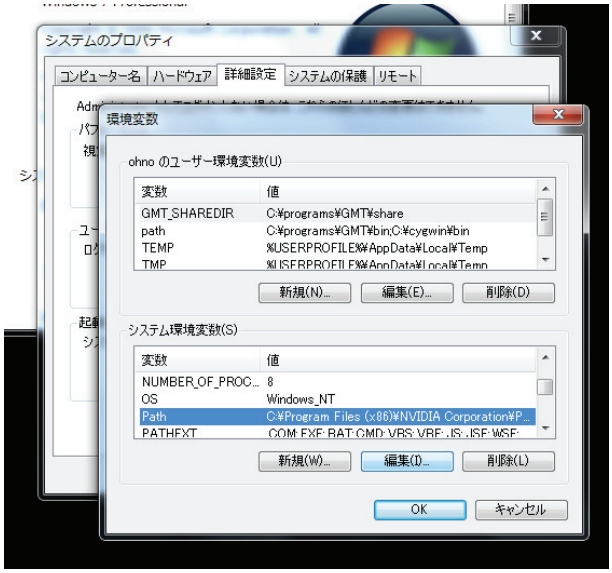

図53. 環境変数 Path の編集（つづき）

「環境変数」のウインドウで，システム環境変数の中から Path を見つけて末尾に WinPython のパスを追加する．

3）matplotlib モジュールのアップデート

インストーラ：matplotlib−1.3.1.win32−py2.7.exe

入手先：http://matplotlib.org/downloads.html

WinPython に同梱されている matplotlib モジュール（バージョン1. 3. 0）は，PC にインストールされている一部のフォントをうまく制御できず，PC の環境によっては起動時にエラーを発生することがあるので，matplotlib モジュールのバージョンを1. 3. 1にアップグレードします．

ＰＣのＣ：￥WinPython−32bit−2.7.5.3にある，「WinPython Control Panel.exe」をダブルクリックして WinPython コントロールパネルを起動してください．「Install/upgrade　packages」のページが開いていることを確認して，ウィンドウ下部にある［Add　packages］ボタンを押します．ファイルを選択するウインドウが開くので，入手した「matplotlib−1.3.1.win32−py2.7.exe」を指定し，アップグレードします．この際，既存の matplotlib−1.3.0をアンインストールする必要はありません．

4）netCDF4モジュールの追加インストール

インストーラ：netCDF4−1.0.5.win32−py2.7.exe

入手先：http://code.google.com/p/netcdf4−python/の「Downloads」

WinPython には，数値計算やグラフィクスなどの機能が含まれていますが，メッシュ農業気象データ配信サーバーからにダイレクトにデータを取得するのに必要なモジュールが含まれていません．そこで，これを追加でインストールします．このモジュールのバイナリは32bit 版しか配布されていません．Python 本体もこれにそろえて，32bit 版をインストールします．

インストールの方法は，matplotlib モジュールのアップデートと同じ手順です．「WinPython Control　Panel.exe」をダブルクリックして WinPython コントロールパネルを起動し，「Install/upgrade packages」ページ下部の［Add packages］ボタンを押した後，入手した「netCDF4−1.0.5.win32−py2.7.exe」を指定してインストールします．

5）IPython へのショートカットの作成

フォルダーを一つ用意して，Python での作業をここでするようにしましょう．まず，適当な場所に「PythonWorks」などの名前でフォルダーを作ります．次に，インストールフォルダ（デフォルトではＣ：￥WinPython−32bit−2.7.5.3￥python−2.7.5）下の Scripts フォルダに“ipython.exe” という実行ファイルがあるので，このショートカットを作成して作業用フォルダに置きます．そして，このショートカットアイコンのプロパティを開いて，「リンク先」を

Ｃ：￥WinPython−32bit−2.7.5.3￥python−2.7.5￥Scripts￥ipython.exe qtconsole −−pylab=inline

とします．このとき，「作業フォルダー：」に，何かが書かれていたらそれを消去しておきます．設定が終わったら「ＯＫ」を押してプロパティウインドウを閉じます．最後に，このショートカットの名前を変更しましょう．「それゆけ Python！」とします．

このように設定すると，作業フォルダーを開いて，それゆけ Python！というアイコンをクリックするだけで Python プログラムを実行するためのウインドウ（IPython）が開くようになります．

6）AMD_Tools.py のコピー

AMD_Tools.py は，メッシュ農業気象データを利用する際に便利なツール類を集めたモジュールが記述されているテキストファイルです．これを利用者専用 Wiki からダウンロードして作業

フォルダに保存します.

7）プログラムファイルとエディタの関連づけ

Python のプログラムはテキストファイルで，ファイルの末尾を「.py」とするのが慣例となっています．そこで，このようなファイルをダブルクリックすると，専用エディタが自動的に起動するように設定します.

作業フォルダーを開き，AMD_Tools.py 等，ファイル名の末尾が「.py」となっているものを一つ選んで右クリックし，「プロパティ」ウインドウを開きます.「全般」タブの「プログラム：」の右にある［変更（C）...］ボタンを押します.「ファイルを開くプログラムの選択」ウインドウが開くので，［参照（B）...］ボタンを押して，「C：￥WinPython−32bit−2.7.5.3￥tools」を開き，その中の「SciTE.exe」を選択して［開く（O）］ボタンを押します.

BOX13 重要：データ配信サーバーの OPeNDAP 機能を利用する際の注意

一般に，インターネットへの出口に「プロキシーサーバー」という機器が設置されているローカルエリアネットワーク（LAN）上の PC では，プログラムがインターネット上のデータにアクセスできるように特別な設定をする必要があります．メッシュ農業気象データを利用する場合も，データを専用のサーバーからインターネットを介して取得するのでこれに相当し，設定が必要です．この設定方法は，次のとおりです.

1．プロキシーサーバーのホスト名と使用するポート番号を調べます.

ネットワーク管理者に聞けば教えてもらえますが，もし何らかの理由で教えてもらえない場合は，Web ブラウザで以下のサイトを閲覧してください．これらのサイトは，ホームページを閲覧する際にブラウザがホームページサーバーに送信している閲覧者の情報（サーバーはこれを頼りに内容を返信してくる）を表示してくれているサイトです.

http://www.hanamoku.com/env/
http://www.cman.jp/network/support/go_access.cgi
http://www.ugtop.com/spill.shtml
http://www.taruo.net/e/?

この表示を見ると，単なる閲覧でも，随分といろいろな情報が通知されていることがわかります．プロキシーサーバーを経由してインターネットに接続されている LAN では，表示のどこかに「プロキシー」または「proxy」としてこれに関する情報が表示されます．この表示から，プロキシーサーバーのホスト名とポート番号を取得します.

2．Windows に環境変数を設定します.

