連載講義

Python を利用した衛星画像の大気補正 第 1 回 衛星データの画像化

眞子直弘*

Atmospheric Correction of Satellite Image Using Python, Lecture Series 1: Imaging of Satellite Data

Naohiro MANAGO*

1. はじめに

Python とは，Guido van Rossum 氏が開発したプログラミング言語の一種であり，オブジェクト指向を取り入れたインタプリタ型スクリプト言語である。C 言語や Fortran のようなコンパイラ言語に比べ，少ない記述で同等の処理が可能であるために新規スクリプトの作成および既存スクリプトの解読が容易，コンパイルが不要であるために手軽に実行可能といった長所がある。Python はとにかくライブラリが充実しており，テキスト・バイナリデータ処理，数値演算，グラフ描画等，必要なデータ解析プログラムが Python だけで簡単に作れることが利点である。

本講義は 3 回の連載を予定しており，以下の順序で Python を利用して衛星画像の基礎的な大気補正を行う方法を解説する。

第 1 回 衛星データの画像化

Python を使ってひまわり 8 号の NetCDF データ，ひまわり標準データ，Gridded データを画像化する。

第 2 回 標準大気モデルを使った衛星画像の大気補正

Python を使って放射伝達コード 6S を制御し，6S に組み込まれた標準大気モデルを使ってひまわり 8 号衛星画像の大気補正を行う。

第 3 回 カスタム大気モデルを使った衛星画像の大気補正

分光放射計スカイラジオメーターで測定されたエアロゾルの物理・光学特性を使って衛星画像の大気補正を行う。

まず最初の今回は，Python を使って衛星データの画像化を行い，Python を使えばそれがいかに容易であるかを示したい。ここで扱う衛星には，2015 年 7 月 7 日からデータが公開され，今大いに注目を集めている最新の静止気象衛星「ひまわり 8 号」を取り上げる。ひまわり 8 号には Table 1 に挙げたいくつかの観測領域があり，気象庁でひまわり標準（HS）データ，NetCDF データ，PNG データ等が作成されている[1)]。また，フルディスクについては千葉大学から Gridded データが公開されている[2)]。以下では Fig. 1 に示した日本域 NetCDF データ，フルディスク HS データおよび Gridded データの画像化について説明する。

Talbe 1 Observation area of Himawari-8

Area	Interval	Data Format
Full disk	10 min.	HS HRIT LRIT Gridded PNG JPEG
Japan area	2.5 min.	HS NetCDF PNG
Target area	2.5 min.	HS NetCDF PNG
Landmark area	0.5 min.	N/A

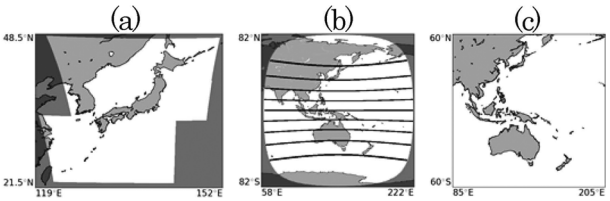

Fig. 1 Data format of Himawari-8
(a) Japan area NetCDF (lat-lon grid, calibrated)
(b) Full disk HS (AHI grid, raw DN, segmented)
(c) Gridded (lat-lon grid, raw DN)

Python はバージョン 2.7 以降（バージョン 3 も可）と以下のパッケージがインストールされていることを前提とす

（2016. 8. 26 受付）

* 千葉大学環境リモートセンシング研究センター
〒263-8522 千葉市稲毛区弥生町 1-33

* Center for Environmental Remote Sensing, Chiba University

Table 2　Wavelength/spatial resolution of Himawari-8

band	Wavel. (μm)	Res. (km)	band	Wavel. (μm)	Res. (km)
1	0.47	1	9	6.9	2
2	0.51	1	10	7.3	2
3	0.64	0.5	11	8.6	2
4	0.86	1	12	9.6	2
5	1.6	2	13	10.4	2
6	2.3	2	14	11.2	2
7	3.9	2	15	12.4	2
8	6.2	2	16	13.3	2

