科学技術館CanSatプロジェクト
第１期

第3回　データの可視化

今回の目標

・前回の最後に、GoogleEarthでGPSモジュールのデータを表示させました。
その過程で、GPSモジュールのデータの読み込みと時刻・緯度・経度のデータの抽出と、セルへの書き込みを行いました。今回はこの処理からスタートします。

・GPSモジュールのデータも、様々な見方がありえます。CanSatの大会で重要な見方の一つは、時刻毎のCanSatの位置です。

・緯度や経度で位置や速度を表しても、直感的な理解はできません。また、CanSatは高さ方向にも移動するので、3次元的な見方が必要になります。

・上記を踏まえて、今回はGPSモジュールのデータを「3次元プロット」にすることを目標とします。それに必要なことを、課題を交えながら一つずつ説明します。

今回の流れ

1　データの表示形式の変更

2　円に関わる数学

3　測地系の基礎

4　座標変換（その1）

5　地球規模での三次元プロット

6　座標変換（その2）

7　地域規模での三次元プロット

※参加者の皆様へ

全部を2時間以内にこなすのは難しいかもしれません。できるところまでやりましょう。

※独習・復習される方へ

この教材は、順番に読んでいただきたいのですが、1回読んで全てを理解しなくても大丈夫です。おそらく、後の方を読んでからでないとわからない所もあります。

実践しながら、繰り返し読むことをお勧めします。

※教材ファイル

kouza1-3教材.xlsm, GPSsample.txt, GPSsample2.txt, kouza1-3課題解答.txt を使います。（当日参加の方は「X:¥CanSat」から入手できます。）

1　データの表示形式の変更

・まずは、今回の教材となるxlsmファイルを開き、VBAエディタを開いてください。

・前回の課題4で、下記の処理を行いました。

＊GPSモジュールのデータの読み込み

＊時刻・緯度・経度のデータの抽出

＊セルへの書き込み

・このときの解答としたSubプロシージャが最初から入っています。（ kadai1_2_4()　）

サンプルデータも用意してあるので、このSubプロシージャを実行してみてください。

課題1：緯度・経度を正負の値で表現する

・kadai1()のソースコードは、あと一歩のところまで作ってあります。

・今回の目標のために、データを下記のように表現する必要があります。

＊経度・緯度の順にする。（必須ではないがわかりやすくなる）

＊東経(E)は正の値、西経(W)は負の値で表現する。

＊北緯(N)は正の値、南緯(S)は負の値で表現する。

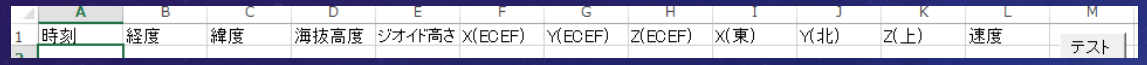
・そのために必要な変更を加えてください。

＜制限時間　5分＞

課題1の解答

・GGAレコード（$GPGGAがある行）の処理を、
　右のようにするとできます。

・ワークシートの上の方に書かれている通り、
　このあとは経度・緯度の順で処理をする
　ので、よく確認してください。



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	時刻	経度	緯度	海抜高度	ジオイド高さ	X(ECEF)	Y(ECEF)	Z(ECEF)	X(東)	Y(北)	Z(上)	速度	
2													

テスト

```
            '行頭に"$GPGGA"があったら、データをセルに入れる
            If spBuf(0) = "$GPGGA" Then
                If spBuf(2) <> "" Then
                    Cells(c, 1) = spBuf(1)  '時刻

                    '経度（東経は正、西経は負）
                    tmpd = convDMMtoDeg(CStr(spBuf(4)))
                    If spBuf(5) = "E" Then
                        Cells(c, 2) = tmpd
                    Else
                        Cells(c, 2) = -tmpd
                    End If

                    '緯度（北緯は正、南緯は負）
                    tmpd = convDMMtoDeg(CStr(spBuf(2)))
                    If spBuf(3) = "N" Then
                        Cells(c, 3) = tmpd
                    Else
                        Cells(c, 3) = -tmpd
                    End If


                    c = c + 1
                End If
            End If
```

例題1：時刻を秒単位にする

・GPSsample2.txtをメモ帳で開いてください。
・GGAレコードの時刻の表現を確認しましょう。
　HHMMSS.SSSとなっています。
　　（HHは時間単位、MMは分単位、SS.SSSは秒単位です。）

・このままでは、最初のレコードから何秒経過しているかが、すぐにはわかりません。

・そこで、この形式を秒単位に変換する方法を考えてみましょう。

・この変換は汎用的な処理なので、複数の場所で実施することが予想されます。
　このようなときは、Functionプロシージャを作ることをおすすめします。

例題1：時刻を秒単位にする

・この処理は下記のように分けてコーディングすると、わかりやすく、
また使い勝手がよくなるように思えます。（このあたりの考え方は、プログラマのセンスによりますが。）

(1)時刻の文字列を分解して、時・分・秒を取り出す。
(2)(1)で得られた時・分・秒から、秒数を計算する。

そこで下記2個のプロシージャを、miscモジュール内に作ってあります。

(1)Public Sub SplitGGATime(ggaTime As String, ByRef h As Long, m As Long, s As Double)
(2)Public Function ConvGGATimeToSec(ggaTime As String) As Double

これらのソースコードを確認しましょう。

例題1：時刻を秒単位にする

(1)Public Sub SplitGGATime(ggaTime As String, ByRef h As Long, m As Long, s As Double)

```
'GGAデータの時刻を時、分、秒に分解する
Public Sub SplitGGATime(ggaTime As String, ByRef h As Long, ByRef m As Long, ByRef s As Double)
    h = CLng(Mid(ggaTime, 1, 2))
    m = CLng(Mid(ggaTime, 3, 2))
    s = CDbl(Mid(ggaTime, 5, Len(ggaTime) - 4))
End Sub
```

・引数についている”ByRef”は、データを受け取るのではなく変数への参照を受け取ることを明示します。
これによって、引数とされた変数にデータを書き込めるようになります。
（実は、元々省略してもByRefとして解釈されますが、ソースコードを読む人のためにも明示することは重要です。）
（データを受け取りたいだけで、変数への書き込みを防ぎたいときは”ByVal”を使います。）

・ Mid([文字列],[取り出す文字列の先頭位置],[取り出す文字列の長さ])
これはVBAで既定の関数です。文字列を部分的に取り出して返します。
・ Len([文字列])
これはVBAで既定の関数です。文字列の文字数を返します。
・CLng([文字列])は整数型への変換、CDbl([文字列])は倍精度浮動小数点数型への変換関数です。

例題1:時刻を秒単位にする

(2)Public Function ConvGGATimeToSec(ggaTime As String) As Double

```
'GGAデータを0:00:00からの秒数に直す
Public Function ConvGGATimeToSec(ggaTime As String) As Double
    Dim h As Long, m As Long, s As Double

    SplitGGATime ggaTime, h, m, s
    ConvGGATimeToSec = h * 3600 + m * 60 + s

End Function
```

・GGAの時刻文字列の前処理にSplitGGAtimeを使います。

・h,m,sへの変換を内部でするのではなく、引数として与えると、より汎用的な関数になります。(汎用的とは、他の用途でも使えそうだということ)

ただし今回のこの関数の目的はGGAレコードの処理のみなので、

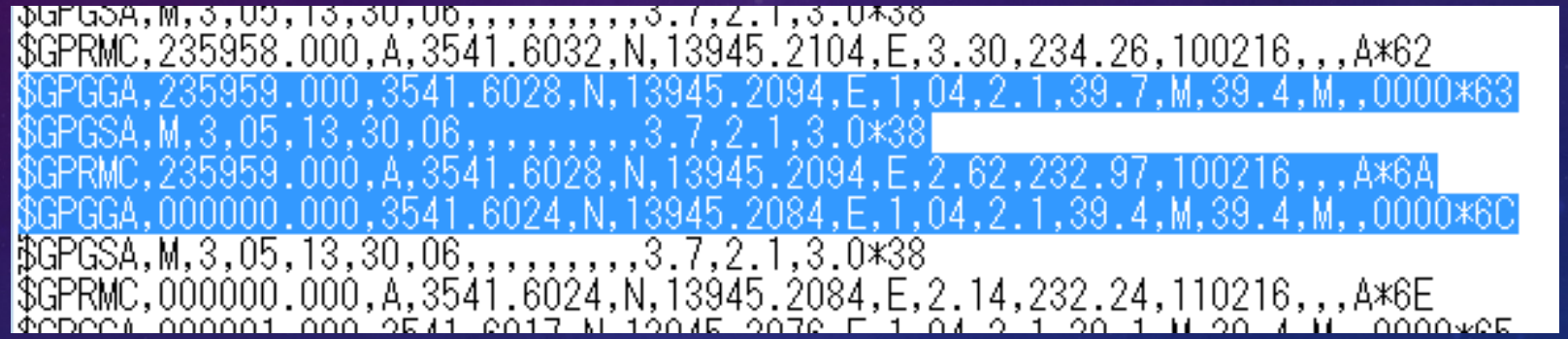
呼び出し側のソースコードが簡単になるような引数の取り方をしています。

(実際には、このようなことをあらかじめ考慮できるとは限りません。呼び出し側のソースコードを作りながら、思いつきで引数の取り方を変えるのもよくあることです。)

