

Car Crash Course

CAR(Computer-Assisted Reporting)の短期集中講座(crash course)として、地図公示などの公的データを素材にして表計算やGISを体験する。

国土数値情報 ダウンロードサービス

「国土数値情報」とは、国土に関する基礎的な空間データセットです
Google™カスタム検索
検索

Web API

「国土数値情報」は、国土形成計画、国土利用計画等の策定や国土政策の推進に資するため、地形、土地利用、公共施設など国土に関する基礎的な情報をGISデータとして整備したもので、現時点においては非営利・非商用でご利用いただけます。

ただし、「国土数値情報」は、概ね国土地理院の2万5000分の1地形図（許容誤差：10m超）をベースに作成されていることから、精度上、建物の判断やナビゲーションなどには適していません。また、更新頻度が高くなっているデータもあります。データの整備年次やライセンスに留意して使用いただくとともに、最新の情報が必要な場合はそれぞれ別途原典の資料で確認するようしてください。

■重大な警告 ■ 本サイトで提供されているデータ（データセットやリソースの説明、API利用等）は、ジャンプしてきたリンク先やフレーム外の記載によらず、「国土数値情報」の利用規約等の本ホームページの記載のみに依拠し、それ以外の記載は全て無効です。ご注意ください。

[初めての方へ](#)

[サイトマップ](#)

[用語集](#)

[よくある質問（FAQ）](#)

データ形式	JPGIS2.1	JPGIS1.0	統一フォーマット (SHP・GML)	統一フォーマット (CSV)
<水域>	行政区域	<施設>	高速道路時系列	
海岸線	DID人口集中地区	国・都道府県の機関	緊急輸送道路	
海岸保全施設	中学校区	市町村役場等及び公的集会施設	道路密度・道路延長メッシュ	
湖沼	医療圏	市区町村役場	バスルート	
河川	景観計画区域	公共施設	バス停留所	
新規		警察署		

基本情報

- [局の概要・組織図\(PDF\)](#)
- [計画・基本方針一覧](#)
- [報道発表資料](#)
- [所管法令](#)
- [所管審議会等](#)
- [予算\(省全体リンク\)](#)
- [事業評価](#)
- [イベント情報](#)
- [アーカイブ\(過去の情報\)](#)

注意

- virtualBox上のUbuntuを使う（Windows/Macでも、基本的には同じことができる）
- 素材データを抜いてきたり、生データを作成する上流部門が一番重要
- 可視化やインターラクティブコンテンツを作成する下流部門が花形

準備体操

端末を開き、コマンドラインの練習をする。

file:///Users/isao/DJN/_build/html/tutorial.html

1/59 ページ

場所

.	現在位置
..	一つ上
~	ホーム

絶対パス: /home/usr/test.txt

相対パス: test.txt

操作

mkdir	make directory
rmdir	remove directory
mv	move/rename
cp	copy
rm	remove

移動

pwd	現在位置
cd	change directory
ls	ファイルリスト

閲覧

cat	表示
head	最初だけ -n 行数
tail	最後だけ -n 行数

echoは鸚鵡返し、catをファイル表示（複数可）。リダイレクトは出力先の変更。

* : ワイルドカード 例) find *.txt

> : リダイレクト 例) echo "Hello" > test.txt

iconv -f encoding -t encoding filename

例) iconv -f SHIFT-JIS -t UTF8 test.txt > test_utf8.txt

<https://ja.wikipedia.org/wiki/UNIXユーティリティーの一覧>

コマンドの種類は多く、それぞれ把握できないほど多機能だが、ここでは「少数のコマンドを組み合わせれば、大抵の場合の用が足りる」 Unixの思想を体感する。

データの入手

[国土数値情報ダウンロードサービス](#) から「統一フォーマット CSV」を選び、「地価公示」を選択して全国のデータ(平成27年)をダウンロードする。

[国土数値情報ダウンロードサービス](#) ファイル選択

該当するファイルは以下のとおりです。

ダウンロードしたいファイルをチェックし、選択ボタンを押してください。

ファイルの詳細は、ファイルをクリックすると表示されます。

ファイル名	ファイル容量 (MB)	年度	版数	データ項目	地域	説明	備考
<input type="checkbox"/> L01-19P-48-01.0a.zip	2.56MB	平成19年	1.0a	地価公示（点）	全国	世界測地系	メタデータ、形式
<input type="checkbox"/> L01-20P-48-01.0a.zip	2.72MB	平成20年	1.0a	地価公示（点）	全国	世界測地系	メタデータ、形式
<input type="checkbox"/> L01-21P-48-01.0a.zip	2.7MB	平成21年	1.0a	地価公示（点）	全国	世界測地系	メタデータ、形式
<input type="checkbox"/> L01-22P-48-01.0a.zip	2.72MB	平成22年	1.0a	地価公示（点）	全国	世界測地系	メタデータ、形式
<input type="checkbox"/> L01-23P-48-01.0a.zip	2.60MB	平成23年	1.0a	地価公示（点）	全国	世界測地系	メタデータ、形式
<input type="checkbox"/> L01-24P-48-01.0a.zip	2.71MB	平成24年	1.0a	地価公示（点）	全国	世界測地系	メタデータ、形式
<input type="checkbox"/> L01-25P-48-01.0a.zip	2.67MB	平成25年	1.0a	地価公示（点）	全国	世界測地系	メタデータ、形式
<input type="checkbox"/> L01-26P-48-01.0a.zip	2.86MB	平成26年	1.0a	地価公示（点）	全国	世界測地系	メタデータ、形式
<input type="checkbox"/> L01-27P-48-01.0a.zip	3.05MB	平成27年	1.0a	地価公示（点）	全国	世界測地系	メタデータ、形式