る。

・Numpy（数値計算モジュール）
・Scipy（科学計算モジュール）
・netCDF4（NetCDF モジュール）
・Matplotlib（描画モジュール）
・Basemap（地図モジュール）
・IPython（インタラクティブな Python シェル）

本講義の内容は 2012 年に千葉大学で行われた第 6 回「地球気候系の診断に関わるバーチャルラボラトリーの形成」講習会で実施した演習の内容をひまわり 8 号用にアレンジしたものであり，Python のインストール方法等については演習テキスト[3] に説明がある。

2.　NetCDF データの取り扱い

NetCDF（Network Common Data Form）データ形式は多次元配列データの格納に適したデータ形式の一種であり，HDF（Hierarchical Data Format）と並んで衛星観測データの保存に広く使われている。NetCDF には自己記述的でポータブルという特徴があり，あらかじめどのようなデータが入っているか知らなくても決められた手順に従って読み出すことができる。現在の NetCDF はバージョン 4 が主流であり，Python では netCDF4 というモジュールを使って扱える。

ひまわり 8 号の NetCDF データは Table 2 に示した観測バンド毎の空間分解能に応じた等緯度経度座標系になっており（例えば空間分解能が 1 km であれば 0.01 度間隔），各座標の物理量（バンド 1〜6：反射率，バンド 7〜16：輝度温度）が格納されている。NetCDF データのファイル名は NC_H08_yyyymmdd_hhnn_Bbb_cccc_Rjj.nc のようになっており，yyyy，mm，dd，hh，nn，bb，cccc，jj にはそれぞれ年，月，日，時，分，バンド，観測領域，空間分解能に基づく値が入っている。

Code 2.1 に NetCDF データを読み込む Python コードを示す。このコードは --pylab オプションを付けて起動した IPython 上で実行することを想定している。紙面の都合上複数行にまたがる行も実際には 1 行で書くこととする。

L	Code
1	`from netCDF4 import Dataset`
2	`nc = Dataset('NC_H08_20160512_0400_B04_JP01_R10.nc','r')`
3	`lat = nc.variables['latitude'][:]`
4	`lon = nc.variables['longitude'][:]`
5	`val = nc.variables['albedo'][:]`
6	`nc.close()`

Code 2.1　Reading NetCDF data

L	Code
1	`import matplotlib.pyplot as plt`
2	`plt.imshow(val,extent=(lon[0],lon[-1],lat[-1],lat[0]))`

Code 2.2　Plotting NetCDF data as image

（実際に Python コードの途中で改行する時は行末に ¥ 記号を付ける必要がある。ただし括弧内では ¥ 記号を省略できる。）Python ではインデントを使ってブロックを区切るが，以下の説明ではインデントを記号（□）で表す。また，Python の変数には内容が分かりやすい自己説明的な名前を付けることが推奨されているが，紙面節約のためにここでは敢えて短い変数名を用いる。Code 2.1 の 1 行目で netCDF4 モジュールの Dataset クラスを使用する準備をしている。import は使う前に一度実行すれば良く，以下で他のモジュール等を import する時も同様である。2 行目で NetCDF データを読み出し用に開いている。NetCDF に格納されている変数名が分からない場合は nc.variables.keys() を使って調べられる。3 行目で NetCDF に格納された ‘latitude’ という配列を指定し，全要素を変数 lat にコピーしている。行末の［：］は配列の範囲を指定するもので，例えば［10：20］とすれば 10〜19 番目の要素を指定できる。なお，配列の属性を知りたい場合，例えば v＝nc.variables［‘latitude’］に対して v.ncattrs() を調べると付随する属性名が分かり，v.getncattr（属性名）で属性値が分かる。

これで NetCDF データが読み込めたので，IPython 上で Code 2.2 を実行すれば Fig. 2 が得られる。2 行目で imshow を使って等間隔 2 次元データを疑似カラー画像として表示している。横軸，縦軸の値はデフォルトでそれぞれ列番号，行番号になるが，extent オプションで左，右，下，上の座標を与えられる。（lon［0］，lon［−1］はそれぞれ lon の最初と最後の要素を指す。imshow の詳細は plt.imshow? で表示される IPython のオンラインヘルプ参照。）得られた画像は例えば plt.savefig（‘netcdf.png’）のようなコードで PNG ファイルに保存できる。