例題1：時刻を秒単位にする

・ConvGGATimeToSecで、00:00:00からの秒数が計算できますが、この時刻をまたぐと秒数がリセットされるという問題があります。

（GPSsample2.txtの431行目に、そのようなデータがあります。）



・この問題は、00:00:00をまたいだ（つまり日付が変わった）ことを検知し、そのたびに秒数を一日分（86400秒）加算することで解決できます。

例題1：時刻を秒単位にする

・test1()を実行し、GPSsample2.txtを読み込んでみてください。

・ワークシートに出力される時刻の形式と内容を確認してください。

・test1()の時刻処理がどのようにされているか確認してください。

```
            '行頭に"$GPGGA"があったら、データをセルに入れる
            If spBuf(0) = "$GPGGA" Then
                If spBuf(2) <> "" Then

                    '最初のGGAデータかどうか調べる
                    If cGGA = 0 Then
                        '最初の時刻を記録する
                        startSec = ConvGGATimeToSec(CStr(spBuf(1)))
                        '1つ前の時刻も現在時刻も同じということにしておく
                        prevSec = startSec
                        curSec = startSec
                    Else
                        '1つ前のGGAデータを退避させる
                        prevSec = curSec
                        '現在のGGAデータの時刻を取得する
                        curSec = ConvGGATimeToSec(CStr(spBuf(1)))
                    End If

                    '日付が変わっていないかどうか調べて、
                    '変わっていたら現在の日数を1日増やす
                    If curSec < prevSec Then
                        curDay = curDay + 1
                    End If

                    '現在の日数分の秒数に、現在の秒数を足し、最初の秒数を引くことで
                    '最初のデータからの経過秒数を求める
                    Cells(c, 1) = curDay * 86400 + curSec - startSec '時刻

                    '経度（東経は正、西経は負）
                    tmpd = convDMMtoDeg(CStr(spBuf(4)))
                    If spBuf(5) = "E" Then
                        Cells(c, 2) = tmpd
                    Else
                        Cells(c, 2) = -tmpd
                    End If

                    '緯度（北緯は正、南緯は負）
                    tmpd = convDMMtoDeg(CStr(spBuf(2)))
                    If spBuf(3) = "N" Then
                        Cells(c, 3) = tmpd
                    Else
                        Cells(c, 3) = -tmpd
                    End If


                    c = c + 1
                    cGGA = cGGA + 1
                End If
            End If
```

2　円に関わる数学

・CanSatが発射から何秒後にどの位置にあったか、具体的に指を差して示すためには、どのような情報が必要でしょうか。経度と緯度から直接わかることではありません。

・「発射地点から見て東にXm、北にYm、高さZmの位置にあった」ということが言えれば、大雑把ではあっても指を差せます。
また、メジャーを使ってその位置の真下に標識を置くこともできます。

・このX,Y,Zの値は、経度・緯度・高さから計算で求めることができます。

・そのための基礎として、円周上の位置を示す「角度」からXY座標を求める方法を説明します。

数学で扱う角度の単位：ラジアン

・半径1の円周の長さは、2×π です。（πは円周率(3.14159265358979…)）

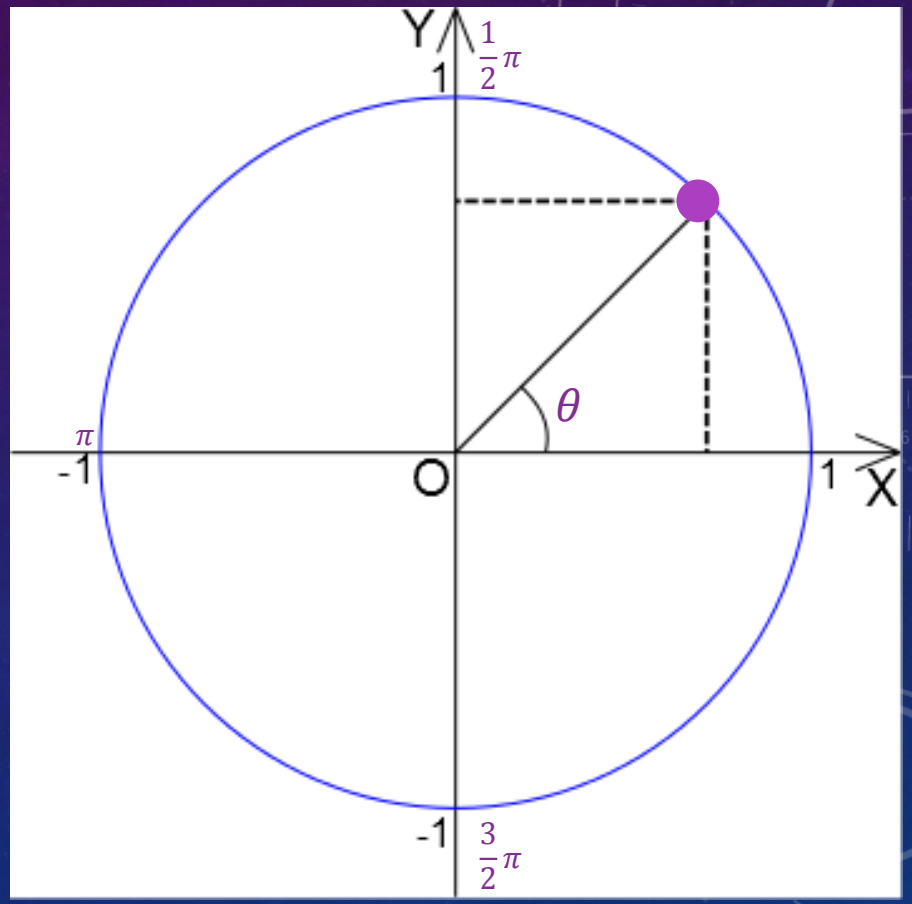
・そこで、360 deg(度)＝2π rad(ラジアン)ということにします。
そうすると、弧の長さで角度を表せるようになります。

シータ
θ (deg)を (rad)に換算するには　$\frac{\pi}{180}\theta$ (rad)　とします。

逆に　φ (rad)は　$\frac{180}{\pi}\varphi$ (deg) となります。

（例）　45 (deg) $= \frac{1}{4}\pi$ (rad)

$\frac{2}{3}\pi$ (rad) $=$ 120 (deg)

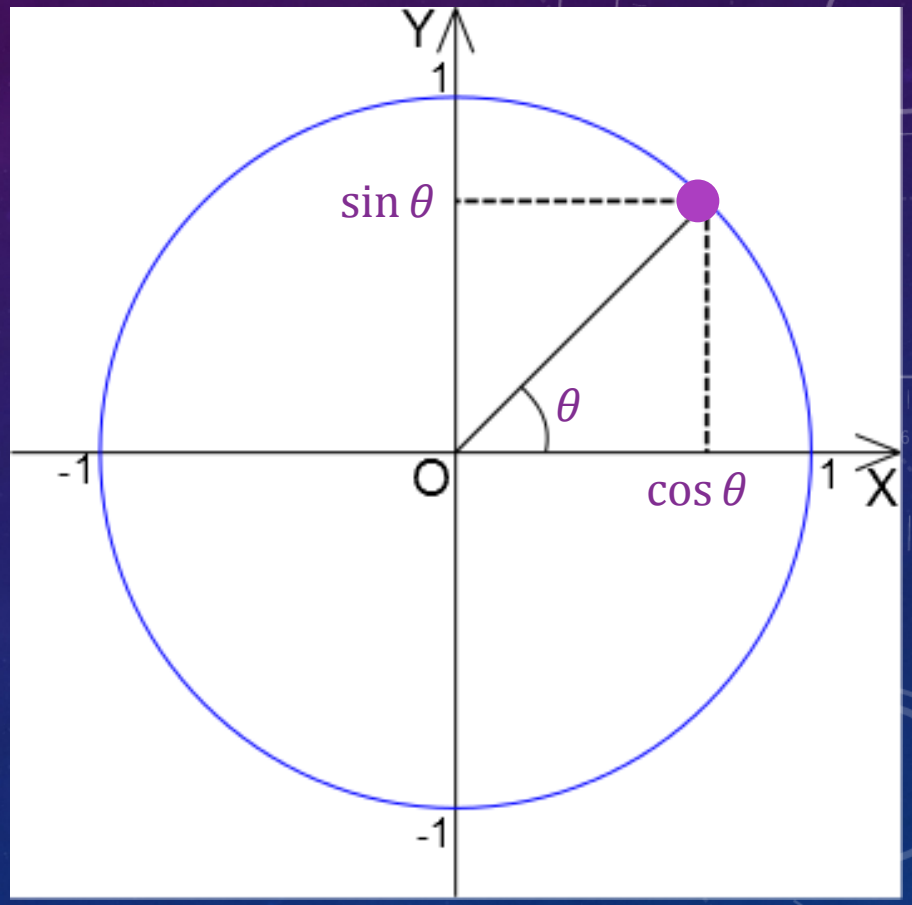


※小学校で1周は360度と習いますが、この360という数字に数学的な根拠はありません。
数学的に便利なように1周の角度を定めるとすれば、360よりは2πの方が合理的といえます。
（大学などで「オイラーの等式」を学ぶと、より実感することでしょう。）