選 択

取 消

戻 る

(平成27年分がL01/L01-27P-12-01.0aにある)

表計算ソフト

データ入手して最初にすることは、どのような形式で保存されているかというフォーマット（書式）の確認だ。

この場合は幸いにもformat_L01_v2.3.htmlというファイルが付属しているが、業界標準のフォーマットを採用している場合は説明ファイルがないこともあり、その際は自分で調べなければならない。

国土数値情報 地価公示(L01-**P-2K_@@) データ形式(第2.3版)

データは、csv(コンマ区切りのテキストデータ) 形式です。**にはデータの調査年（和暦の数字部分）が2桁で入ります。また@@には都道府県コードが入ります。（例：北海道の平成5年データ：L01-05P-2K_01.csv）

データのフォーマットは、平成14年度決定
平成16年度一部修正（第1.0版）
平成19年度一部修正・変更（第2.0版）
平成26年度一部修正・変更（第2.2版）
平成27年度一部修正・変更（第2.3版）

項目	内容
経度（X座標）	単位[秒]（例：501376.613 = 139.2712816° = 139°16'16.613"）
緯度（Y座標）	座標値は世界測地系
所在地コード	第一法規出版「全国地方公共団体コード」
用途	000：住宅地 003：宅地見込地 005：商業地 007：準工業地 009：工業地 010：市街化調整区域内の現況宅地 013：市街化調整区域内の現況林地
連番	一連番号
年次	西暦、最終選定年次
前	前年所在地コード 第一法規出版「全国地方公共団体コード」 000：住宅地

試しに、csvファイルをエディタで開いてみると、以下のようなだった。

このように、文字化けしても驚いてはいけない。

計算機と文字コード

コンピューターは基本的に数値を計算し、数値を保存する機械だった。ところが、ある時、誰かは知らないが、1ならa、2ならb、のように数字に文字を対応させれば、文章も保存できることに気がついた。その対応表がASCII（アスキー、American Standard Code for Information Interchange, 1963）と呼ばれる約束事になった。アメリカ人は欧文しか考えていなかったので、8bit(0-256)で1文字を表すことにし、8bitのことを1Byteと呼ぶことにした。(正確には7bitで8bit目は予備)

デジタルは数字

```
abc.txt
1 abc
2 123
3 日本語
```

$$\text{"a"} = 0x61 = 97$$

$$\text{"b"} = 0x62 = 98$$

$$\text{"1"} = 0x31 = 49$$

$$\begin{aligned}\text{"語"} &= 0xE8 + 0xAA + 0x9E \\ &= -24 + -86 + -98\end{aligned}$$

Type	Value
8 bit signed	97
8 bit unsig...	0x61
16 bit signed	
16 bit unsi...	
32 bit unsi...	
32 bit signed	
64 bit unsi...	
64 bit signed	

Hex LittleEndian Insert ASCII Offset: 0 Selection: 1

日本語は漢字が数千種類あるので、256まででは足りない。そこで、日本では2byte(65536)で表すことにしたが、メーカーやOSが乱立した結果、複数の対応表（文字コード）が出回ってしまった。

主なものにJIS,shift-JIS,EUC(unix),cp932(windows)がある。

文字化けの正体

あ

shift-jis 82A0

EUC A4A2

utf-8 E38182

その後、日本語に中国語を混ぜ書きするような場合、2byteでも足りないことになった。（エンジニアのバカさ加減にも程がある）そこで、UNICODEという世界統一文字コードが作られ、utf8(utf16(utf32)などの規格が普及した。

文字化けとは、ファイルが作られたときの対応表と、ファイルを読み取るときに想定した対応表が一致していない場合に起きる。（賢いソフトはこのチェックが上手いので、文字化けが起こらない）

OpenOffice(Excel)には、ちゃんと文字コードを読み分ける選択肢が用意されている。ただし、その前に「データの型」の問題を理解しておく必要がある。

データの型(type)

コンピューターはもともと計算機だから、1に2を足せば3だと計算してくれる。しかし、「あ」に「い」を足したら「う」ではなく「あい」と答えてほしい。「12:59」の次は「13:00」だと知っていて欲しい。

もし、1+2を与えて「12」が現れたら、コンピューターは数字を文字と誤解しているのだろう。12:59の次に12:60を表示したら、時間を表していることを理解しているとはいえない。

要するに、コンピューターには、データの型(type)を理解して、それにふさわしい処理をしてほしい。もちろん、コンピューターは「あ」という文字を数字と誤解するほどバカではない。しかし「747」が数字なのか、飛行機の機種名なのかを推測するほど賢くはない。