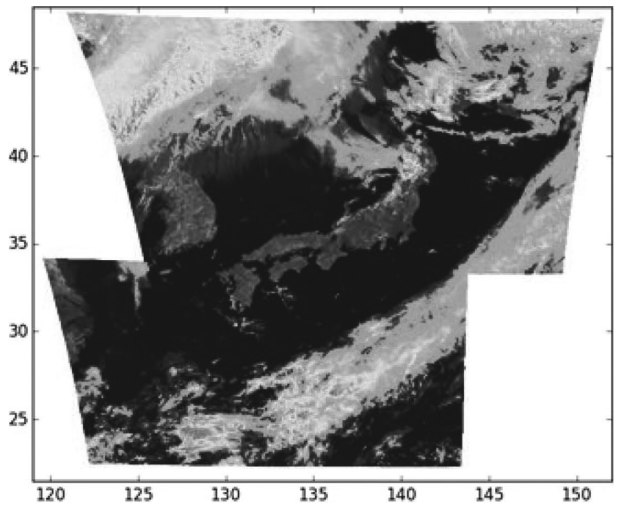

Fig. 2　Pseudo-color example image of NetCDF data

3.　HS データの取り扱い

3. 1　HS データの読み込み

HS データはひまわり独自のバイナリデータ形式になっており，可視赤外放射計（AHI）の観測で得られたカウント値（Digital Number）と様々なヘッダー情報が含まれている。ここでは HS データのユーザーガイド[4] に従って必要最低限の情報のみ読み出すことにする。

ひまわり 8 号のフルディスク HS データに収められている AHI 観測データの列数および有効ビット数を Table 3 にまとめる。ここで使う HS データは Fig. 1（b）のように 10 個のセグメントに分割されており，1 セグメント（1 ファイル）当たりの行数は列数の 1/10 である。また，AHI の各画素で得られたカウント値は有効ビット数に関わらず 2 バイトの符号なし整数として記録されている。HS データのファイル名は NetCDF データと同様，HS_H08_yyyymmdd_hhnn_Bbb_cccc_Skkll.DAT のようになっており，kkll にはセグメントに基づく値が入っている。

Code 3.1 に HS データを読み込むコードを示す。ここでは AHI 観測データの列数，行数とファイルサイズからヘッダーサイズを計算しているが，これらの値はヘッダー情報を解析して得ることもできる。4 行目では整数の商を得るために切り捨て除算演算子 // を用いている。（Python3 では整数同士に除算演算子 / を使うと浮動小数点数の商が得られる。）8 行目では fromstring の引数に dtype を与えて変数の型を指定している。‘u’ は符号なし整数を意味し，後ろの 2 はバイト数を意味している。‘u’ 以外にも ‘i’，‘f’ 等のフォーマット文字が使え，それぞれ符号付き整数，浮動小数点数を意味する。デフォルトのバイトオーダーは計算機に依存するが，フォーマット文字の前に ‘<’，‘>’ を置くとそれぞれリトルエンディアン，ビッグエンディアンを指定できる。型の指定方法はいくつかあるが，ここではコードが一番短いものを使用している。（例えば ‘u2’ の代わりに np.dtype（‘u2’）や np.uint16 が使える。）fromstring のデフォルトは ‘f8’ である。

Table 3　Himawari-8 HS data

band	#row	#bit	band	#row	#bit
1	11000	11	9	5500	11
2	11000	11	10	5500	12
3	22000	11	11	5500	12
4	11000	11	12	5500	12
5	5500	11	13	5500	12
6	5500	11	14	5500	12
7	5500	14	15	5500	12
8	5500	11	16	5500	11

L	Code
1	`import os`
2	`fnam = 'HS_H08_20160512_0400_B04_FLDK_R10_S0210.DAT'`
3	`NCOL = 11000`
4	`NLIN = NCOL//10`
5	`hsiz = os.path.getsize(fnam)-NCOL*NLIN*2`
6	`with open(fnam,'rb') as fp:`
7	`□head = fp.read(hsiz)`
8	`□data = np.fromstring(fp.read(), dtype='u2').reshape(NLIN,NCOL)`