三角関数：sin（サイン）, cos（コサイン）, tan（タンジェント）

- 半径が1の円周上の特定の点は、図のように角度θで表すことができます。
- その点からX軸に下ろした垂線の足の位置が、その点のX座標です。Y軸についても同様です。
- これらはθによって決まる値なので、それぞれ $\cos\theta$, $\sin\theta$ という関数で表すことができます。
- $\cos\theta$, $\sin\theta$ の具体的な値を求める計算は難しいのですが、Excelではそれらを計算する関数が用意されています。
- 今回は使いませんが、数学の世界では $\tan\theta$ も加えた3関数は、まとめて「三角関数」と呼ばれます。

なお、 $\tan\theta = \frac{\sin\theta}{\cos\theta} = \frac{y}{x}$ です。



数学演習1：半径1 (m)の円周のXY座標を求める

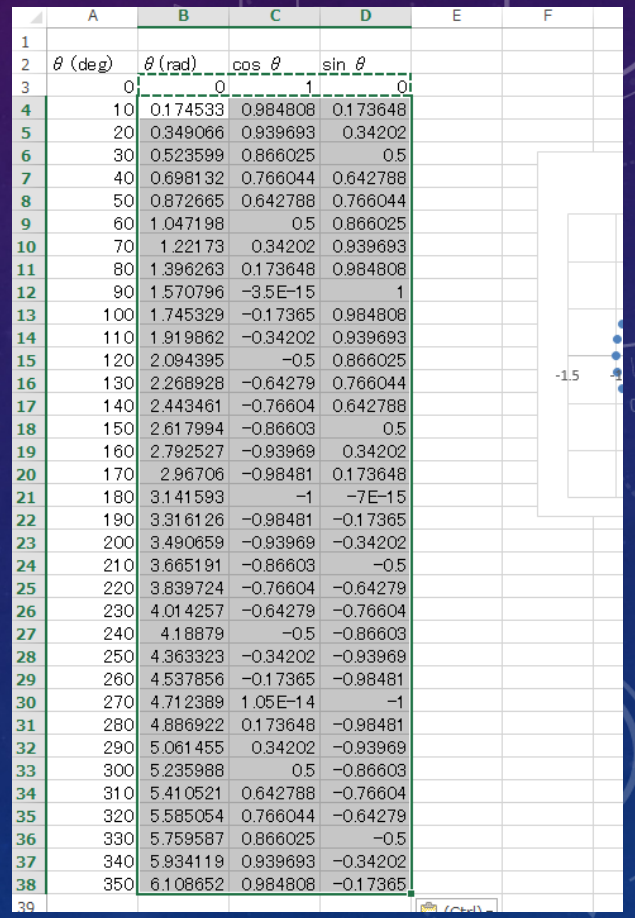
・ワークシートを「数学演習1」に切り替えてください。

・円周上の位置を示す角度θは、10度刻みで1周分書かれています。

・半径1なので、cosθとsinθは、そのままXY座標となります。

・θ＝0の点のみ、ラジアン値とXY座標を求める数式が入っています。数式の意味を確認してください。

・数式の部分を他の行にコピペすると、その行の値が計算され、グラフに表示されます。1周分やってみてください。



	A	B	C	D
2	θ (deg)	θ (rad)	cos θ	sin θ
3	0	0	1	0
4	10	0.174533	0.984808	0.173648
5	20	0.349066	0.939693	0.34202
6	30	0.523599	0.866025	0.5
7	40	0.698132	0.766044	0.642788
8	50	0.872665	0.642788	0.766044
9	60	1.047198	0.5	0.866025
10	70	1.22173	0.34202	0.939693
11	80	1.396263	0.173648	0.984808
12	90	1.570796	-3.5E-15	1
13	100	1.745329	-0.17365	0.984808
14	110	1.919862	-0.34202	0.939693
15	120	2.094395	-0.5	0.866025
16	130	2.268928	-0.64279	0.766044
17	140	2.443461	-0.76604	0.642788
18	150	2.617994	-0.86603	0.5
19	160	2.792527	-0.93969	0.34202
20	170	2.96706	-0.98481	0.173648
21	180	3.141593	-1	-7E-15
22	190	3.316126	-0.98481	-0.17365
23	200	3.490659	-0.93969	-0.34202
24	210	3.665191	-0.86603	-0.5
25	220	3.839724	-0.76604	-0.64279
26	230	4.014257	-0.64279	-0.76604
27	240	4.18879	-0.5	-0.86603
28	250	4.363323	-0.34202	-0.93969
29	260	4.537856	-0.17365	-0.98481
30	270	4.712389	1.05E-14	-1
31	280	4.886922	0.173648	-0.98481
32	290	5.061455	0.34202	-0.93969
33	300	5.235988	0.5	-0.86603
34	310	5.410521	0.642788	-0.76604
35	320	5.585054	0.766044	-0.64279
36	330	5.759587	0.866025	-0.5
37	340	5.934119	0.939693	-0.34202
38	350	6.108652	0.984808	-0.17365

数学演習2：半径6,378,137(m)の円周のXY座標を求める

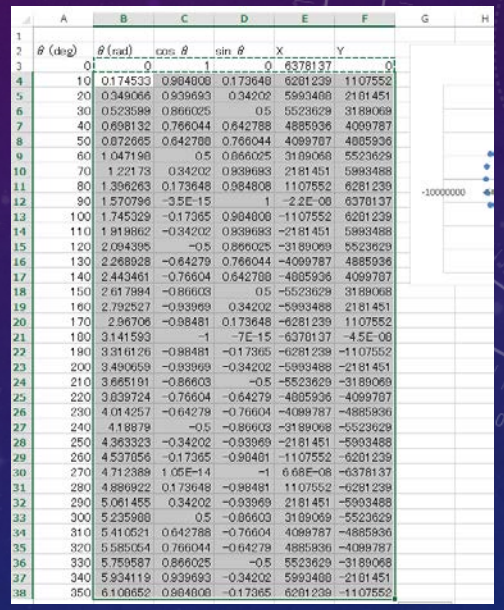
・ワークシートを「数学演習２」に切り替えてください。

・θ＝0の点のみ、ラジアン値とXY座標を求める数式が入れてあります。

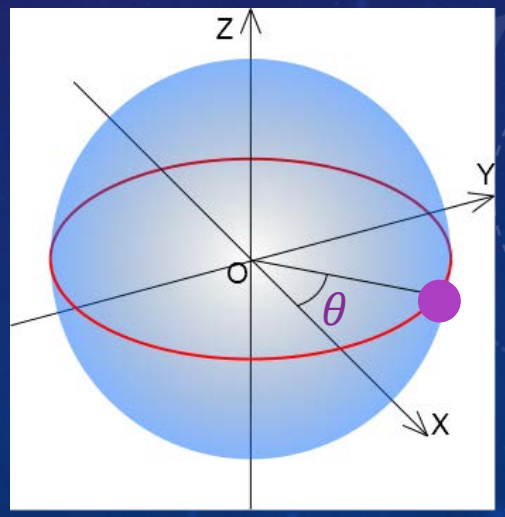
・半径6,378,137の円は、単に半径１の円を6,378,137倍に引き伸ばした形をしています。このことから、数式の意味を考えてください。