コントロールパネル ＞ システムとセキュリティ ＞ システムと辿って，詳細設定. さらに［環境変数（N）］を押します．そして，システム環境変数の［新規］ボタンを押して，以下を設定して［OK］ボタンを押してください．以降，アクセスできるようになります.

変数名：HTTP_PROXY
変数値：http://プロキシー.サーバーの.ホスト名:ポート番号

2 IDV のインストールと設定

IDV（Integrated Data Viewer）は米国の Unidata プロジェクト（http://www.unidata.ucar.edu/）が開発し無償で一般に公開している極めて多機能なデータ可視化ソフトウエアです．メニューも設定も全部英語なので使い方を理解するのに一苦労しますが，メッシュや GIS の表示，三次元表示，断面図作成，アニメーションなど，それを補ってあまりある機能があります.

IDV は Windows インストーラーが作られているのでインストール自体はとても楽です．ただし，インストールファイルを入手するにあたって，氏名等の登録が要求され，これが終了するとダウンロードサイトにアクセスできるようになります．執筆時点では，バージョン4.1が最新版

ですが，64bit，32bit 版ともに，Ⅴ-1-4)で示す機能に問題が認められるので，ここでは64bit，32 bit 版ともに動作確認のできているバージョン4.0u1をインストールします．

なお，すでに古いバージョンの IDV がインストールされている場合に時としてインストール後の設定がうまく行かないことがあるので，IDV のアンインストールから始めることをすすめます．

1）すでにインストールされている IDV のアンインストール

IDV にはアンインストーラーがあるので，これを利用します．Windows の「コントロールパネル」＞「プログラムのアンインストール」を選択して「プログラムのアンインストールまたは変更」ウインドウを開き，一覧の中から「Integrated Data Viewer ##」を選択します．［アンインストールと変更］を押し，プログラムをアンインストールします．次に，C：￥ユーザー￥user 名フォルダー内にある隠しフォルダー「.unidata」を削除します．これは IDV に加えた個別の設定が保存されるフォルダーで，アンインストーラーでは削除されません．このため，Windows エクスプローラーを用いて手動で削除します．

ところが，デフォルト状態の Windows エクスプローラーは，隠しフォルダを表示しない設定になっているためこれが表示されません．そこでまず，表示の設定を変えます．Windows エクスプローラーを開き，「整理」メニューをクリックし，「フォルダーと検索のオプション」をクリックします（図48）．「フォルダーオプション」画面が表示されるので，「表示」タブをクリックします．「詳細設定」の「ファイルとフォルダーの表示」で，「隠しファイル，隠しフォルダー，および隠しドライブを表示する」にチェックをつけ，［OK］ボタンをクリックします（図49）．これで隠しフォルダーも表示されるようになります．あとは，C：￥ユーザー￥user 名フォルダー内にある「.unidata」フォルダーを削除します．

2）インストーラーの入手

まず，Unidata の IDV のホームページにアクセスします．URL は次の通りです．http://www.unidata.ucar.edu/software/idv/

ここにあるリンク IDV version 4.0u1をクリックします（図54）．次に，遷移したページの「It can be downloaded at」（ここから取得できます）の後のリンクをクリックします（図55）．

ログイン画面が出るので，未登録の場合はリンク Register をクリックします(図56)．すると，登録情報入力画面が表示されるので，赤いアスタリスクが付いている箇所に氏名等を入力します．全部入力したら，ページの一番下にある利用規約を読み，同意する場合はチェックを入れ，［Submit Registration Information］ボタンを押して登録します（図57）．次に，登録情報確認ページが出ますので，この内容でよい場合は，下の［Finish Registration］ボタンを押して登録を完了します（図58）．

間もなく，登録したメールアドレスに「Unidata Website registration confirm」というタイトルのメールが届き，その中にユーザー名として使用したメールアドレスと，パスワードが記されています．先ほどの Unidata の IDV のホームページ（http://www.unidata.ucar.edu/account/account.jsp）上で，上の方にあるメニューの「Downloads」のサブメニューの中から IDV を選択します（図59）．すると，再び図56のログイン画面が出るので，今度は右側の Sign In 欄にメールで送られた Email Address と Password を入力し Sign In ボタンを押します．IDV ダウンロードページが開くので，4.0u1をクリックします（図60）．

すると，https://www.unidata.ucar.edu/downloads/idv/4_0u1/index.jsp へ遷移します(図61)．

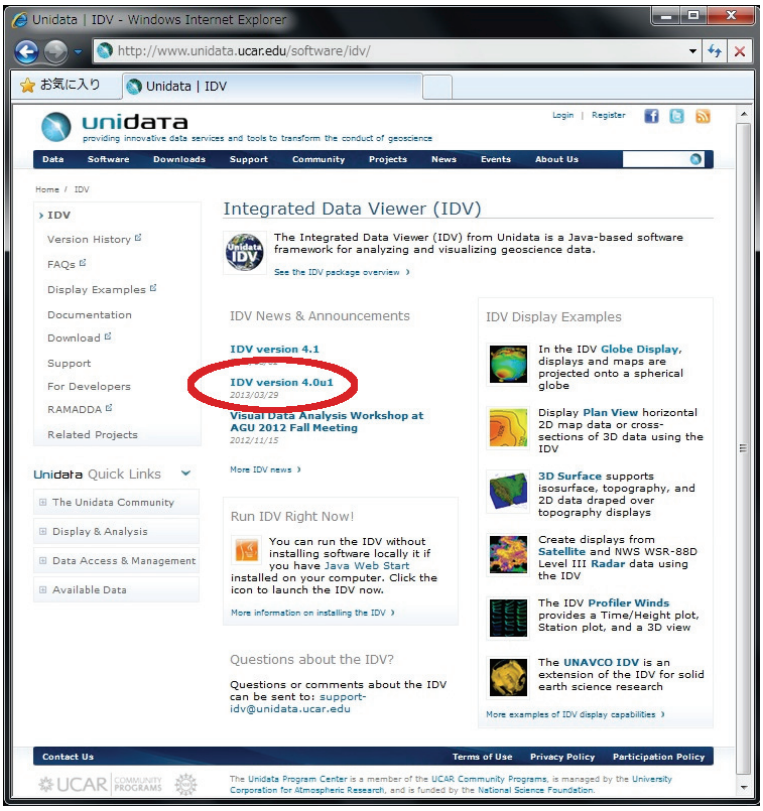

図54.　Unidata IDV のホームページ

赤丸のリンクをクリックして解説ページに進む.