Code 3.1　Reading HS data

3. 2　HS データのヘッダー情報の解析

以上により HS データが読み込めたので，例えば plt.imshow（data, vmax＝2**11）のようなコードで描画できる。しかしながら，読み込んだ値はカウント値であり，物理量を得るにはラジオメトリック校正が必要である。また，AHI 独自の空間座標系になっているために幾何補正も必要である。そのためにヘッダー情報の解析を行う。

Code 3.2 に HS データのヘッダー情報を解析するコードを示す。HS データのヘッダーは 11（＝imax）個のブロックに分かれており，各ブロックの先頭にブロック番号とブロックサイズが格納されている。2〜6 行目でヘッダーをブロック毎に分け，それ以後はブロック毎に必要なパラメータを取り出している。ここではシーケンスのアンパックという手法が使われている。シーケンスとはリストやタプル，ndarray といった Python の配列型のことである。例えば 7 行目の左辺は要素数 2 のタプル，右辺は要素数 2 の ndarray であり，右辺の ndarray がアンパック（開梱）されて左辺のそれぞれの要素に代入されている。（Python のタプルは丸括弧で囲んで表すが，ここでは丸括弧が省略されている。）要素数 1 のタプルは値の後ろにコンマを付ける

L	Code
1	imax, = np.fromstring(head[3:5], dtype='u2')
2	j,h = 0,[]
3	for i in range(imax):
4	□n, = np.fromstring(head[j+1: j+3],dtype='u2')
5	□h.append(head[j:j+n])
6	□j += n
7	sub_lon, = np.fromstring(h[2] [3:11])
8	cfac,lfac = np.fromstring(h[2] [11:19],dtype='u4')
9	coff,loff = np.fromstring(h[2] [19:27],dtype='f4')
10	p1,p2,p3,p4,p5,p6,p7 = np.fromstring(h[2][27:83])
11	band, = np.fromstring(h[4] [3:5],dtype='u2')
12	wlen, = np.fromstring(h[4][5:13])
13	verr,vout = np.fromstring(h[4] [15:19],dtype='u2')
14	gain,cnst = np.fromstring(h[4] [19:35])
15	lnum, = np.fromstring(h[6] [5:7],dtype='u2')

Code 3.2　Analysis of HS data header part

という規則があり，1 行目の左辺にはコンマが付いている。（コンマを付けると imax は整数値になるが，そうしないと ndarray になる。）

3. 3　HS データのラジオメトリック校正・幾何補正

ここで得られたヘッダー情報を基に HS データのラジオメトリック校正を行うコードを Code 3.3 に示す。3 行目では lrad＜LMIN が成立する lrad の要素の値を LMIN に変更している。cnd＝（lrad＜LMIN）とすると cnd は lrad が LMIN より小さい要素は True，そうでない要素は False の ndarray になる。（複数条件の and や or を取る場合は各条件を括弧で囲んでおくとよい。）このように ndarray の要素を選択するためには True または False のフラグが入った配列が使えるほか，欲しい要素のインデックスを入れた配列を使うこともできる。（例えば，indy, indx＝np.where（lrad＜LMIN）とすると lrad＜LMIN が成立するインデックスを取得でき，lrad［indy, indx］で要素を選択することができる。）4〜12 行目では追加のヘッダー情報を取得して校正された放射輝度（lrad）を反射率（バンド 1〜6）または輝度温度（バンド 7〜16）に変換している。

HS データの幾何補正方法としては，AHI 画像座標（列番号，行番号）を緯度経度に変換する方法と，適当な等緯度経度座標を AHI 画像座標に変換し，その格子点における AHI 観測値を 2 次元補間によって求める方法が考えられる。ここでは結果が等緯度経度座標系になる後者の方法を採用する。変換に用いる数式は LRIT/HRIT Global Specification[5] で説明されている。

L	Code
1	LMIN = 1.0e-60
2	lrad = gain*data+cnst
3	lrad[lrad < LMIN] = LMIN
4	if band > 6:
5	□wlen *= 1.0e-6
6	□lrad *= 1.0e6
7	□c0,c1,c2,c_0,c_1,c_2,c_c,c_h,c_k = np.fromstring(h[4][35:107])
8	□t_e = c_h*c_c/(c_k*wlen*np.log(2*c_h*c_c**2/(wlen**5*lrad)+1))
9	□val = c0+c1*t_e+c2*t_e**2
10	else:
11	□coef, = np.fromstring(h[4] [35:43])
12	□val = coef*lrad