・数式の部分を他の行にコピペすると、その行の値が計算され、グラフに表示されます。１周分やってみてください。

・これで、右の図のように赤道上で経度が指定された点の座標が計算できたことになります。

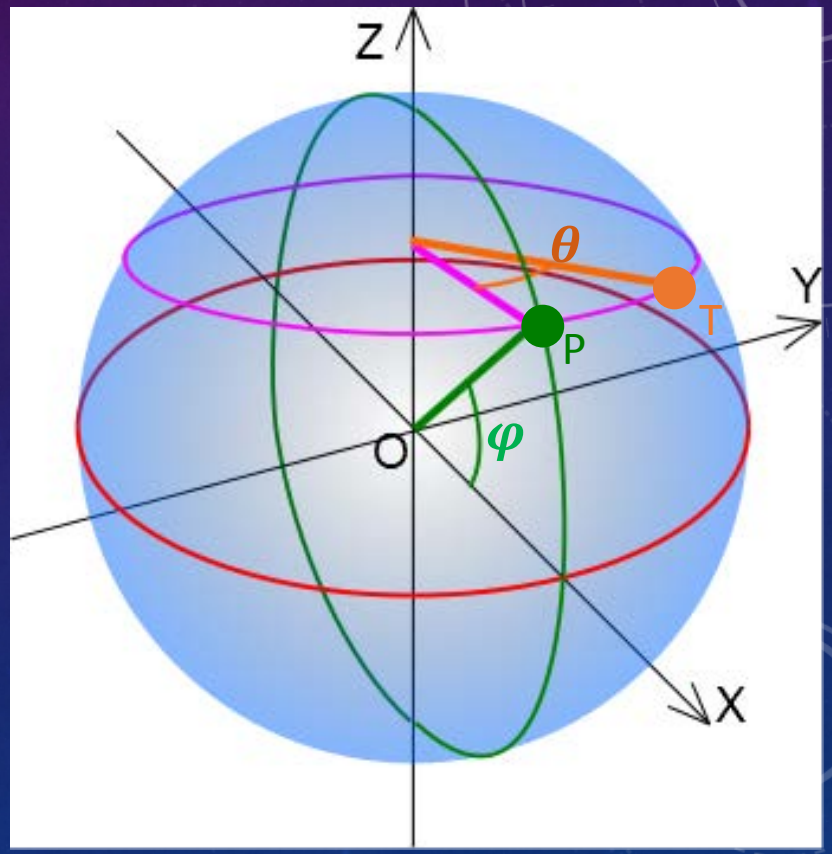


	A	B	C	D	E	F	G	H
1								
2	θ (deg)	θ (rad)	cos θ	sin θ	X	Y		
3	0	0	1	0	6378137	0		
4	10	0.174533	0.984808	0.173648	6281239	1107552		
5	20	0.349066	0.939693	0.34202	5993488	2181451		
6	30	0.523599	0.866025	0.5	5523629	3189068		
7	40	0.698132	0.766044	0.642788	4885936	4099787		
8	50	0.872665	0.642788	0.766044	4099787	4885936		
9	60	1.047198	0.5	0.866025	3189068	5523629		
10	70	1.22173	0.34202	0.939693	2181451	5993488		
11	80	1.396263	0.173648	0.984808	1107552	6281239		
12	90	1.570796	-3.5E-15	1	-2.2E-08	6378137		
13	100	1.745329	-0.17365	0.984808	-1107552	6281239		
14	110	1.919862	-0.34202	0.939693	-2181451	5993488		
15	120	2.094395	-0.5	0.866025	-3189069	5523629		
16	130	2.268928	-0.64279	0.766044	-4099787	4885936		
17	140	2.443461	-0.76604	0.642788	-4885936	4099787		
18	150	2.617994	-0.86603	0.5	-5523629	3189068		
19	160	2.792527	-0.93969	0.34202	-5993488	2181451		
20	170	2.96706	-0.98481	0.173648	-6281239	1107552		
21	180	3.141593	-1	-7E-15	-6378137	-4.5E-08		
22	190	3.316126	-0.98481	-0.17365	-6281239	-1107552		
23	200	3.490659	-0.93969	-0.34202	-5993488	-2181451		
24	210	3.665191	-0.86603	-0.5	-5523629	-3189069		
25	220	3.839724	-0.76604	-0.64279	-4885936	-4099787		
26	230	4.014257	-0.64279	-0.76604	-4099787	-4885936		
27	240	4.18879	-0.5	-0.86603	-3189068	-5523629		
28	250	4.363323	-0.34202	-0.93969	-2181451	-5993488		
29	260	4.537856	-0.17365	-0.98481	-1107552	-6281239		
30	270	4.712389	1.05E-14	-1	6.68E-08	-6378137		
31	280	4.886922	0.173648	-0.98481	1107552	-6281239		
32	290	5.061455	0.34202	-0.93969	2181451	-5993488		
33	300	5.235988	0.5	-0.86603	3189069	-5523629		
34	310	5.410521	0.642788	-0.76604	4099787	-4885936		
35	320	5.585054	0.766044	-0.64279	4885936	-4099787		
36	330	5.759587	0.866025	-0.5	5523629	-3189068		
37	340	5.934119	0.939693	-0.34202	5993488	-2181451		
38	350	6.108652	0.984808	-0.17365	6281239	-1107552		



数学演習3：経度・緯度からXYZ座標を求める

・経度0の子午線は、イギリスのグリニッジを通ります。
　この線の上で緯度φの点Pを想像しましょう。
・Pのところで地球を「水平」に切った断面は円となります。
　半径は地球の半径の$\cos\varphi$倍となります。
・この円（緯線）の上で経度θの点Tの位置を考えましょう。
　先ほどと同様の方法でXY座標が計算できます。
・点TのZ座標は、図から明らかなように、$\sin\varphi$となります。
・これが、経度・緯度が指定された地球上の位置をXYZ座標で
　表す方法の基本です。



※ GPSで使用されているIERS基準子午線は、グリニッジ天文台から100mほど東にずれています。

※ θ とか φ でどの角度を表すか、ということにはある程度慣例がありますが、慣例にとらわれすぎると他の慣例と整合性がとれなくなります。結局、何にどの記号を使うかを、きちんと定義するのが基本です。

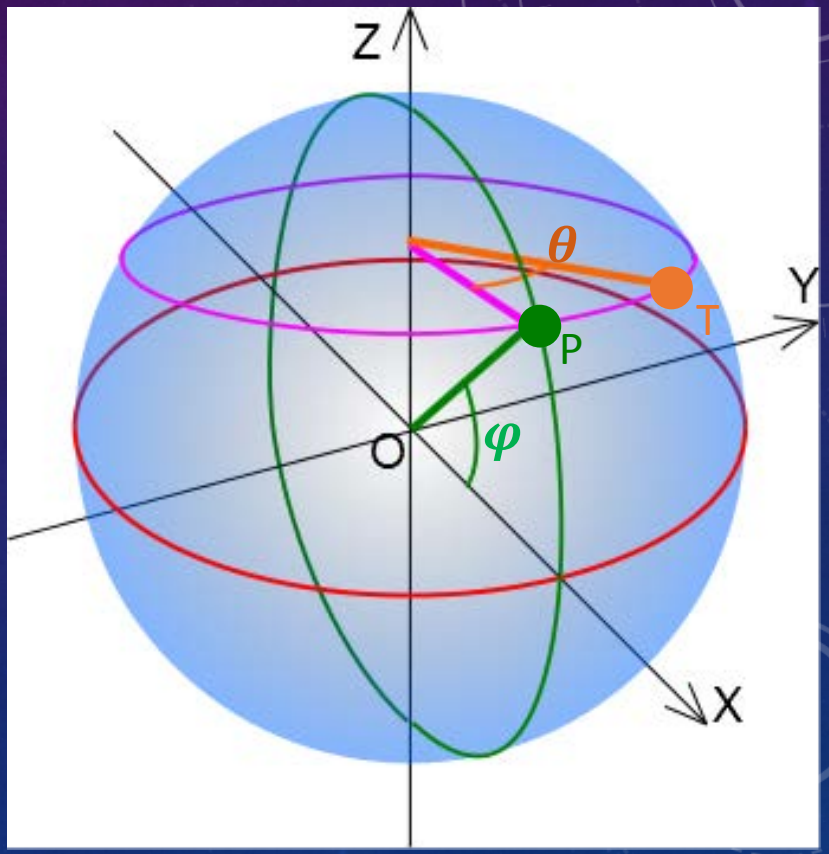
数学演習3：経度・緯度からXYZ座標を求める

・ワークシートを「数学演習3」に切り替えてください。

・経度θ＝0、緯度φ＝0の点のみ、ラジアン値とXYZ座標を求める
　数式が入れてあります。

・数式の部分を他の行にコピペすると、その行の値が計算され、
　グラフに表示されます。北緯80度までの行を埋めてみてください。

	A	B	C	D	E	F	G	H
1								
2	経度(deg)	緯度(deg)	経度(rad)	緯度(rad)	緯度線の半径(m)	X	Y	Z
3	0	0	0	0	6378137	6378137	0	0
4	30	0	0.523599	0	6378137	5523629	3189069	0
5	60	0	1.047198	0	6378137	3189068	5523629	0
6	90	0	1.570796	0	6378137	-2.2E-08	6378137	0
7	120	0	2.094395	0	6378137	-3189069	5523629	0
8	150	0	2.617994	0	6378137	-5523629	3189068	0
9	180	0	3.141593	0	6378137	-6378137	-4.5E-08	0
10	210	0	3.665191	0	6378137	-5523629	-3189069	0
11	240	0	4.18879	0	6378137	-3189068	-5523629	0
12	270	0	4.712389	0	6378137	6.68E-08	-6378137	0
13	300	0	5.235988	0	6378137	3189069	-5523629	0
14	330	0	5.759587	0	6378137	5523629	-3189068	0
15	0	20	0	0.349066	5993488	5993488	0	2049894
16	30	20	0.523599	0.349066	5993488	5190513	2996744	2049894
17	60	20	1.047198	0.349066	5993488	2996744	5190513	2049894