だから、コンピューターに最初にデータを与える時、ユーザーは「これは数字ですよ」「これは文字ですよ」と教えてやる必要がある。

表計算ソフトの場合、身長や体重や「数値」として読み込まなければならない。背番号は、見た目は数字でも（背

番号の合計や平均には意味がないのだから）「文字列」として読み込まなければならぬ。飛行機のフライトナンバーも数値ではない。

データを開く

L01-27P-2K.csvを開く。用途、利用の現況、属性変更など、一見数字に見えるが文字列として使われている列は「テキスト」として読み取ること。



(特に変更しない限り) 文字列は左揃え、数値は右揃えに表示される。

A2 **数値は右揃え 文字列は左揃え** = 446937.606

	C	D	E	F	前年所在地コード	前年用途	前年連番	I	J	住居表示
1	経度	緯度	所在地コード	用途	連番	年次	前年連番	市区町村名		
2	446937.606	87649.186	47207	005	002	2015	47207	石垣	沖縄県	石垣
3	446965.808	87644.207	47207	000	002	2015	47207	石垣	沖縄県	石垣
4	446976.551	87627.02	47207	005	001	2015	47207	石垣	沖縄県	石垣
5	446991.851	87618.362	47207	000	001	2015	47207	石垣	沖縄県	石垣
6	447032.84	87632.788	47207	000	003	2015	47207	石垣	沖縄県	石垣
7	450999.961	89297.089	47214	005	002	2015	47214	宮古島	沖縄県	宮古島
8	451014.001	89289.864	47214	000	001	2015	47214	宮古島	沖縄県	宮古島
9	451019.527	89276.828	47214	000	003	2015	47214	宮古島	沖縄県	宮古島
10	451022.371	89292.143	47214	005	001	2015	47214	宮古島	沖縄県	宮古島
11	451056.406	89281.13	47214	000	002	2015	47214	宮古島	沖縄県	宮古島
12	451168.625	89140.619	47214	000	006	2015	47214	宮古島	沖縄県	宮古島
13	451365.286	89171.41	47214	000	005	2015	47214	宮古島	沖縄県	宮古島
14	451412.456	89113.252	47214	000	004	2015	47214	宮古島	沖縄県	宮古島
15	459555.448	94114.202	47210	009	001	2015	47210	糸満	沖縄県	糸満
16	459575.561	94249.429	47201	000	011	2015	47201	那覇	沖縄県	那覇
17	459575.86	94307.213	47201	005	017	2015	47201	那覇	沖縄県	那覇
18	459586.498	94326.077	47201	000	017	2015	47201	那覇	沖縄県	那覇
19	459593.366	94107.751	47210	000	005	2015	47210	糸満	沖縄県	糸満
20	459600.084	94064.85	47210	005	002	2015	47210	糸満	沖縄県	糸満