Code 3.3　Radiometric calibration of HS data

Code 3.4 に HS データの幾何補正を行うコードを示す。ここでは ndarray のブロードキャスティングという手法が使われている。これを使うと，例えば要素数が（1, nx）と（ny, 1）の ndarray の 2 項演算をするとあたかもそれぞれが要素数（ny, nx）の ndarray であるかのように扱われる（足りない行または列には同じ値が使われる）。9〜12 行目ではブロードキャスティングが行われるように reshape（全要素数を保ったまま ndarray の次元数や各次元の要素数を変更すること）を行っている。ここで，1 つの次元の要素数に-1 を指定すると，その次元の要素数は全要素数が保たれるように自動計算される。

この例ではバンド 4 観測データの空間分解能が 1 km であることから東経 139.5°〜141°，北緯 36°〜34.8° の範囲を 0.01 度間隔で区切り，緯度経度座標を AHI 画像座標に変換して格子点上の AHI 観測値を RectBivariateSpline という 2 次元補間メソッドを用いて求めている。

以上の結果を用いて imshow（val_out, extent＝（lon［0］, lon［－1］, lat［－1］, lat［0］）, interpolation＝‘none’）を実行すると Fig. 3 が得られる。（以前の図は適宜 plt.clf（）で消去する。）

ラジオメトリック校正および幾何補正で得られたデータは np.save（‘val.npy’, val_out）というコードで ‘val.npy’ というファイルに保存できる。また，保存されたデータは val＝np.load（‘val.npy’）というコードで読み込める。

L	Code
1	from scipy.interpolate import RectBivariateSpline
2	lon = np.arange(139.5,141.001,.01)
3	lat = np.arange(36.0,34.799,-0.01)
4	col = np.arange(NCOL)+1
5	lin = np.arange(NLIN)+lnum
6	rad_lat = np.radians(lat)
7	c_lat = np.arctan(p5*np.tan (rad_lat))
8	c_lon = np.radians(lon-sub_lon)
9	cos_lat = np.cos(c_lat) .reshape(-1,1)
10	sin_lat = np.sin(c_lat) .reshape(-1,1)
11	cos_lon = np.cos(c_lon) .reshape(1,-1)
12	sin_lon = np.sin(c_lon) .reshape(1,-1)
13	rl = p3/np.sqrt(1.0-p4*cos_lat**2)
14	r1 = p1-rl*cos_lat*cos_lon
15	r2 = -rl*cos_lat*sin_lon
16	r3 = rl*sin_lat
17	rn = np.sqrt(r1*r1+r2*r2+r3*r3)
18	x = np.degrees(np.arctan(-r2/r1))
19	y = np.degrees(np.arcsin(-r3/rn))
20	col_out = coff+x/65536*cfac
21	lin_out = loff+y/65536*lfac
22	val_out = RectBivariateSpline (lin,col,val).ev(lin_out,col_out)

Code 3.4　Geometric correction of HS data

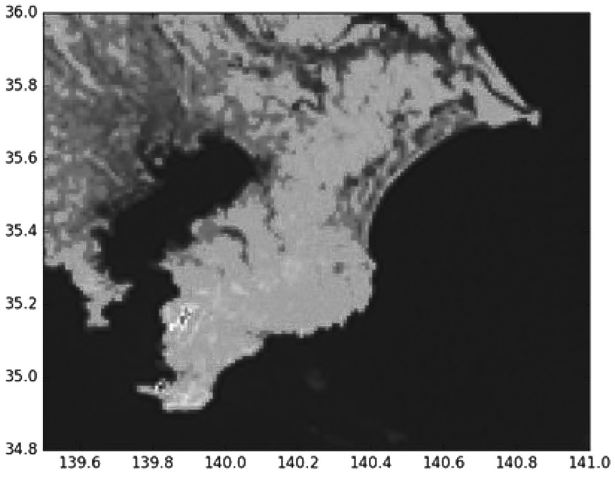

Fig. 3　Pseudo-color example image of HS data (around Chiba)