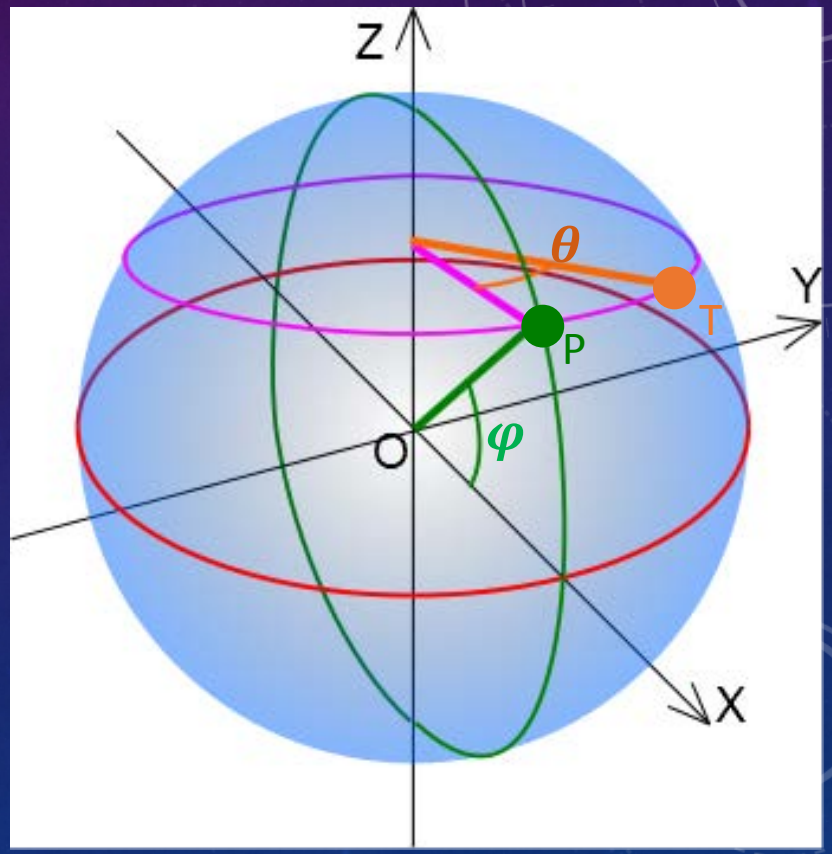


※この方法で、それらしいXYZ座標値が出てきますが、
　あまり実用にはなりません。理由はこの後説明します。

3　測地系の基礎

・数学演習3の方法で、地球上の経度・緯度からXYZ座標を
　得ることができます。
　この座標系（座標の取り方）は
　ECEF（地球中心・地球固定直交座標系）と呼ばれます。

・地球中心が原点で、X軸は経度0度の子午線と赤道の交点を
　通ります。
・Y軸は東経90度の子午線と赤道の交点を通り、Z軸は北極を
　通ります。



しかし・・・

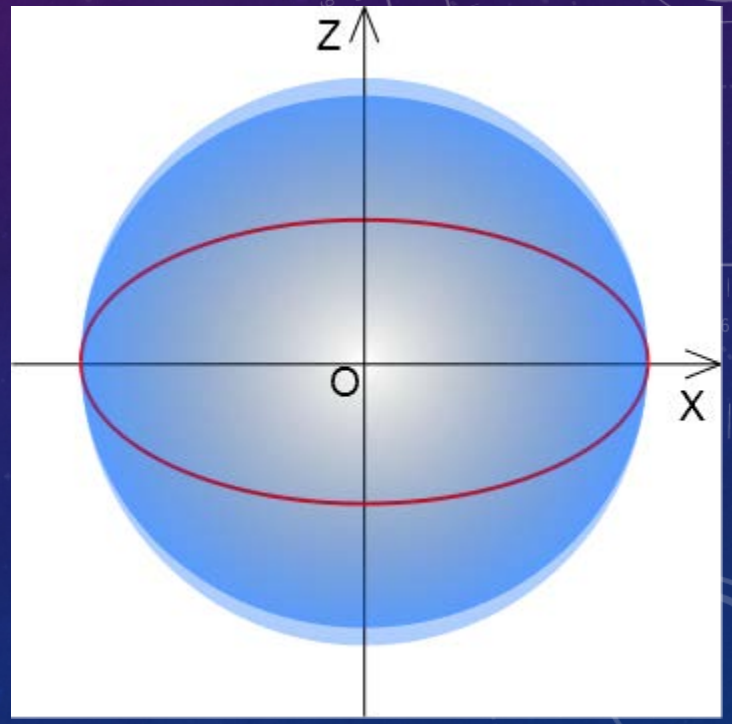
3　測地系の基礎

・この計算方法のままでは、大きなずれが生じてしまいます。
　その理由は下記の通りです。

　＊地球は球体がZ軸方向に約0.3%つぶれたような形をしています。
　　(数十km規模のずれ。)

・これについては、WGS84（米国防総省が定めた世界測地系であり
　GPSで採用）で定められた「地球楕円体」の形状を前提として計算
　すれば補正できます。

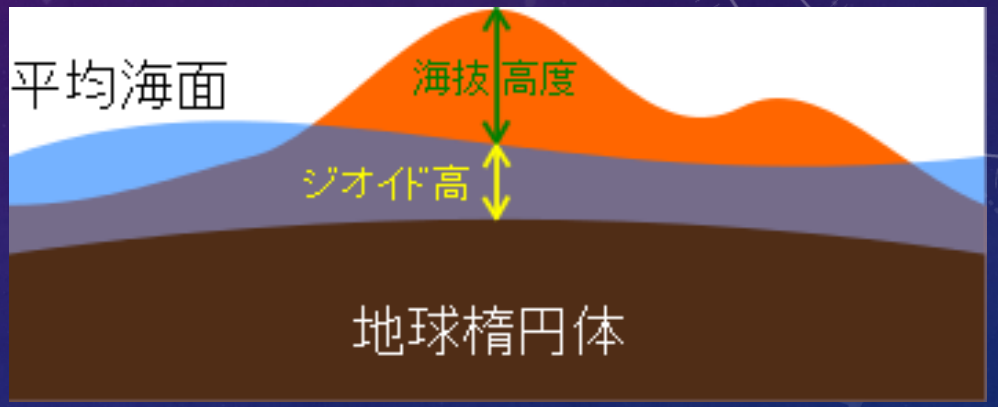
・これで地球楕円体表面の座標が得られますが、GPSの高さ方向の
　データを反映させるにはどうしたらよいか考えてみましょう。



3　測地系の基礎

・高さのデータは、GGAレコードの海抜高度（平均的な海面からの高さ）から得られます。
しかし、海抜高度は地球楕円体からの高さではありません。その主な理由は下記の通りです。

＊地形や地下の岩石等の密度分布により、地表の重力が変動するので、ジオイド高（平均的な海面高度）が場所によって変わります。(数十m規模の変動)



・これを補正する手段として、EGM96（米国防総省による）などのジオイド高の分布モデルが存在します。
今回使用しているGPSモジュールは、そのようなデータまたは計算式を内蔵しており、
GGAレコードに含めて出力しています。これを使って、
ジオイド高さ＋海抜高度＝地球楕円体からの高さ　という計算をすればよいことになります。

4　座標変換（その1）

・測地系の基礎を踏まえて、ECEFのXYZ座標を計算することを目的とします。

・ワークシートを「GPGGAデータ」に切り替えてください。
・ソースコードはすでにある(test2(),test3())ので、よく読んでその意味を理解できるようがんばってください。

・今回使用するサンプルデータは"GPSsample2.txt"です。
前回使った”GPSsample.txt”も、時間に余裕がある人は使ってみてください。

・まずは、メモ帳でGGAレコードを見ましょう。9番目のフィールドが海抜高度、
11番目のフィールドがジオイド高です。（レコード内の1個ずつのデータを「フィールド」と呼びます。）