表示しやすいように、経度、緯度、住所コード、用途、住居表示、共通地点区分、H26価格、H27価格、属性移動H27以外を削除する。

F1

	A	B	C	D	E	F	G	H	I
1	経度	緯度	所在地コード	用途	住居表示	共通地点区分	H26価格	H27価格	属性移動H27
2	446937.606	87649.186	47207	005	沖縄県 石垣市新栄町70番12	false	61900	62300	1000001000
3	446965.808	87644.207	47207	000	沖縄県 石垣市字新川喜田盛14番1	false	38400	38600	1000001000
4	446976.551	87627.02	47207	005	沖縄県 石垣市字大川中ノ八力207番	false	121000	122000	1000000000
5	446991.851	87618.362	47207	000	沖縄県 石垣市字登野城村内104番	false	40200	40200	1000000000
6	447032.84	87632.788	47207	000	沖縄県 石垣市字平得西原129番3	false	35100	35400	1000000000
7	450999.961	89297.089	47214	005	沖縄県 宮古島市平良字下里西里61番	false	40400	40000	1000000000
8	451014.001	89289.864	47214	000	沖縄県 宮古島市平良字西里前比屋27	false	36000	35800	1000000000
9	451019.527	89276.828	47214	000	沖縄県 宮古島市平良字下里大原8424	false	31600	31300	1000000000
10	451022.371	89292.143	47214	005	沖縄県 宮古島市平良字西里羽立3911	false	58900	58000	1000000000
11	451056.406	89281.13	47214	000	沖縄県 宮古島市平良字東仲根ソデ山	false	28200	28100	1000000000
12	451168.625	89140.619	47214	000	沖縄県 宮古島市上野字野原東方原11	false	4560	4560	1000000000
13	451365.286	89171.41	47214	000	沖縄県 宮古島市城辺字比嘉比嘉125	false	5550	5540	1000000000
14	451412.456	89113.252	47214	000	沖縄県 宮古島市城辺字福里西方630	false	8910	8880	1000000000
15	459555.448	94114.202	47210	009	沖縄県 糸満市西崎町5丁目8番7外	false	22200	22800	1000000000
16	459575.561	94249.429	47201	000	沖縄県 那覇市具志2-25-20	false	90500	90900	1000001000
17	459575.86	94307.213	47201	005	沖縄県 那覇市赤嶺2丁目13番11外	false	139000	144000	1000000000
18	459586.498	94326.077	47201	000	沖縄県 那覇市金城3丁目8番3	false	134000	136000	1000000000
19	459593.366	94107.751	47210	000	沖縄県 糸満市西崎2-39-6	false	69700	70700	1000000000
20	459600.084	94064.85	47210	005	沖縄県 糸満市字糸満南組1944番	false	70800	71200	1000000000
21	459600.534	94297.385	47201	000	沖縄県 那覇市字田原不知嶺原2434番	false	107000	109000	1000000000
22	459601.927	94294.613	47201	005	沖縄県 那覇市字田原不知嶺原240番	false	120000	121000	1000000000
23	459608.454	94086.227	47210	000	沖縄県 糸満市西川町6-14	false	60000	60000	1000000000
24	459609.318	94220.158	47212	000	沖縄県 豊見城市字伊良波伊良波78	false	57600	58700	1000000000
25	459613.342	94065.132	47210	005	沖縄県 糸満市字糸満新組909番	true	71600	71900	1000000000
26	459619.402	94096.048	47210	000	沖縄県 糸満市字兼城河尻原476番2	false	56100	56100	1000000000
27	459623.568	94246.09	47212	000	沖縄県 豊見城市字我那覇漢原383番	true	77100	77900	1000000000
28	459627.707	94484.282	47201	009	沖縄県 那覇市港町3-7-54	false	87400	87600	1000000000
29	459630.126	94062.805	47210	000	沖縄県 糸満市字糸満佐場地原1589番	false	42400	42400	1000000000
30	459632.131	94380.433	47201	000	沖縄県 那覇市久米2-20-6	false	158000	160000	1000000000
31	459634.248	94311.313	47201	005	沖縄県 那覇市鏡原町27-29	false	141000	142000	1000000000
32	459634.936	94393.444	47201	005	沖縄県 那覇市若狭2-3-19	false	192000	195000	1000000000
33	459635.743	94319.016	47201	000	沖縄県 那覇市鏡原町15-15	true	122000	123000	1000000000
34	459638.248	94221.983	47212	000	沖縄県 豊見城市字上田上田原211番	false	72000	72800	1000000000
35	459639.522	94269.424	47201	005	沖縄県 那覇市カヌホ1-3-11	false	206000	216000	1000000000

データの公式解説

国交省が公表している報告書「分科会等で検討した地価公示価格形成要因等の概要」の千葉県部分には、以下のよ
うな分析が載っている。記者向けレクで配布される資料である。

全体の地価動向

- ・千葉県の住宅地の平均変動率は+0.1%(±0.0%)と前年比で僅かながら上昇となった。
- ・都市部では前年に引き続き地価の上昇傾向が見られるが、景気回復に陰りが見えたことから、上昇率は前年
に比べて鈍化している。
- ・木更津市・君津市の優良な住宅地では需要増にみあう土地の供給量が不足していることから地価が大幅に上
昇した。
- ・東日本大地震により被災を受けた地域は、まだ影響が残っているものの、全体的には復旧が進み落ち着きを
取り戻しつつある。
- ・今回の公示では、浦安市の美浜、舞浜地区で僅かながら上昇となった。
- ・地方部においては主要都市以外の地域では需給の停滞から大きな価格変動は見られず、ほぼ横這いとなっ
ている。
- ・放射線が高いと報じられた、東葛地区(柏市、松戸市、流山市、我孫子市)では地区外からの需要は依然とし
て少ないが現在では、その影響はほとんど見られない。
- ・我孫子市の成田線沿線で地価の大幅な下落が見られた。特に布佐地区では水害の影響もあり需要が激減し、
地価も大幅に下落した。

ここでいう平均変動率は、前年にも評価されたデータの変動率の平均である。これは国交省の担当部署に確認しな
ければならない。

本当に「千葉県の住宅地の平均変動率は0.1%」だったのか、確認したい。

表計算ソフトできること

まず、千葉のデータだけを抜き出したい。

所在地コードは都道府県コードと市町村コード（3桁）の組み合わせである。千葉県のコードは「12」なので、
「12***」という所在地コードが千葉県を表す。

列Jを「都道府県」とし、関数LEFTを使い、文字列の左から2文字を抜き出す。

The screenshot shows the OpenOffice Calc interface with the formula bar displaying '=LEFT(C2,2)'. The 'LEFT' function dialog is open, showing '文字列(C)' set to C2 and '文字数(D)' set to 2. The result is shown as 47. The formula bar also displays the result as 47.