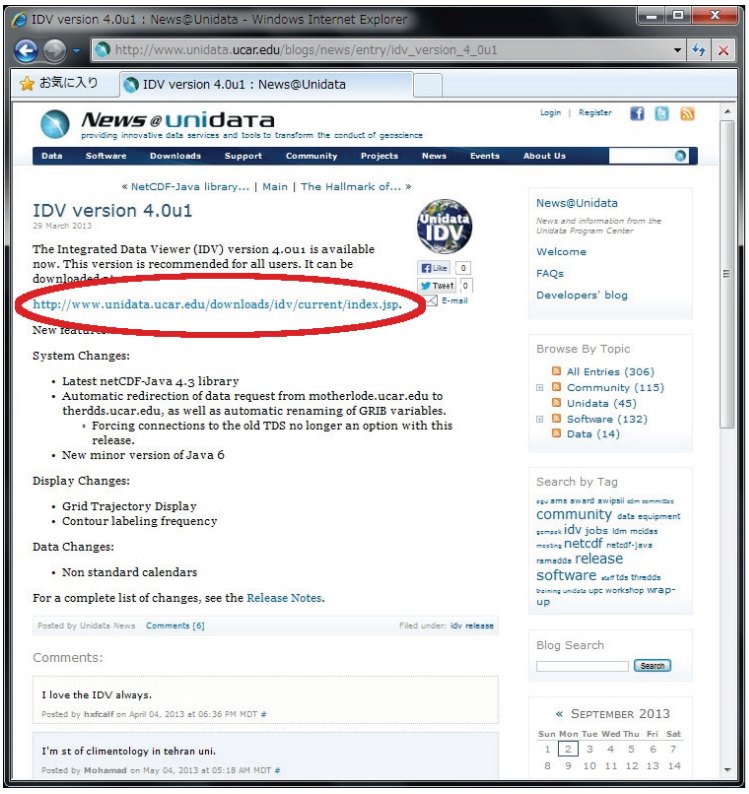

図55.　IDV バージョン4.0u1の解説ページ

赤丸のリンクをクリックしてダウンロードページに進む.