例題2：海抜高度とジオイド高さの読み取り

・test2()を実行し"GPSsample2.txt"を読み込んでください。

・ソースコードを見て、海抜高度とジオイド高をどのように読み取っているか、どのように処理しているかを確認してください。

```
'測定場所の海抜高度と、WGS84準拠楕円体表面からのジオイド高度（海面の高さ）を出力する
Sub test2()
    Dim fn As String, buf As String
    Dim c As Long
    Dim spBuf As Variant
    Dim tmpd As Double

    '使用中のセルをすべて消去する
    ClearCells 2, 1, LastRow, LastColumn

    'xlsmファイルと同じフォルダをカレントフォルダにして、
    'ユーザーにファイルを選択させる
    ChDrive ThisWorkbook.Path
    ChDir ThisWorkbook.Path
    fn = Application.GetOpenFilename

    'ファイルを1番に割り当てて開く（後で必ず閉じる）
    Open fn For Input As #1

    'ファイルの終端に達するまで繰り返す
    c = 2

    Dim cGGA As Long, startSec As Double, prevSec As Double, curSec As Double, curDay As Long

    cGGA = 0
    curDay = 0

    Do Until EOF(1)
        Line Input #1, buf  '1行読み込む
        If Left(buf, 1) <> "#" Then

            'Variant型の配列を作り、1行をカンマで
            '区切って分割して入れる
            spBuf = Split(buf, ",")

            '行頭に"$GPGGA"があったら、データをセルに入れる
            If spBuf(0) = "$GPGGA" Then
                If spBuf(2) <> "" Then

                    '最初のGGAデータかどうか調べる
                    If cGGA = 0 Then
                        '最初の時刻を記録する
                        startSec = ConvGGATimeToSec(CStr(spBuf(1)))
                        '1つ前の時刻も現在時刻も同じということにしておく
                        prevSec = startSec
                        curSec = startSec
                    Else
                        '1つ前のGGAデータを退避させる
```

例題3：ECEFでのXYZ座標の計算

・test3()を実行して"GPSsample2.txt" を読み込み、XYZ座標が出てくることを確認してください。

・海抜高度とジオイド高さの和が、地球楕円体からの高さです。

・WGS84に基づく計算は複雑なので、そのための「クラス」を用意してあります。

・クラスを使うときは、必ず下記のような初期化処理が必要です。

Set [クラス変数名] = new [クラス名]([初期化時の引数])

・WGS84クラスのメンバプロシージャをどのように使っているかに注目してください。

次は、このXYZ座標をグラフにしてみたいのですが・・・

```
'WGS84に基づく地球中心・地球固定直交座標系(ECEF)のXYZ座標に変換する
Sub test3()
    Dim fn As String, buf As String
    Dim c As Long
    Dim spBuf As Variant
    Dim tmpd As Double

    '使用中のセルをすべて消去する
    ClearCells 2, 1, LastRow, LastColumn

    'xlsmファイルと同じフォルダをカレントフォルダにして、
    'ユーザーにファイルを選択させる
    ChDrive ThisWorkbook.Path
    ChDir ThisWorkbook.Path
    fn = Application.GetOpenFilename

    'ファイルを1番に割り当てて開く（後で必ず閉じる）
    Open fn For Input As #1

    'ファイルの終端に達するまで繰り返す
    c = 2

    Dim cGGA As Long, startSec As Double, prevSec As Double, curSec As Double, curDay As Long
    Dim longD As Double, latD As Double, altD As Double
    Dim clWGS84 As WGS84, posV(3) As Double

    cGGA = 0
    curDay = 0
    Set clWGS84 = New WGS84

    Do Until EOF(1)
        Line Input #1, buf  '1行読み込む
        If Left(buf, 1) <> "#" Then

            'Variant型の配列を作り、1行をカンマで
            '区切って分割して入れる
            spBuf = Split(buf, ",")

            '行頭に"$GPGGA"があったら、データをセルに入れる
            If spBuf(0) = "$GPGGA" Then
                If spBuf(2) <> "" Then

                    '経度（東経は正、西経は負）
                    tmpd = convDMMtoDeg(CStr(spBuf(4)))
                    If spBuf(5) = "E" Then
                        longD = tmpd
```

```
Option Explicit

Private Const cA = 6378137
Private Const cF = 1 / 298.257223563
Private Const cPI = 3.1415926535898

Private esq As Double
Private localRotMat As Matrix3d
Private localPosV(2) As Double

'クラスの初期化
Public Sub Class_Initialize()
    esq = 2 * cF - cF * cF
    Set localRotMat = New Matrix3d
End Sub


'経度、緯度、高度からWGS84に基づく地球中心・地球固定直交座標系(ECEF)のXYZ座標を求める
Public Sub GetECEF3d(longD As Double, latD As Double, altD As Double, ByRef p() As Double)
    Dim coeN As Double, longRad As Double, latRad As Double

    longRad = DegToRad(longD)
    latRad = DegToRad(latD)
    coeN = cA / Math.Sqr(1 - esq * Math.Sin(latRad))
    p(0) = (coeN + altD) * Math.Cos(latRad) * Math.Cos(longRad)
    p(1) = (coeN + altD) * Math.Cos(latRad) * Math.Sin(longRad)
    p(2) = (coeN * (1 - esq) + altD) * Math.Sin(latRad)

End Sub

'地平座標の回転行列と原点をセットする
Public Sub SetLocalOrigin(longD As Double, latD As Double, altD As Double)
    Dim posV(2) As Double

    GetECEF3d longD, latD, altD, posV
    '指定された位置で回転行列を初期化する
    localRotMat.LoadRotation 2, DegToRad(longD)
    localRotMat.MulRotation 1, DegToRad(90 - latD)
    localRotMat.MulRotation 2, DegToRad(90)
    '指定された位置を原点とする
    localPosV(0) = posV(0)
    localPosV(1) = posV(1)
    localPosV(2) = posV(2)

    localRotMat.Transform3d localPosV

End Sub

'ECEF座標から地平座標へ変換する
Public Sub ConvECEFtoLocal(ByRef ecefPosV() As Double)
    localRotMat.Transform3d ecefPosV
    ecefPosV(0) = ecefPosV(0) - localPosV(0)
```

課題2：ECEFでのXYZ座標のファイル出力

・Excelでは、残念ながらXYZの三次元の方向を持つグラフ「三次元プロット」を作ることができません。

・この分野でよく使われるフリーソフトとして"gnuplot"（グヌープロット）があるので、今回はそれを使います。

・gnuplotで読み込みやすいように、すべてのXYZ座標を下記の形式でファイル出力するプロシージャを作ります。

```
[1番目のX座標値] [1番目のY座標値] [1番目のZ座標値]
[2番目のX座標値] [2番目のY座標値] [2番目のZ座標値]
        ・
        ・
        ・
```

・出力ファイル名は"ecefXYZ.txt"とします。

・kadai2()のソースコードが途中まで作ってあるので、それを完成させて、実行してください。

(ヒント：Print #1, ...)

＜制限時間　5分＞

課題2の解答

・For～Nextループの中で、下記のように処理をすればOKです。

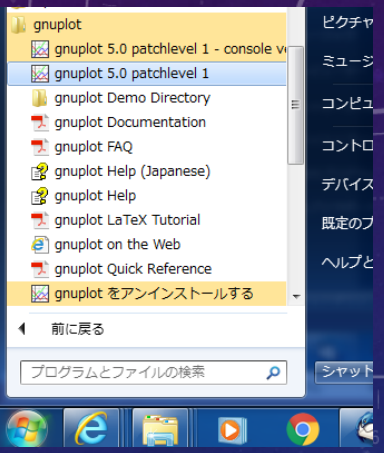
```
    'ファイルを1番に割り当てて開く（後で必ず閉じる）
    Open fn For Output As #1

    'データの終端に達するまで繰り返す
    For i = 2 To LastRow
        Print #1, CStr(Cells(i, 6)) + " " + CStr(Cells(i, 7)) + " " + CStr(Cells(i, 8))
    Next

    'ファイルを閉じる
    Close #1
```