	C	K
1	所在地コード	
2	47207	
3	47207	
4	47207	
5	47207	
6	47207	
7	47214	
8	47214	
9	47214	
10	47214	
11	47214	
12	47214	
13	47214	
14	47214	
15	47210	
16	47201	
17	47201	
18	47201	
19	47210	
20	47210	
21	47201	
22	47201	
23	47210	
24	47212	
25	47210	
26	47210	
27	47212	
28	009 沖縄県 那覇市港町 3 - 7 - 5 4 false 87400 87600 1000000000	
29	000 沖縄県 糸満市字糸満佐場地原 1 5 8 9 false 42400 42400 1000000000	
30	000 沖縄県 那覇市久米 2 - 2 0 - 6 false 158000 160000 1000000000	
31	005 沖縄県 那覇市鏡原町 2 7 - 2 9 false 141000 142000 1000000000	
32	005 沖縄県 那覇市若狭 2 - 3 - 1 9 false 192000 195000 1000000000	
33	000 沖縄県 那覇市鏡原町 1 5 - 1 5 true 122000 123000 1000000000	
34	000 沖縄県 豊見城市字上田上田原 2 1 1 番 false 72000 72800 1000000000	
35	005 沖縄県 那覇市タタラ 1 - 3 - 1 1 false 306000 316000 1000000000	

(住居表示の列を使い、最初の3文字が「千葉県」であるものを抜き出す方法でもよい)

「データ」の「フィルタ」から「オートフィルタ」を選び、「12」を選択すれば、千葉県だけが表示される。

L01-27P-2K.ods - OpenOffice Calc

J5 =LEFT(C5;2)

	C	D	E	F	G	H	I	J	K
1	所在地コード	用道番号	住居表示	共通地点区分	H 2 6 値	H 2 7 値	属性移動	H 2 都道府県	
17992	12208	000	千葉県 野田市関宿台町字東十一 2 5 4	false	21400	20500	1000001000	07	
18169	12226	000	千葉県 富津市富津字水主町 1 2 8 9番	false	15500	15400	1000000000	08	
18190	12208	000	千葉県 野田市次木字三島東 3 1 6番	false	29300	29200	1000000000	09	
18204	12226	000	千葉県 富津市富津字東下洲原 2 4 0 5	false	15300	15200	1000000000	10	
18218	12208	000	千葉県 野田市岡田字下尻坪 5 8 8番	false	24500	24300	1000000000	11	
18232	12208	000	千葉県 野田市木間ケ瀬字上羽貫 7 3 0	false	27700	27000	1000000000	12	
18247	12208	000	千葉県 野田市岩名 1 丁目 2 0番	1 2 false	81000	81000	1000000000	13	
18255	12208	005	千葉県 野田市尾崎字堂山 3 0 7番	8 false	87200	87200	1000000000	14	
18269	12208	009	千葉県 野田市中里字尾崎境 1 4 3番	1 2 false	39000	38900	1000000000	15	
18273	12208	000	千葉県 野田市尾崎字南谷原 9 0 2番	2 2 false	65600	65100	1000000000	16	
18278	12208	000	千葉県 野田市日の出町 9番	2 4 false	52900	52300	1000000000	17	
18302	12208	000	千葉県 野田市岩名字香取脳 1 0 5番	1 5 false	33000	32700	1000000000	12	
18304	12226	000	千葉県 富津市青木 3 丁目 2 0番	2 8 false	18900	18800	1000000000	12	
18307	12205	000	千葉県 館山市宮城字越地 4 3番	3 false	21600	21600	1000000000	12	
18317	12208	000	千葉県 野田市春日町 4 3番	1 3 true	66100	65700	1000000000	12	
18327	12208	000	千葉県 野田市つみ野 2 丁目 1 1番	1 1 false	62400	62200	1000000000	12	
18341	12226	000	千葉県 富津市青木字下龜塚 1 5 3 6番	6 false	17300	17200	1000000000	12	
18346	12208	000	千葉県 野田市光葉町 2 丁目 1 1番	1 8 false	69800	69800	1000000000	12	
18351	12226	000	千葉県 富津市大堀 3 丁目 1 9番	1 9 false	23900	23800	1000000000	12	
18361	12208	000	千葉県 野田市清水字出井ノ下 6 7 3番	3 false	65000	64500	1000000000	12	
18365	12226	000	千葉県 富津市下飯野字西飛附 1 6 3 6	6 false	9900	9800	1000000000	12	
18366	12226	000	千葉県 富津市大堀 2 丁目 1 2番	1 7 false	24900	24800	1000000000	12	
18367	12208	000	千葉県 野田市中野台字櫻内 4 0 7番	3 3 false	63100	62300	1000000000	12	
18380	12226	000	千葉県 富津市千種新田字水越 3 0 5番	5 false	17100	17000	1000000000	12	
18393	12205	000	千葉県 館山市那古字溜 1 5 3 6番	2 8 false	21600	21600	1000000000	12	
18394	12226	005	千葉県 富津市大堀字根岸 4 3番	5 false	26700	26500	1000000000	12	
18404	12226	000	千葉県 富津市下飯野字内小平田 1 0 1	1 false	15500	15400	1000000000	12	
18408	12208	000	千葉県 野田市清水字下原付 2 5 5番	1 1 false	68300	68100	1000000000	12	
18410	12208	005	千葉県 野田市野田字東上町 5 8 4番	1 2 false	93200	90900	1000000000	12	
18417	12226	000	千葉県 富津市大堀字東 1 4 9 9番	2 1 true	18300	18200	1000000000	12	
18421	12205	005	千葉県 館山市北条字浜通 1 8 6 6番	5 5 false	52600	52600	1000000000	12	
18427	12205	000	千葉県 館山市長須賀字塩焚 8 6番	1 2 false	29500	29500	1000000000	12	
18431	12208	000	千葉県 野田市上花輪新町 2 9番	2 2 false	61800	61100	1000000000	12	
18436	12205	000	千葉県 館山市北条字鶴ヶ谷 2 0 0 0番	0 false	29600	29600	1000000000	12	