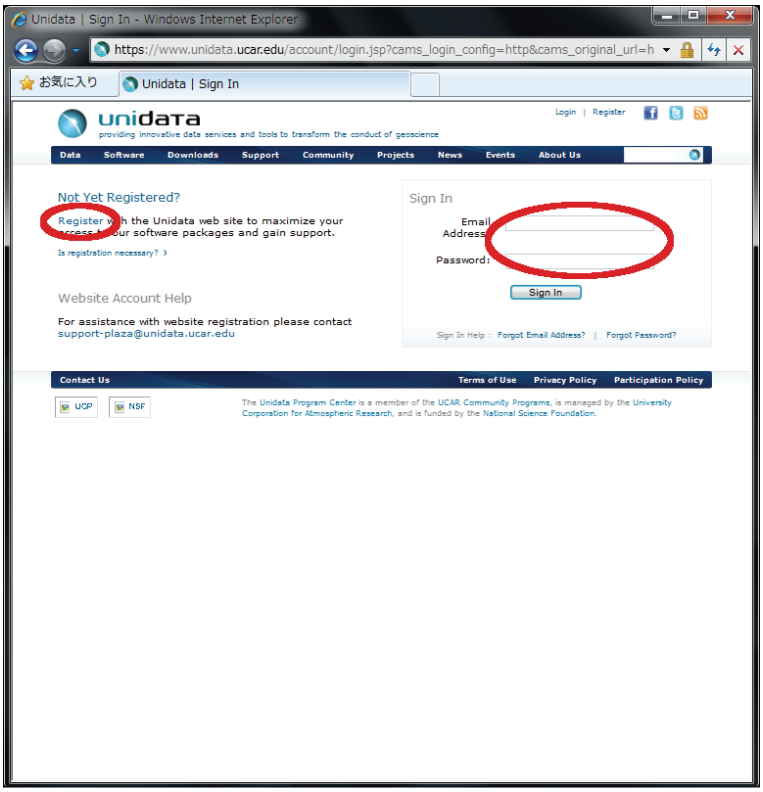

図56. 利用者ログインのページ

未登録の場合は左側の赤丸のリンクをクリックして登録ページに進む．登録済みの場合は右側の赤丸内に入力してログインする．

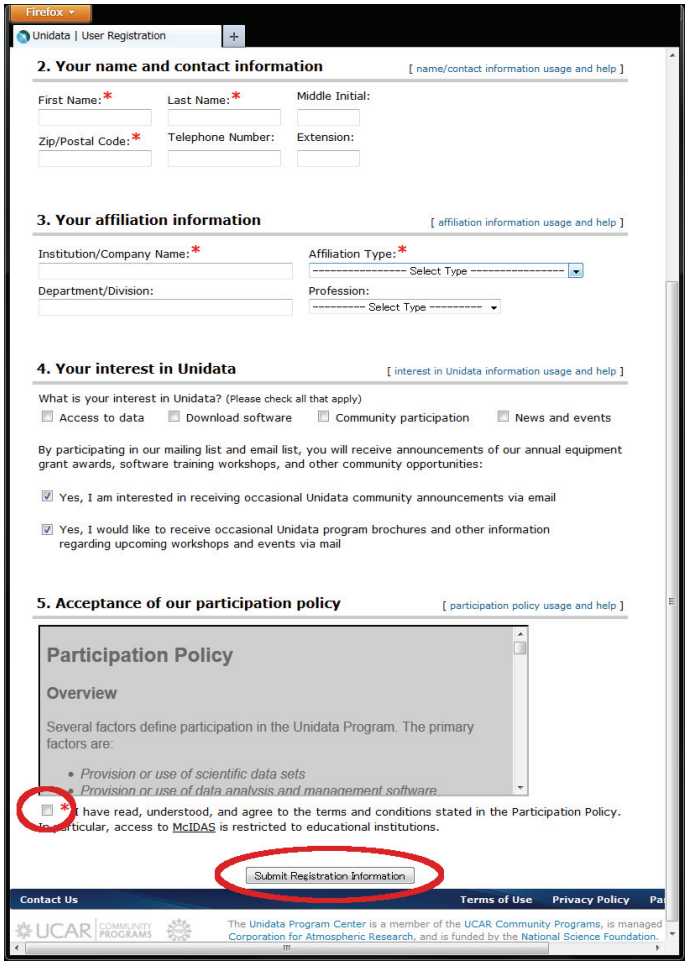

図57. 利用者登録のページ

必須事項を入力し，利用規約の同意にチェックをしたら，赤丸のボタンを押して登録を確定する．

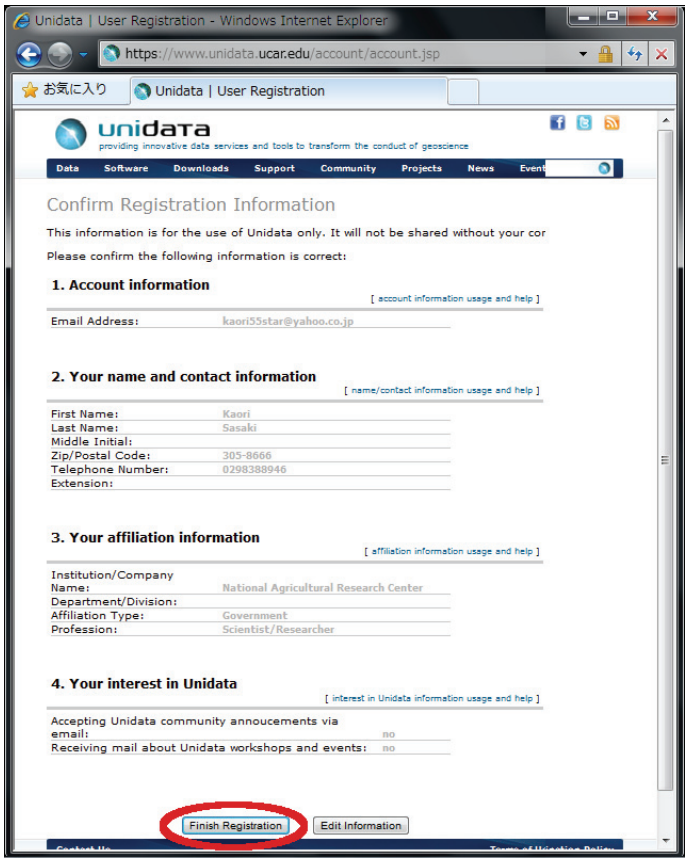

図58.　登録情報の確認ページ

これでよい場合は赤丸のボタンを押して登録を終了する.

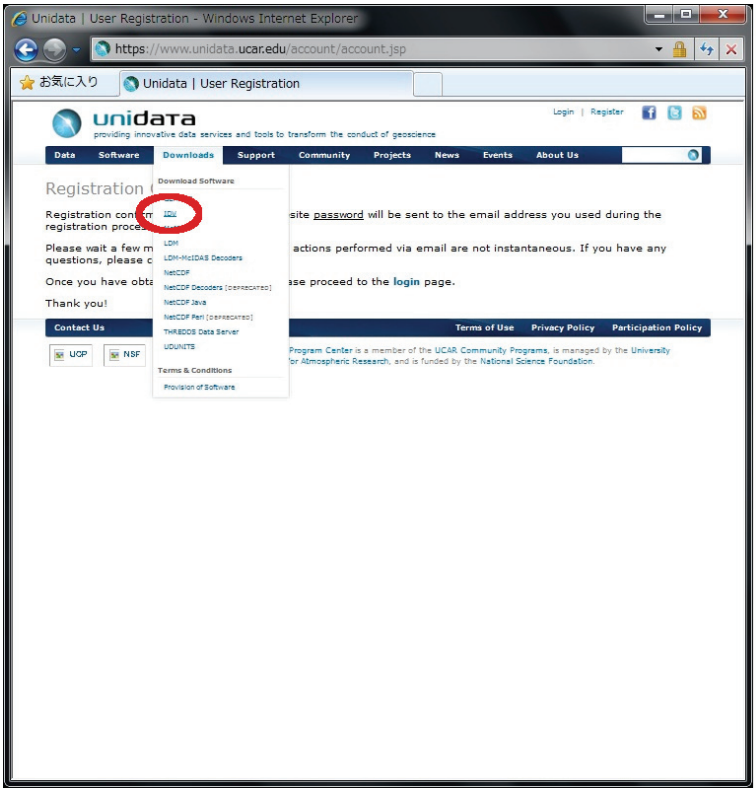

図59.　登録完了ページ

メニュー Downloads のサブメニューの中から IDV を選んでダウンロードに進む.

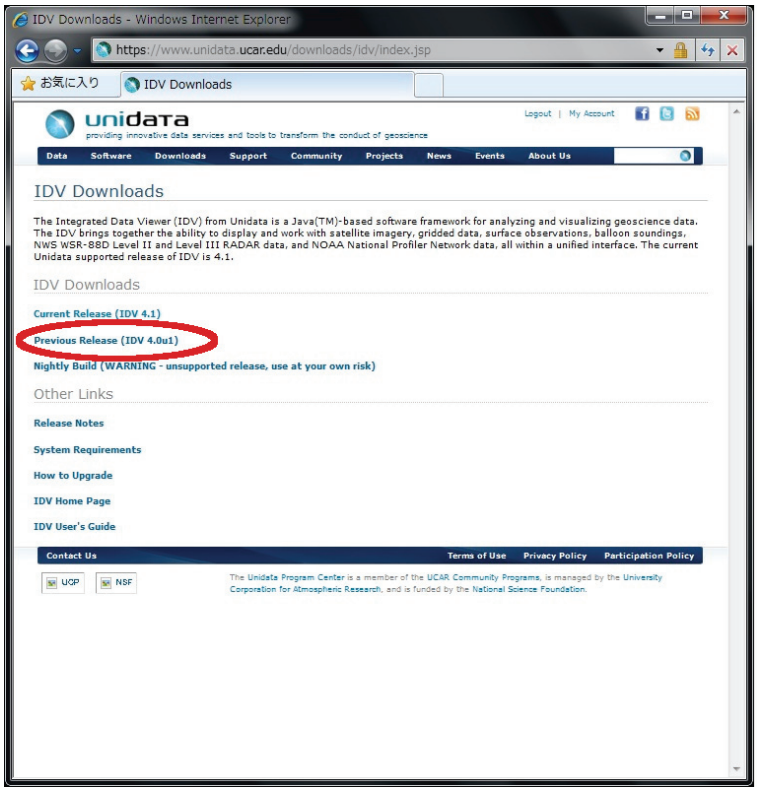

図60. IDV のダウンロードページ

「IDV 4.0u1」をクリックする.