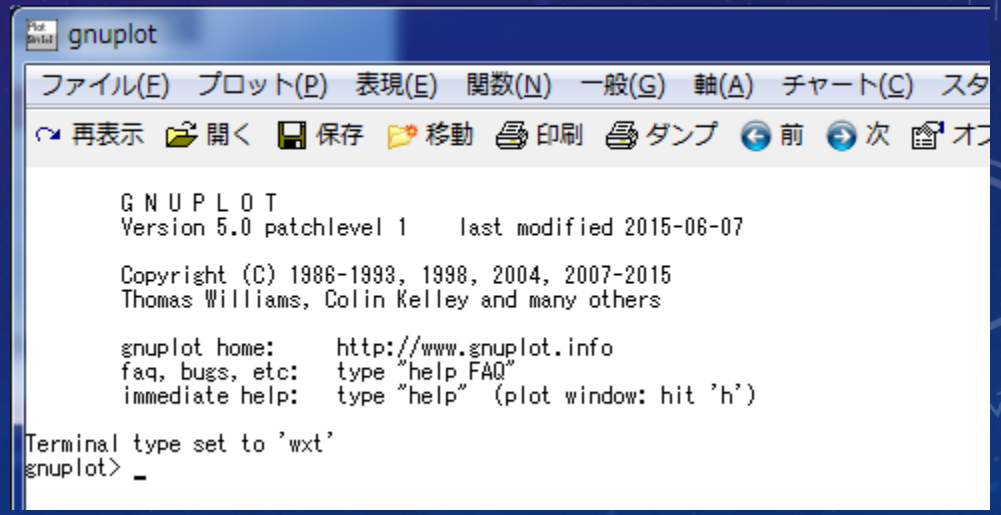
5　地球規模での三次元プロット

・スタートメニューから「gnuplot 5.0 patchlevel 1」を起動してください。

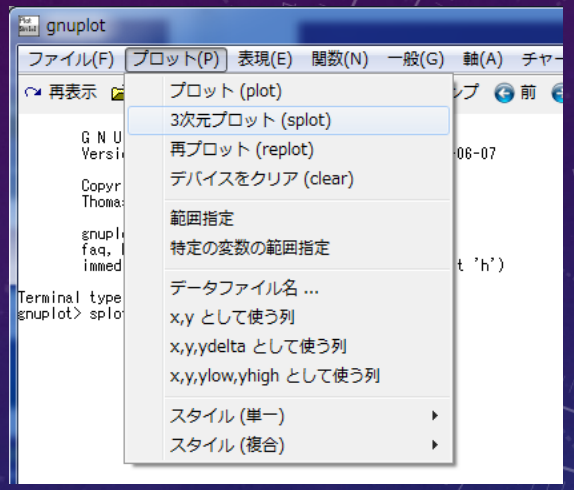


・コンソール（コンピュータと文字で対話するための画面）が開きます。
　この画面でコマンド（命令文）を打ち込んで
　グラフを描くのが基本的な使い方です。

・コマンドを覚えるのは大変なので、ある程度
　自動化するためのメニューが備えられています。

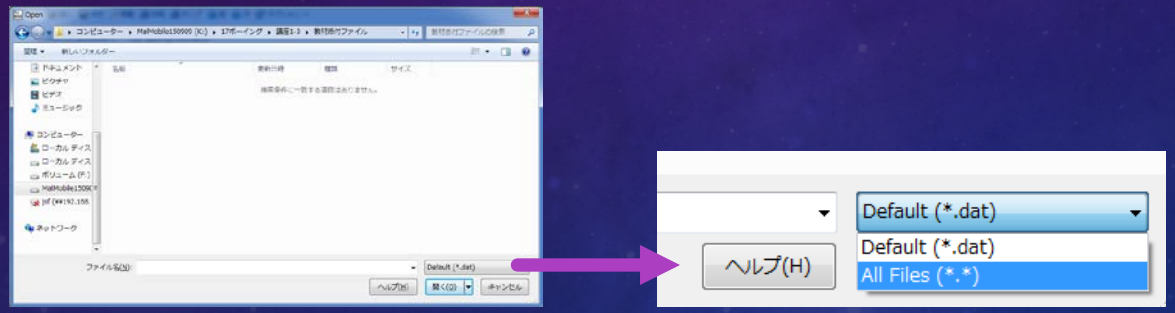


5　地球規模での三次元プロット



・「プロット」から「三次元プロット」を選択し、コンソールに"splot"が表示されることを確認してください。

・「プロット」から「データファイル名...」を選択すると、ファイル選択画面がでます。右下の拡張子フィルタを「All Files (*.*)」に変更してください。



これで、拡張子がtxtのファイルも表示されるようになります。

・課題2で保存した"ecefXYZ.txt"を選択すると、そのファイル名がコンソールに表示されます。

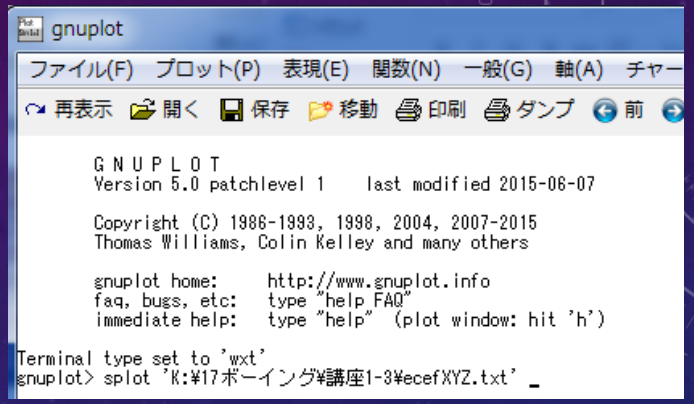
5　地球規模での三次元プロット

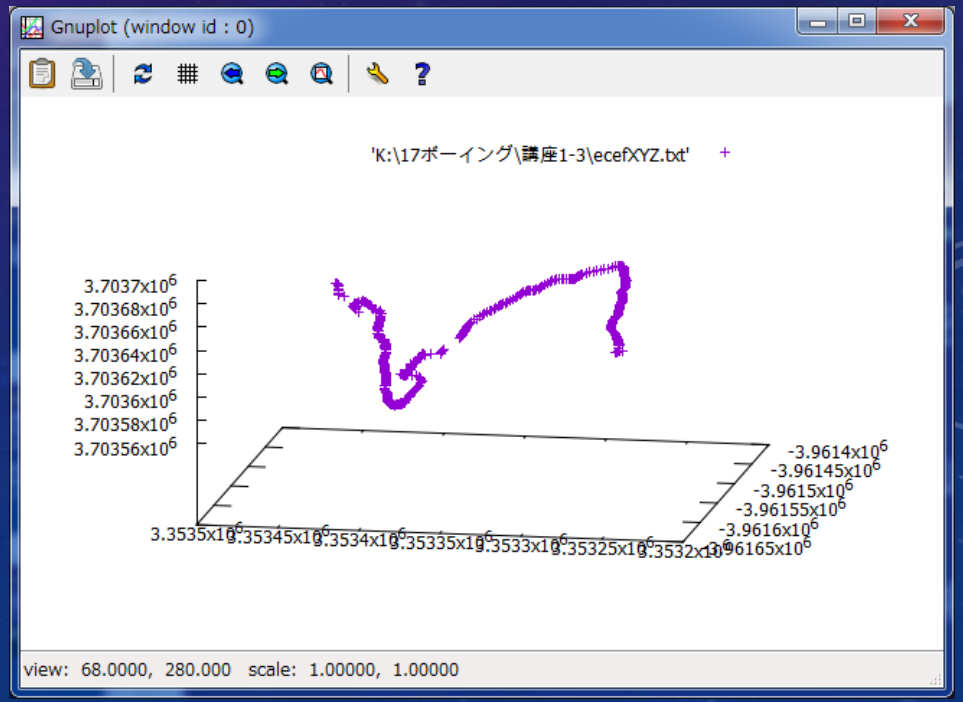
・コンソールでEnterキーを押すとグラフが出てきます。

・このグラフは、マウスで回転させることができます。

・形状を確認してください。地球表面が傾いているので、
　データが傾いています。

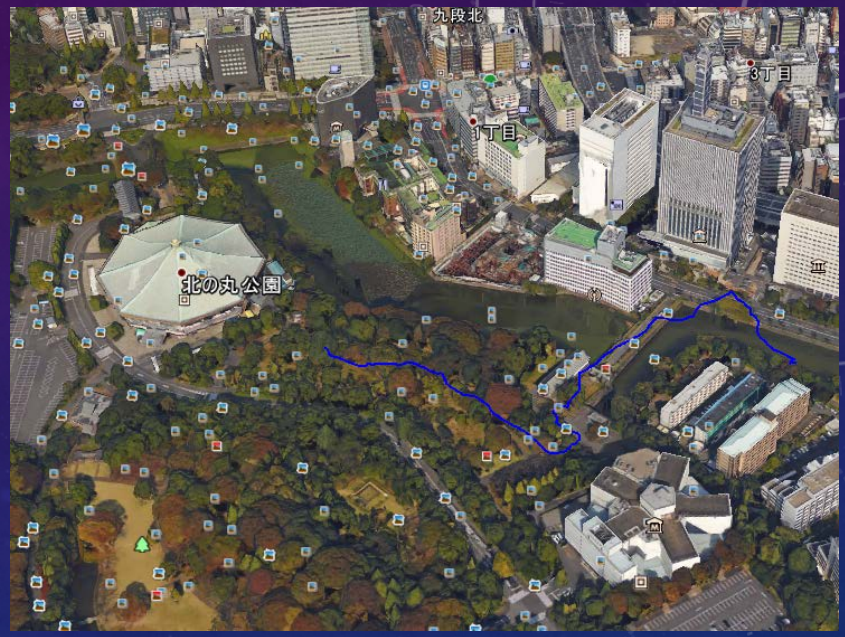
・GPSsample.txtのプロットも見てみましょう。データを
　変えるにはtest3()からの手順を繰り返してください。