Table 4　Himawari-8 Gridded data

band	Gridded		#row	band	Gridded		#row
3	EXT	1	24000	15	TIR	3	6000
1	VIS	1	12000	16		4	
2		2		7		5	
4		3		8		6	
5	SIR	1	6000	9		7	
6		2		10		8	
13	TIR	1	6000	11		9	
14		2		12		10	

L	Code
1	data = np.fromfile('201605120400 .vis.01.fld.geoss',dtype= '>u2') .reshape(12000,12000)

Code 4.1　Reading Gridded data

4.　Gridded データの取り扱い

前述のように HS データの座標系は AHI 独自のもので少々扱いにくい面があるが，フルディスク HS データを等緯度経度座標系に変換した Gridded データが千葉大学の Web サイトで公開されている[2)]。また，Gridded データの取得から読み込み，校正，描画まで全自動で行ってくれる便利なスクリプトを含むサンプルプログラム（Fortran，C 言語）のセット（count2tbb_v101.tgz）も提供されている。ここでは Gridded データを Python で取り扱う方法について説明する。Gridded データにはこれまでの静止気象衛星データとの関連性から気象庁とは異なるルールでバンド名が付けられている。Table 4 にひまわり 8 号の観測バンド名と Gridded データのバンド名の対応を示す。例えばひまわり 8 号のバンド 3 は Gridded データの EXT1 に対応する。Gridded データのファイル名は yyyymmddhhnn.XXX.ZZ.fld.geoss のようになっており，XXX と ZZ には Gridded データのバンド名に基づく値が入っている。

Gridded データの範囲は東経 85°～205°，北緯 60°～南緯 60° で，西から東，北から南の順に格納されている。列数と行数は同じであり，バンド毎の列数は Table 4 の通りである。HS データと同じ 2 バイト符号なし整数がビッグエンディアンのバイトオーダーで記録されている。Code 4.1 に Gridded データを読み込むコードを示す。

Gridded データのサンプルプログラムセットには AHI の観測データを物理量に変換するためのルックアップテーブルが含まれている。これを用いて Gridded データのラジオメトリック校正を行うコードを Code 4.2 に示す。Gridded データのバンド名が付いたファイル（vis.01）にはカウント値と物理量（反射率の百分率）が入っており，1 行目で物理

L	Code
1	`lut = np.loadtxt('vis.01',usecols =(1,))`
2	`b = lut[data]`

Code 4.2　Radiometric calibration of Gridded data

L	Code
1	`g = np.loadtxt('vis.02',usecols= (1,))[np.fromfile('201605120400.vis.02.fld.geoss',dtype='>u2').reshape(12000,12000)]`
2	`r = np.loadtxt('ext.01',usecols= (1,))[(np.fromfile('201605120400.ext.01.fld.geoss',dtype='>u2').reshape(12000,2,12000,2).mean(-1).mean(1)+0.5).astype('u2')]`
3	`rgb = np.power(np.dstack((r,g,b)). clip(0.001,100.0)*0.01,1.0/2.2)`

Code 4.3　RGB composite of Gridded data

量のみを lut に読み込んでいる。今の場合，lut の各要素のインデックスはカウント値に等しく，2 行目でインデックスを使って lut の要素を選択して物理量に変換している。このようにインデックスを使って ndarray の要素を選択する場合，インデックスの数は ndarray の要素数と等しくなくても構わない。（True/False のフラグを使って ndarray の要素を選択する場合はフラグ数と ndarray の要素数が等しくないと Warning が表示される。）

VIS1（b）と同様に，VIS2（g），EXT1（r）のデータを読み込んでラジオメトリック校正を行い，RGB 合成を行うコードを Code 4.3 に示す。2 行目では，EXT1 だけ配列サイズが（24000, 24000）と大きいため，行，列ともに 2 ビン毎の小グループに分け，グループ毎に平均を取ることで 2×2 のビニングを行っている。3 行目では r，g，b を 1 つにまとめ，値の範囲を 0.001～100 に限定してから百分率を割合に直し，ガンマ補正をかけている。得られた RGB データは今まで同様 imshow を使って描画できるが，RGB カラーで正しく表示できるのは 1 バイト符号なし整数，または 0 以上 1 以下の浮動小数点数データに限られる。Code 4.4 に Basemap というモジュールを使って Gridded データに地図を重ねるコードを示す。（Basemap の詳細は Basemap? で表示される IPython のオンラインヘルプ参照。）このコードを実行すると Fig. 4 のような図が得られる。