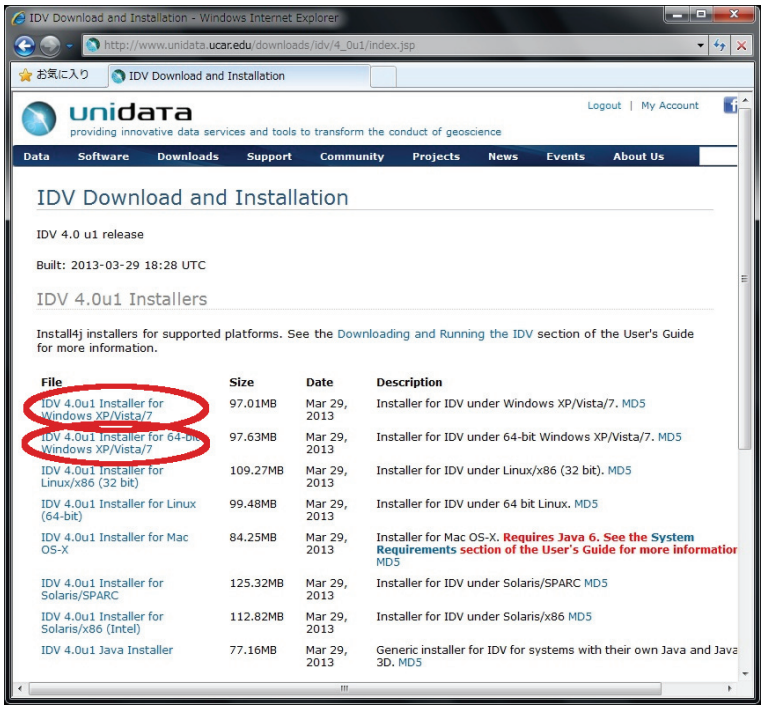

図61. IDV のダウンロードページ

赤丸のリンクをクリックしてインストールファイルをダウンロードする.

一番上のリンク「IDV 4.0u1 Installer for Windows XP/Vista/7」が Windows32bit 版で，リンク「IDV 4.0u1 Installer for 64−bit Windows XP/Vista/7」が64bit 版です．一般には前者で OK ですが，OS が64bit 版であることが確実で，ソフトも64bit 版で揃えたい場合には後者をダウンロードしてください．

３）インストール

インストールファイルをダブルクリックするとインストールプロセスが開始します．すべてデフォルトでインストールしてください．

４）設定

IDV は，タイトルバーに「Dashboard」と表示されているウインドウでデータの選択や，表示範囲，色合いなどを設定し，View と表示されているウインドウで画像を表示します．図の拡大縮小，表示範囲の移動などはこのウインドウで行います．

IDV は，とても高機能なデータ可視化ツールで，カスタマイズもかなり柔軟にできますが，英語版であることもあり，初めのうちは操作に手間取ることも少なくありません．筆者もまだ完全には理解していませんが，筆者が日本域の分布データを表示させるために使用している初期設定の仕方を説明します．

まず，日本地図を表示するようにします．Dashboard ウインドウのメニュー Edit から Preference を選択し，設定ウインドウを表示させます．設定可能項目は多岐に亘りますが，さしあたり，２番目と３番目のタブである，Format ＆ Data と，View を設定します．図62を参考に初期設定を変更して下さい．設定が終了したら［OK］ボタンを押します．すると，地図の背景が灰色に変わります．続いて，IDV を終了し，再び起動してください．

すると今度はアジア地域の地図が表示されます．Map View ウインドウ左端ツールを使って，日本地図が丁度良い形でウインドウに表示されるようにズームします．その状態で Dashboard ウインドウのメインメニューから File＞Default Bundle＞Save としてこの状態をデフォルトとし

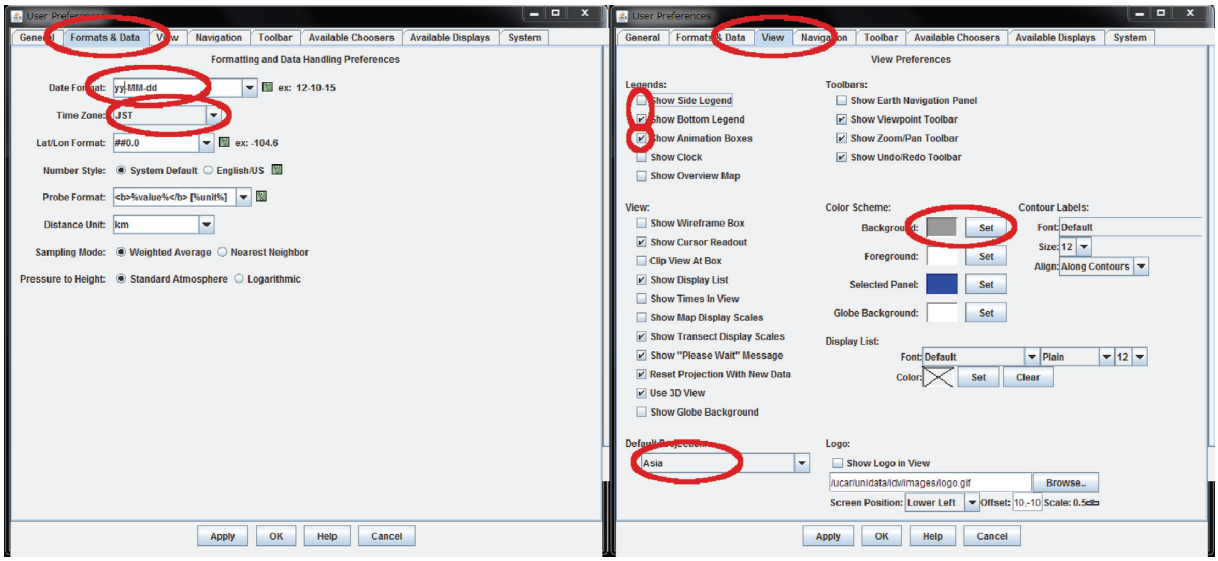

図62. IDV の設定ウインドウ

メインメニューから Edit > Preference と進んで表示させる．

て保存します（図63）．これで，以降 IDV を起動直後の初期画面は日本周辺になります．

なお，利用ガイドはホームページとして次の URL に用意されています（英語です）．
http://www.unidata.ucar.edu/software/idv/docs/userguide/