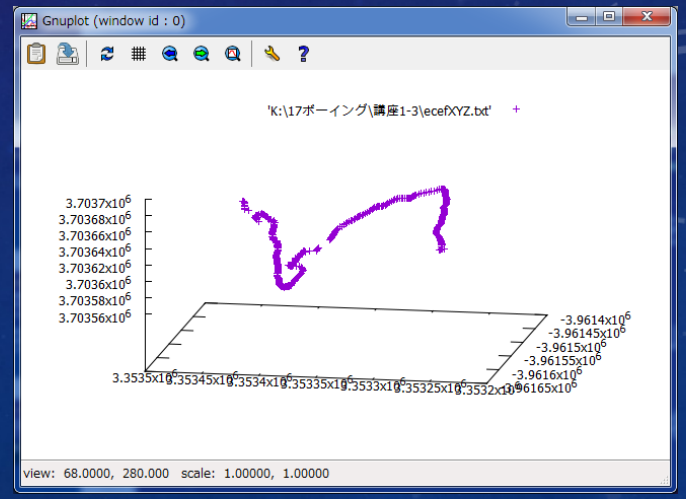


5　地球規模での三次元プロット

・GPSsample2.txtのデータは、GoogleEarthで見るとこうなります。
実際の地形と三次元プロットの結果は、あまり一致していません。



・GPSから得られる高度の値はかなり精度が低く、10m程度は簡単にずれてしまいます。
その理由についても考えてみてください。



6　座標変換（その2）

・場所によって地面の傾きが違うので、ECEFの三次元プロットでは、極付近でない限り直感的にとらえにくくなってしまいます。

・直感的にとらえやすい座標系の例は、下記のようなものです。

　　X軸は、東西方向で、東を正とする。

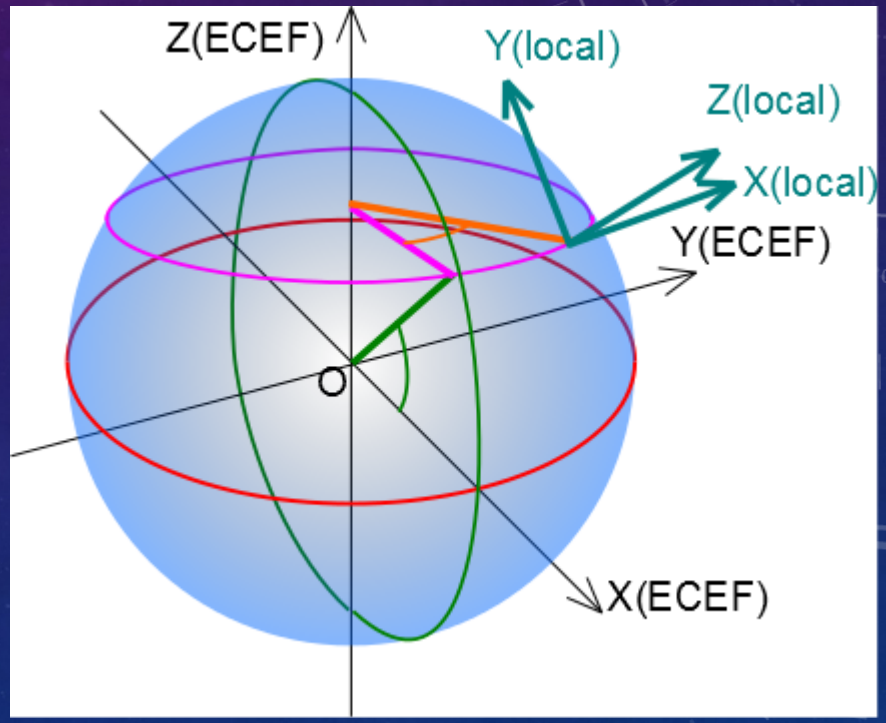
　　Y軸は、南北方向で、北を正とする。

　　Z軸は、高さ方向で、上を正とする。

　このような座標系を「地平座標系」と呼びます。

　（右図では"local"と表記しています）

　この座標系になれば、下記のことが直感的に理解できます。



*CanSatが打ち上げ地点から何m上昇してから分離されたか

*東西南北に、どのように移動しながら降下したか

※ただし地球は丸いので、この座標系ではあまり大きな距離は扱えません。

例題4：地平座標系でのXYZ座標の計算

- test4()を実行して、先ほどとは別のXYZ座標が出てくることを確認してください。
- この座標系では、最初のGGAレコードが示す地点を原点とします。それをどのように処理しているか、ソースコードをよく見て理解してください。
- ECEFからの座標軸の傾きを打ち消す計算に「行列」を使っています。その計算は複雑なので行列計算を扱うクラス"Matrix3d"を用意してあり、WGS84クラスの中で使っています。

（なお行列を使わないと、さらに複雑になります。）

```
'WGS84に基づく地球中心・地球固定直交座標系(ECEF)のXYZ座標を経由して地平座標に変換する
Sub test4()
    Dim fn As String, buf As String
    Dim c As Long
    Dim spBuf As Variant
    Dim tmpd As Double

    '使用中のセルをすべて消去する
    ClearCells 2, 1, LastRow, LastColumn

    'xlsmファイルと同じフォルダをカレントフォルダにして、
    'ユーザーにファイルを選択させる
    ChDrive ThisWorkbook.Path
    ChDir ThisWorkbook.Path
    fn = Application.GetOpenFilename

    'ファイルを1番に割り当てて開く（後で必ず閉じる）
    Open fn For Input As #1

    'ファイルの終端に達するまで繰り返す
    c = 2

    Dim cGGA As Long, startSec As Double, prevSec As Double, curSec As Double, curDay As Long
    Dim longD As Double, latD As Double, altD As Double
    Dim clWGS84 As WGS84, posV(3) As Double

    cGGA = 0
    curDay = 0
    Set clWGS84 = New WGS84

    Do Until EOF(1)
        Line Input #1, buf  '1行読み込む
        If Left(buf, 1) <> "#" Then

            'Variant型の配列を作り、1行をカンマで
            '区切って分割して入れる
            spBuf = Split(buf, ",")

            '行頭に"$GPGGA"があったら、データをセルに入れる
            If spBuf(0) = "$GPGGA" Then
                If spBuf(2) <> "" Then

                    '経度（東経は正、西経は負）
                    tmpd = convDMMtoDeg(CStr(spBuf(4)))
                    If spBuf(5) = "E" Then
                        longD = tmpd
```

課題3：地平座標系でのXYZ座標のファイル出力

・課題2と同様の方法で、地平座標系のXYZ座標を下記の形式でファイル出力しましょう。

・kadai2をコピペしてkadai3を作れば、あとは出力元のセル指定と出力ファイル名を変更するだけです。

・ファイル名は“localXYZ.txt"とします。

```
［1番目のX座標値］［1番目のY座標値］［1番目のZ座標値］
［2番目のX座標値］［2番目のY座標値］［2番目のZ座標値］
        ・
        ・
        ・
```

＜制限時間　5分＞

課題3の解答

・kadai2()をコピペして、For～Nextループの中のPrintに使っているデータを、
　下記のように変更すればOKです。

```
'地平座標を"X Y Z"の形式で保存する
Sub kadai3()

    Dim fn As String, buf As String, i As Long

    ChDrive ThisWorkbook.Path
    ChDir ThisWorkbook.Path
    fn = ThisWorkbook.Path + "¥localXYZ.txt"

    'ファイルを1番に割り当てて開く（後で必ず閉じる）
    Open fn For Output As #1

    'データの終端に達するまで繰り返す
    For i = 2 To LastRow
        Print #1, CStr(Cells(i, 9)) + " " + CStr(Cells(i, 10)) + " " + CStr(Cells(i, 11))
    Next

    'ファイルを閉じる
    Close #1

End Sub
```

7　地域規模での三次元プロット

・gnuplotが起動していない場合は、もう一度起動してください。

・メニューバーから「三次元プロット」を選択し、コンソールに"splot"が表示されることを確認してください。

・メニューバーから「データファイル名…」を選択し、課題2で保存した"ecefXYZ.txt"を選択し、コンソールでEnterキーを押してください。

・地球表面の傾きはなくなっていて、東西・南北・高さ方向が座標軸に一致していることを確認してください。

まとめ

・今回は、時刻と共に変化する位置情報をどのように可視化するか、というテーマで一つの例を体験していただきました。

・単純に「位置」といっても、地球規模で考えると意外と考えることが多く、複雑な処理が必要となることがわかると思います。

・この第１期の講座では、VBAで可能になることの、ほんの一部を扱ったに過ぎません。
他のOffice製品やOS、そして様々な外部ライブラリの機能（音声・映像・USB機器など）を呼び出し、使いこなすことで、市販できるようなレベルのソフトも作ることができます。
（個別の顧客向けに、業務システムを丸ごとVBAで作って納品するケースもあります）

まとめ

・CanSatを作るという目標までは、まだまだ長い道のりがあります。
（本物の人工衛星を作るのに比べれば、かなり短いですが）

・皆さんは今後、ソフトウェア開発技術の別の用途に目覚めるかもしれません。
数をこなすことが大事なので、科学計算でもゲーム作成でも、とにかくやってみてください。

・ソフトウェア開発技術のうち、特にプログラミング（コーディング）で必要な考え方は、
言語が違っても大体一緒です。
講座に参加した方は、C/C++, C#, Java, PHP等も、今後あまり苦労せず学ぶことができるでしょう。

・どのようなことでも、今期の講座の内容がいつか役に立つことを願います。

＜おわり＞