5.　おわりに

以上，Python を使ってひまわり 8 号の各種データを画像化する方法をざっと説明したが，紙面の制約により説明できなかったことも多い。説明が不足している点については Web 上の情報や IPython のオンラインヘルプ等を参照していただければ幸いである。

L	Code
1	`from mpl_toolkits.basemap import Basemap`
2	`m = Basemap(projection="cyl",lat_0 =0,lon_0=145,llcrnrlat=-60,llcrnrlon=85,urcrnrlat=60,urcrnrlon=205,resolution='i',area_thresh=100,fix_aspect=True)`
3	`m.imshow(rgb,origin='upper')`
4	`m.drawcoastlines()`
5	`m.drawparallels(np.arange(-90,91,30),labels=[1,0,0,0])`
6	`m.drawmeridians(np.arange(0,360,30),labels=[0,0,0,1])`
7	`plt.xlabel('Longitude (deg)',labelpad=20)`
8	`plt.ylabel('Latitude (deg)',labelpad=40)`

Code 4.4　Plotting Gridded data with map

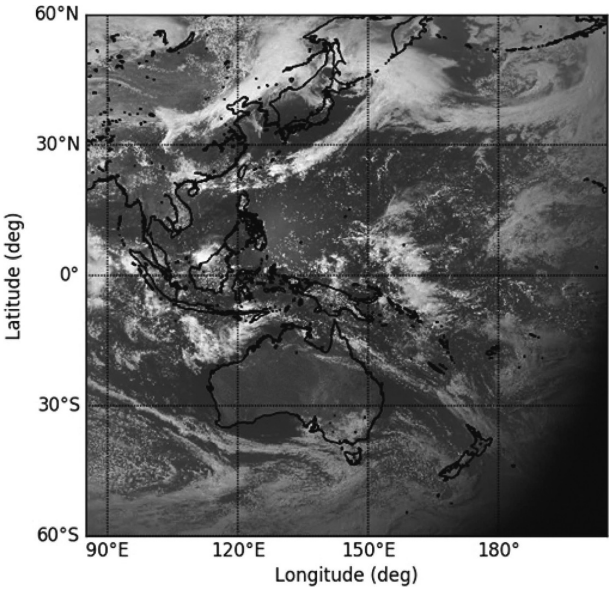

Fig. 4　RGB-color example image of Gridded Data

謝　辞：ひまわり 8 号の NetCDF データ，HS データは情報通信研究機構（NICT）のサイエンスクラウドから，Gridded データは千葉大学環境リモートセンシング研究センター（CEReS）から取得したものを使わせていただきました。

引 用 文 献

1）ひまわり 8 号・9 号
http : //www. data. jma. go. jp/mscweb/ja/himawari89/index. html (2016. 8. 25)
2）ひまわり 8/9 号 フルディスク（FD）gridded data 公開について
http : //www. cr. chiba-u. jp/databases/GEO/H8_9/FD/index_jp. html (2016. 8. 25)
3）衛星画像の大気補正，第 6 回「地球気候系の診断に関わるバーチャルラボラトリーの形成」講習会（http : //www. cr. chiba-u. jp/~kuze-lab/members/manago/VL/VL2012_Chiba University_PartB.pdf 2016. 8. 25）
4）Himawari-8/9 Himawari Standard Data User's Guide (http : //www. data. jma. go. jp/mscweb/en/himawari89/space_segment/hsd_sample/HS_D_users_guide_en_v12.pdf 2016. 8. 25)
5）LRIT/HRIT Global Specification (http : //www.cgms-info.org/documents/cgms-lrit-hrit-global-specification-(v2-8-of-30-oct-2013).pdf 2016. 8. 25)