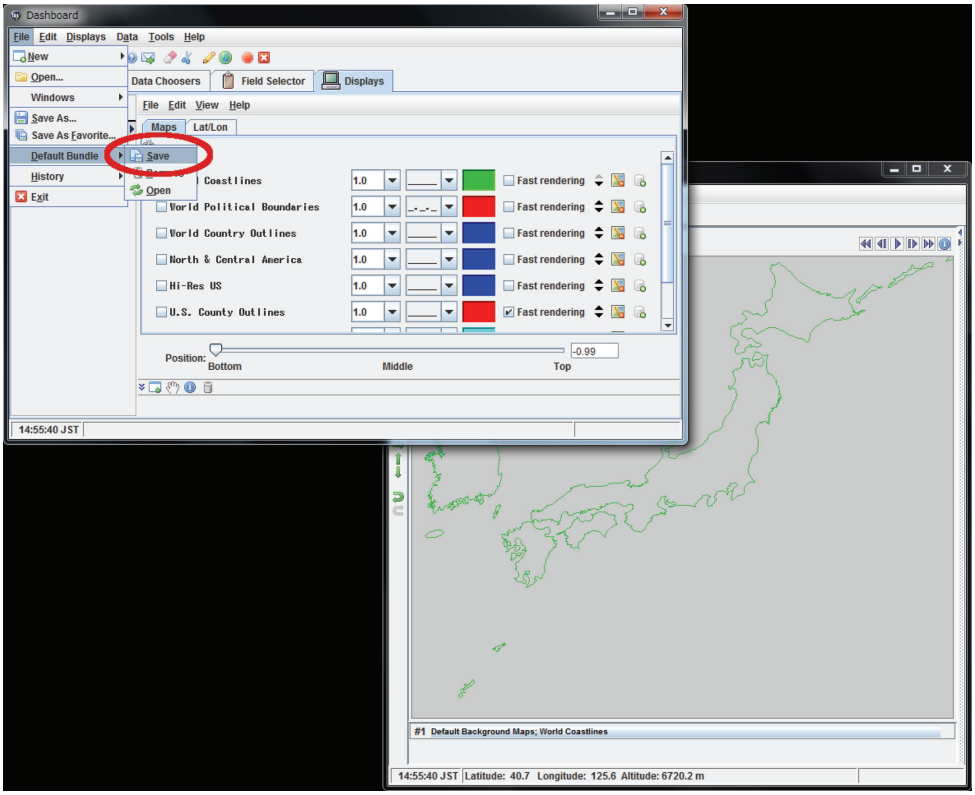

図63. 初期画面の記憶

日本付近を表示させ，それをデフォルトとして保存する．

5）実行の確認

インストーラが作るデスクトップアイコンをダブルクリックして実行します．ロゴが表示された後に３つのウインドウが開きます．そのうち真っ黒いウインドウは，操作に関係ないけれど閉じると IDV が終了してしまうウインドウです．気になるようであれば最小化しておくと良いでしょう．ただし，この黒いウインドウに下のようなエラーメッセージ；

[Jaba3D] Warning: Fail to lock Vertex Buffer = D3DERR_DRIVERINTERNALERROR

が幾つも表示されている場合は，インストールが失敗しています．これは，お使いの PC のディスプレイドライバーが IDV の要求を満たしていないために生じます．ドライバーを最新のものに更新して頂くか，残念ですがより新しい PC を使用してください．極まれに，IDV の設定を変更して三次元表示機能を停止するとエラーメッセージがでなくなることがあります．設定の変更は Preference ウインドウの一番左の最後の system タブで行います（図64）．

３　GMT のインストールと設定

GMT（Generic Mapping Tools）は，ハワイ大学の海洋地球科学技術教室が開発する，地図やメッシュデータ，地点観測データ，さらに，一般的なプロットグラフまでもたいへん綺麗に描画

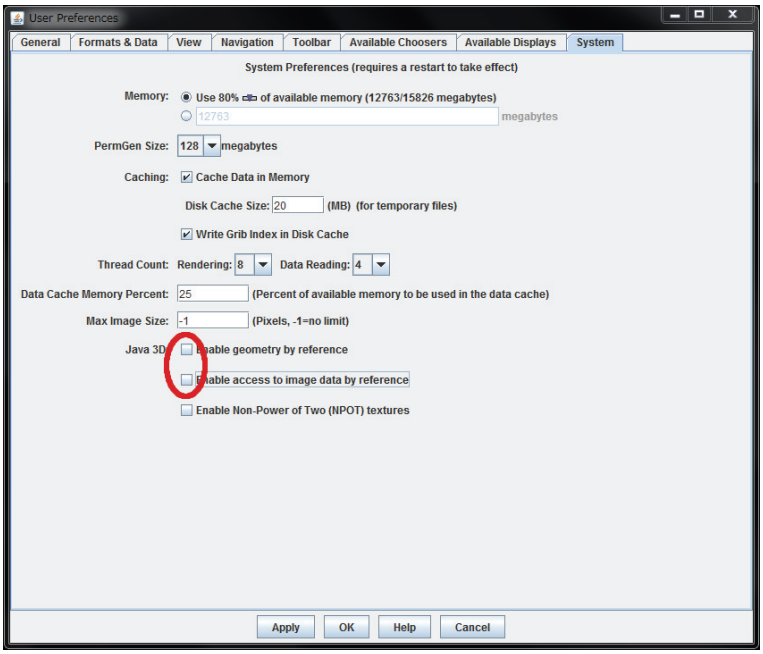

図64. グラフィックス機能が不足する PC の設定方法

IDV の設定ウインドウの System ページでチェックボックス外し三次元表示機能を停止する.

するフリーソフトウエアです．Windows 上で動作するグラフィックソフトウエアの殆どがマウスを操作しながらインタラクティブに図を作ってゆくのと対照的に，GMT は作図のためのコマンドを手順書に書いた上でそれを実行して図を作ります．どちらかというと，プログラムを書いて実行する Python に似ています．ですから，繰り返し使う定番の図や学術論文に投稿する図をしっかり描くのに向きます.

GMT には Windows 版が用意されており，GMT 本体を PC にインストールするのはとても簡単ですが，GMT を実用的に利用するには,Windows 版 GMT の他に，Cygwin と GhostView という二つのツールが別途必要です．Cygwin は，Windows PC 上に Linux に似た操作環境を付け加えるツールで，ここでは，描画の手順書を GMT に受け渡す目的で使用します．GMT は描いた図を postscript と呼ばれるファイル形式で出力します．他の形式では出力できません．具合の悪いことに，Windows は postscript 形式のファイルを表示するツールを持っていません．そこで，GhostView をインストールし，GMT が描いた図を表示したり印刷したりできるようにします.

１）Cygwin のインストール

Cygwin のインストールは，インストーラー（setup.exe）で実行します（図65）．setup.exe は http://www.cygwin.com/から入手します．このインストーラーには，指定したサーバから必要なファイルを取り寄せる手順だけが書かれており，インストールすべきプログラムは含まれていません．このため，インストールにはインターネット環境が必要で，時間も20～30分程度必要です.

まず，ブラウザで http://www.cygwin.com/に接続し，このホームページから setup.exe をデスクトップなどにダウンロードして，それをダブルクリックします．インストーラーが開始すると，ウイザードが表示されるので順次実行します．基本的にはデフォルトを選択してください．Cygwin のインストール先はC：¥Cygwin とします（図66）.

図65. Cygwin のインストール(1)

ダウンロードした setup.exe をダブルクリックするとインストールウイザードが開始される.