ただし、これは「千葉以外が表示されていない」だけである。別のシートに千葉だけ抜き出すことにする。

「挿入」から「シート」を選び、新しいシートを「千葉」と名付ける。

元のシートに戻り、「データ」の「フィルタ」から「標準フィルタ」を選び、「詳細オプション」で「フィルタ結果の貼り付け先」をチェックし、新シート「千葉」のA1を指定する。

The screenshot shows the 'Standard Filter' dialog box in OpenOffice Calc. The dialog has the following settings:

- Filter conditions:**
 - Column: 都道府県 (Prefecture)
 - Operator: =
 - Value: 12
- Options:**
 - 大文字と小文字を区別(S)
 - 正規表現(E)
 - 列ラベルを含む範囲(C)
 - 重複なし(N)
 - フィルタ結果の貼り付け先(R)
 - ソース範囲とリンクする(K)
- Data range:** \$Sheet1.\$A\$1:\$J\$23364 (Anonymous_Sheet_DB_1)

1125件が抜き出された。次に、変動率の対象になる「H 2 6 價格」があるものだけを抜き出す。新しいシート「千葉継続」を作つておく。

平成27年に新たに評価された場所は「H 2 6 價格」がゼロになっていることを利用する事にする。列Kを「継続」とし、関数IFを選ぶ。「H 2 6 價格=0」なら1、それ以外は0になるような設定をする。

L01-27P-2K.ods - OpenOffice Calc

	B	C	D	E	F	G	H	I	J	K	L
1	緯度	所在地コード	用途	住居表示	共通地点区分	H 2 6 價格	H 2 7 價格	属性移動	H 2 都道府県	継続	
2	129923.489	12208	000	千葉県	野田	false	21400	20500	1000001000	12	
3	127122.57	12226	000	千葉県	富津	false	15500	15400	1000000000	12	
4	129737.376	12208	000	千葉県	野田	false	29300	29200	1000000000	12	
5	127105.333										
6	129655.246										
7	129712.95										
8	129515.393										
9	129528.119										
10	129579.635										
11	129536.132										
12	129554.179										
13	129470.414										
14	127167.84										
15	125928.684										
16	129518.786										
17	129401.244										
18	127166.033										
19	129492.41										
20	127186.423										
21	129423.87										
22	127125.536										
23	127195.51										
24	129394.213										
25	127057.356										
26	126065.286										
27	127196.028										
28	127157.22										
29	129435.494										
30	129412.44										
31	127209.483										
32	125983.66	<input checked="" type="checkbox"/> 行列(A)									
33	125955.745										
34	129370.781										
35	125006.015										

関数ウィザード

IF

数式の結果 0

論理式の結果に応じて、指定した値を返します。

FALSE の場合 (任意)

論理式の結果が FALSE のときに返す値。

論理式(C) G2=0

TRUE の場合(D) 1

FALSE の場合(E) 0

数式(M)

結果 0

=IF(G2=0;1;0)

ヘルプ(H) キャンセル OK

先と同様に、列「継続」が0の行だけを新シート「千葉継続」に抜き出す。

The screenshot shows a spreadsheet application window with a toolbar at the top. A filter dialog box is open over the main data area. The dialog has sections for 'フィルタ条件' (Filter Conditions) and 'オプションを減らす' (Reduce Options). It contains several dropdowns and checkboxes. One checkbox is checked: '列ラベルを含む範囲' (Include column labels in range). Another checkbox is checked: 'ソース範囲とリンクする' (Link to source range). The 'データ範団' (Data Range) is set to '\$千葉.\$A\$1:\$K\$1126 (無題)'.

経度	緯度								都道府県	継続
503239.352	129923.4									
503345.2	127122.									
503356.514	129737.3									
503364.794	127105.3									
503374.255	129655.2									
503385.293	129712.									
503394.628	129515.3									
503399.29	129528.1									
503412.671	129579.6									
503416.379	129536.1									
503419.388	129554.1									
503432.942	129470.4									
503433.464	127167.									
503434.213	125928.6									
503442.213	129518.7									
503450.006	129401.2									
503456.476	127166.0									
503458.232	129492.									
503462.243	127186.4									
503470.696	129423.									
503475.156	127125.5									
503475.552	127195.									
503476.405	129394.2									
503486.446	127057.3									
503493.426	126065.2									
503493.883	127196.0									
503499.258	127157.22	12226	000	千葉県	富津>false	15500	15400	1000000000	12	
503501.605	129435.494	12208	000	千葉県	野田>false	68300	68100	1000000000	12	
503502.718	129412.44	12208	005	千葉県	野田>false	93200	90900	1000000000	12	
503506.528	127209.483	12226	000	千葉県	富津>true	18300	18200	1000000000	12	
503508.186	125983.66	12205	005	千葉県	館山>false	52600	52600	1000000000	12	
503512.247	125955.745	12205	000	千葉県	野田>false	29500	29500	1000000000	12	
503514.36	129370.781	12208	000	千葉県	館山>false	61800	61100	1000000000	12	
503519.002	125006.015	12205	000	千葉県	館山>false	38600	38600	1000000000	12	

列Lを「変動率」とし、「=(H2-G2)/G2」とする。

The screenshot shows a spreadsheet application window with a toolbar at the top. The formula bar at the top left contains the formula $= (H2-G2)/G2$. The main area displays a table with data and calculated results in column L.