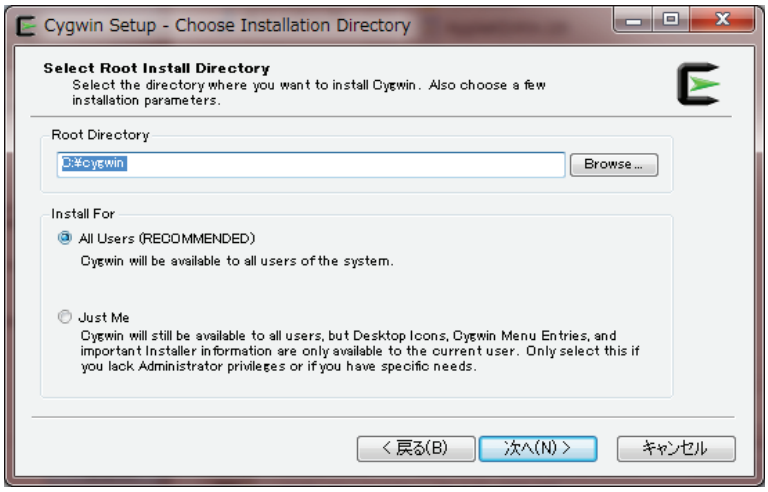

図66. Cygwin のインストール(2)

インストール先をデフォルト（C:¥Cygwin）とする.

インストールするプログラムの入手方法を指定します．デフォルトの，「インターネット経由」を選択します（図67）.

インストールするツールを選択します（図68）．デフォルトのまま次に進みます.

データを取得するサーバーを選択します（図69）．近くのサーバーを選ぶとインストール時間を短くすることができます．どこを選ぶかよく分からない場合は，ftp://ftp.jaist.ac.jp を選択します.

以上の設定の終了の後インストーラーはサーバーから必要なプログラムを取得して PC にコピーします．この間画面には進行状況を表す帯グラフが表示されます.

Cygwin のインストーラーが終了したら，どの作業フォルダにいても Cygwin の機能をすぐに呼び出せるように，システム環境変数「Path」に，Cygwin がインストールされているディレクトリを追加します.

図70のように，コントロールパネル＞システムを開き，ウインドウ左側にある「システムの詳

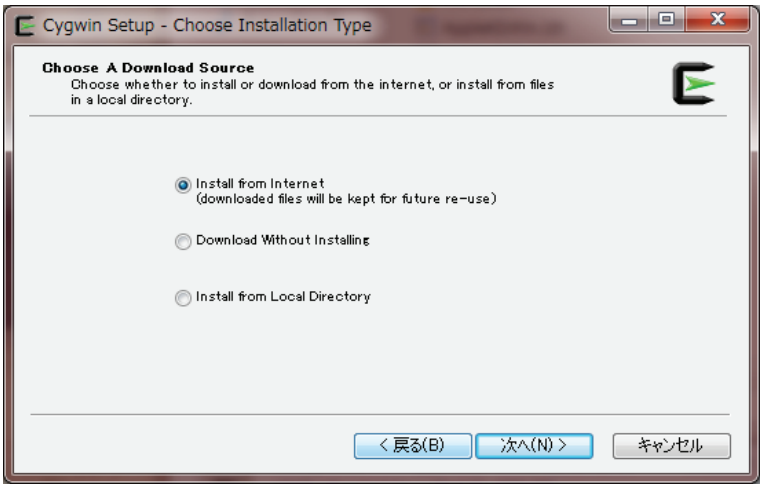

図67. Cygwin のインストール(3)

プロフラムファイルの入手方法の選択画面では「インターネット経由」を選択する.

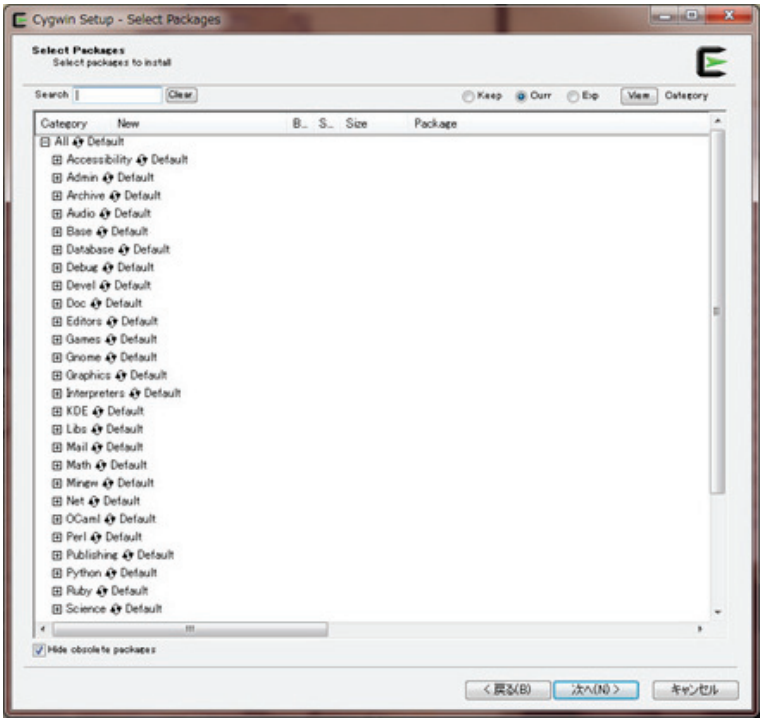

図68. Cygwin のインストール(4)

インストールするツールを選択する画面では，デフォルトのまま次に進む.

細設定」を選択します．すると，図71のようなウィンドウが表示されるので，システム環境変数「Path」をハイライトして，「編集（Ｉ）」を選択します．すると，文字列を入力する小さなウィンドウが開くので，この中の文字列の最後に，「;Ｃ:￥Cygwin￥bin」を追加します.

２）GMT 本体のインストール

GMT の Windows 版は，ハワイ大学のホームページ(http://gmt.soest.hawaii.edu/)，のミラーサイトである東海大学のサーバー（http://gmt.soest.hawaii.edu/gmt/gmt_windows_TOKAI.

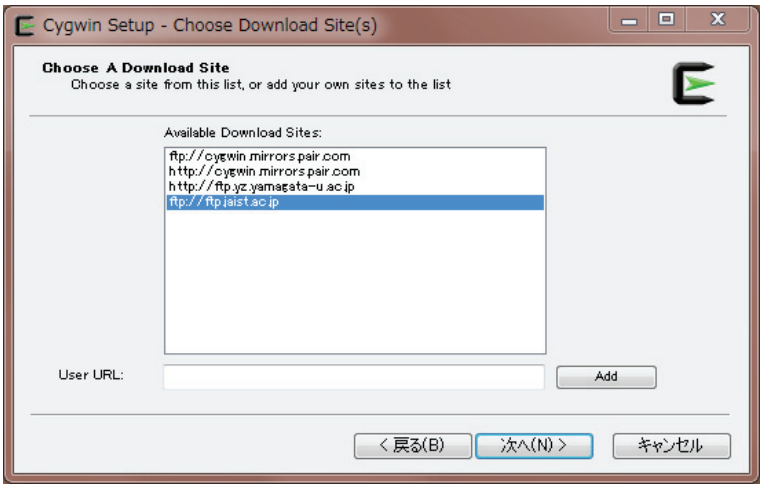

図69. Cygwin のインストール(5)