用途	住居表示	共通地点区分	H 2	価格	H 2	価格	属性移動	H 2	都道府県	継続	変動率
000	千葉県	野田>false		21400		20500	100000	1000	12	0	-0.042056075
000	千葉県	富津>false		15500		15400	1000000000	000	12	0	
000	千葉県	野田>false		29300		29200	1000000000	000	12	0	
000	千葉県	富津>false		15300		15200	1000000000	000	12	0	

計算結果のセルの上で右クリックし、「書式設定」を選び、パーセント表示に変更する。

The screenshot shows a spreadsheet application window with a toolbar at the top. The formula bar displays the formula $=\text{AVERAGE}(L3:L1113)$. The main area contains a table with columns labeled F through L. The last column, L, is titled '変動率' (Change Rate) and contains numerical values. A context menu is open over the data, specifically the 'Format Cells' dialog for cell L2. This dialog shows the formula as $=\text{AVERAGE}(L3:L1113)$ in the '書式(O)' tab. Other tabs include '数' (Number), 'フォント' (Font), '配置' (Orientation), '日本語文の体裁' (Japanese Style), '外枠' (Border), '背景' (Background), and 'セルの保護' (Cell Protection). The '書式(O)' tab also shows a preview of the current cell's value.

2行目に移動し、「挿入」から「行」を選び、新しい行を2行目に挿入する。平均値を計算すると、0.217%になる。

	F	G	H	I	J	K	L
	共通地点区分	H 2 6 価格	H 2 7 価格	属性移動 H 2 ▶ 都道府県	都道府県	継続	変動率
野田▶false		21400	20500	1000001000	12	0	0.2179%
富津▶false		15500	15400	1000000000	12	0	-0.65%
野田▶false		29300	29200	1000000000	12	0	-0.34%
富津▶false		15300	15200	1000000000	12	0	-0.65%

報告書は「住宅地の平均変動率」だった。シート「千葉」に戻り、「継続かつ住宅地」を列Jに上書きする。

住宅地は、定義によれば「用途」が「000」である。今度は関数ANDを使い、「D2="000"」かつ「G2<>0」を指定する。

The screenshot shows the OpenOffice Calc interface with the 'Function Wizard' dialog open over a spreadsheet. The formula in cell J2 is `=AND(D2="000";G2<>0)`. The 'Function Wizard' dialog is displayed, with the 'AND' function selected. The result is set to TRUE. The logical values are defined as D2="000" and G2<>0. The formula bar at the bottom of the dialog also shows `=AND(D2="000";G2<>0)`.

結果は以下のようになる。

=	<code>=AND(D3="000";G3<>0)</code>					
D	E	F	G	H	I	J
用途	住居表示	共通地点区分	H 2 6 価格	H 2 7 価格	属性移動 H 2 → 繼続かつ住宅地	系
000	千葉県 野田▶false		21400	20500 1000001000	TRUE	
000	千葉県 富津▶false		15500	15400 1000000000	TRUE	
000	千葉県 野田▶false		29300	29200 1000000000	TRUE	
000	千葉県 富津▶false		15300	15200 1000000000	TRUE	
000	千葉県 野田▶false		24500	24300 1000000000	TRUE	
000	千葉県 野田▶false		27700	27000 1000000000	TRUE	
000	千葉県 野田▶false		81000	81000 1000000000	TRUE	
005	千葉県 野田▶false		87200	87200 1000000000	FALSE	
009	千葉県 野田▶false		39000	38900 1000000000	FALSE	
000	千葉県 野田▶false		65600	65100 1000000000	TRUE	
000	千葉県 野田▶false		52900	52300 1000000000	TRUE	
000	千葉県 野田▶false		33000	32700 1000000000	TRUE	
000	千葉県 富津▶false		18900	18800 1000000000	TRUE	
000	千葉県 館山▶false		21600	21600 1000000000	TRUE	

新シート「千葉継続住宅地」に抜き出す。平均を計算すると、確かに「1.028%」になる。

Σ = =AVEDEV(K3:K862)