プログラムファイルを取得するサーバーを選択する画面では，ftp://ftp.jaist.ac.jp を選択する．

図70. 環境変数 Path の編集

スタートメニューからコントロールパネル＞　システムとして開く．さらに，左側の「システムの詳細設定」を選択する．

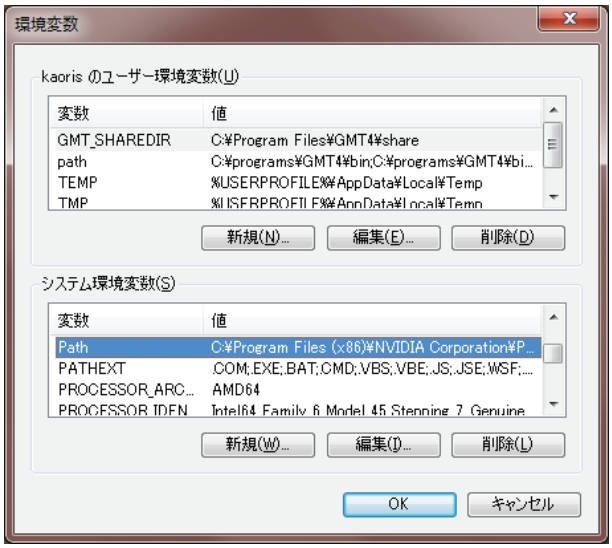

図71. 環境変数 Path の編集（つづき）

「環境変数」のウインドウでシステム環境変数「Path」をハイライトし，「編集(I)」を選択して末尾に Cygwin へのパスを追加する．

html）からダウンロードします.

執筆時点での最新インストーラーは gmt−4.5.9_install32.exe です．お使いの Windows が64bit 版の場合は，gmt−4.5.9_install_64.exe でも構いません．インストーラをダウンロードしてダブルクリックして実行します.

GMT のインストール先のデフォルトは「C：￥programs」になっています．特段の事情がなければ，ここにします（図72）.

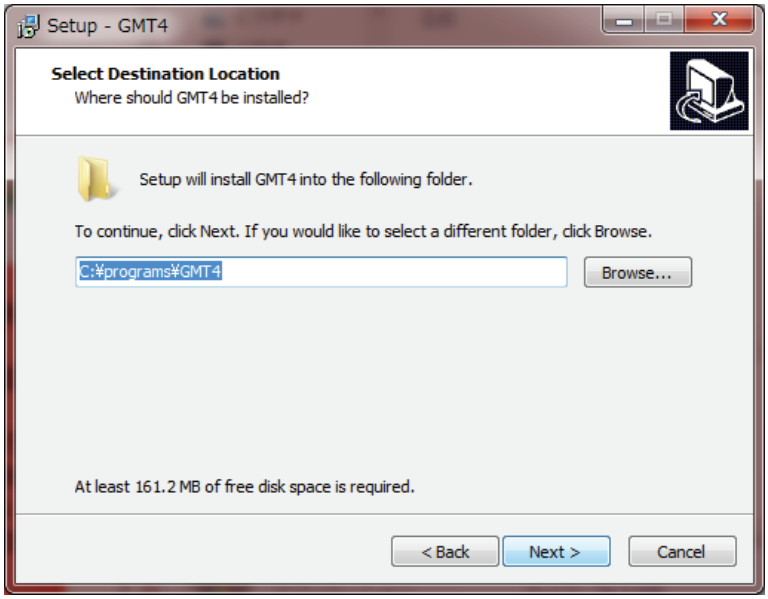

図72. GMT のインストール画面

デフォルト設定でインストールする.

３）海岸線データのインストール

海岸線データは GMT を利用するのに必須ですが，インストーラーが GMT 本体のものとは別になっています．GMT 本体と同じサイトから「gshhg−2.2.2_install.exe」をダウンロードして実行してください．このとき，インストール先ディレクトリーをデフォルトではなく，以下に設定し直してください（図73）.

C：￥programs￥GMT4￥share￥coast

４）GhostView のインストール

少々ややこしいですが，GhostView は，GhostView と GhostScript からできています．まず，GhostScript のインストーラーをダウンロードします．http://www.ghostscript.com/download/gsdnld.html の表で，プラットフォーム＝「Ghostscript 9.06 for Windows（32bit）」，ライセンス＝「GNU Public License」の組み合わせのインストーラをダウンロードし，ダブルクリックして実行します．お使いの Windows が64bit 版の場合，プラットフォームは「Ghostscript 9.06 for Windows（64bit）」でも構いません.

次に，GhostView のインストーラーをダウンロードします．http://pages.cs.wisc.edu/~ghost/gsview/get50.htm の上の方に並んでいるインストーラより，「gsv50w32.exe」をダウンロードし，ダブルクリックして実行します．お使いの Windows が64bit 版の場合，「gsv50w64.exe」でも構いません.

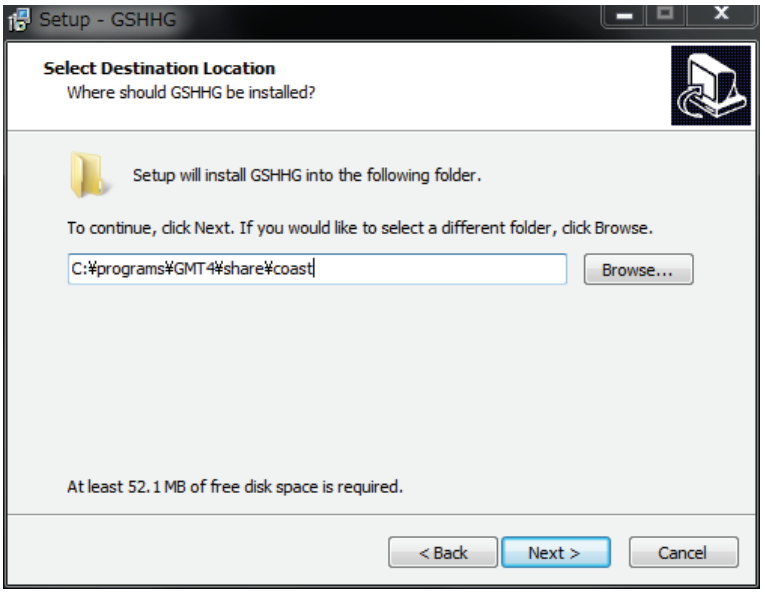

図73. 海岸線データのインストール画面

インストール先をデフォルトから図の通りに変更する.