D	E	F	G	H	I	J	K
ード	用途	住居表示	共通地点区分	H 2 6 価格	H 2 7 価格	属性移動 H 2 → 繼続かつ住宅	変動率
000	千葉県	野田	false	21400	20500	1000001000	TRUE -4.21%
000	千葉県	富津	false	15500	15400	1000000000	TRUE -0.65%
000	千葉県	野田	false	29300	29200	1000000000	TRUE -0.34%
000	千葉県	富津	false	15300	15200	1000000000	TRUE -0.65%
000	千葉県	野田	false	24500	24300	1000000000	TRUE -0.82%
000	千葉県	野田	false	27700	27000	1000000000	TRUE -2.53%
000	千葉県	野田	false	81000	81000	1000000000	TRUE 0.00%
000	千葉県	野田	false	65600	65100	1000000000	TRUE -0.76%
000	千葉県	野田	false	52900	52300	1000000000	TRUE -1.13%
000	千葉県	野田	false	33000	32700	1000000000	TRUE -0.91%
000	千葉県	富津	false	18900	18800	1000000000	TRUE -0.53%
000	千葉県	館山	false	21600	21600	1000000000	TRUE 0.00%
000	千葉県	野田	true	66100	65700	1000000000	TRUE -0.61%
000	千葉県	野田	false	62400	62200	1000000000	TRUE -0.32%
000	千葉県	富津	false	17300	17200	1000000000	TRUE -0.58%
000	千葉県	野田	false	69800	69800	1000000000	TRUE 0.00%
000	千葉県	富津	false	23900	23800	1000000000	TRUE -0.42%
000	千葉県	野田	false	65000	64500	1000000000	TRUE -0.77%
000	千葉県	富津	false	9900	9800	1000000000	TRUE -1.01%

報告書には、県内最高値もある。念のため、シート「千葉」で「データ」「並べ替え」を使って確かめてみよう。

3. 特徴的な地点の地価動向とその要因

●県全体について

区分	標準地番号	所在地	価格	変動率	変動要因
最高価格地	住宅地 市川- 35	市川市菅野1丁目20番2	327,000	+0.3 (+2.8)	千葉県を代表する古くからの高級住宅地。需要に見合う供給がないため供給があれば高値となっている。景気回復の一服感から上昇率は縮小した。
	商業地 千葉中央5- 1	千葉市中央区富士見2丁目2番1外	1,500,000	-1.3 (-1.9)	千葉市駅前のロータリーに面する商業ビル。現在都市銀行の支店が入居しているがオフィス需要の減退、商況の衰退に伴い引き続き下落傾向にある。
上昇率1位又は下落率最小	住宅地 木更津- 14	木更津市請西南3丁目33番5	42,200	+9.9 (+9.7)	木更津市街地南東部に位置する区画整理地内の住宅地。地域の熟成に伴い需要が増加し、高値の取引が見られるようになった。地価は上昇局面にある。
	商業地 君津5- 5	君津市南子安6丁目21番5	43,100	+6.4 (+3.6)	南子安地区を南北に走行する国道127号線沿いの路線商業地である。近年店舗の進出需要が見られるが、供給が少なく、また価格帯も低いことから価格は上昇基調にある。
下落率1位	住宅地 我孫子- 29	我孫子市布佐西町68番13	35,000	-10.9 (-6.4)	被災地の隣接地区であり需要が弱い地域であるが、台風による大規模な水害に見舞われたため、需要は激減し、平成26年に入り価格は急落している。
	商業地 流山5- 2	流山市江戸川台東2丁目10番	171,000	-2.8 (-1.7)	江戸川台駅前商店街に位置する店舗兼住宅であるが、地域の高齢化、衰退に伴い商業地としての活気が見られず、商業地としての需要は著しく低い。

SQLとXML

リレーションナルデータベースとは

数十万件以上のデータがある場合、表計算ソフトでは読めない。RDBはデータの検索に特化したソフトウェアで、その操作にSQLという特別の言語を使う。

(半世紀前にできた)RDBは、貴重なメモリーを節約するため、重複部分を別のテーブルに分けて格納する。そうすれば修正も関連場所だけで済む。

Relational Databaseとは

時間	送り手	支店	口座番号	受け手	支店	口座番号	送金額
1/1	山田太郎	東京	648123	佐藤二郎	大阪	812313	12000
1/2	加藤三郎	東京	231123	佐藤二郎	大阪	812313	24000
1/2	山田太郎	東京	648123	加藤三郎	東京	231123	10000
1/4	佐藤二郎	大阪	812313	山田太郎	東京	648123	38000

テーブルを分割して重複データを減らす

時間	送り手	受け手	送金額	ID	名前	支店	口座
1/1	1	2	12000	1	山田太郎	東京	648123
1/2	3	2	24000	2	佐藤二郎	大阪	812313
1/2	1	3	10000	3	加藤三郎	東京	231123
1/4	2	1	38000				

詳細なテーブルが必要な時だけ、データを再結合すればよい。

```
select * from tblA
inner join tblB on tblA.送り手 = tblB.ID and
inner join tblB as B on tblA.送り手 = B.ID and
```

時間	送り手	受け手	送金額	ID	名前	支店	口座
1/1	1	2	12000	1	山田太郎	東京	648123
1/2	3	2	24000	2	佐藤二郎	大阪	812313
1/2	1	3	10000	3	加藤三郎	東京	231123
1/4	2	1	38000				

時間	送り手	支店	口座番号	受け手	支店	口座番号	送金額
1/1	山田太郎	東京	648123	佐藤二郎	大阪	812313	12000
1/2	加藤三郎	東京	231123	佐藤二郎	大阪	812313	24000
1/2	山田太郎	東京	648123	加藤三郎	東京	231123	10000
1/4	佐藤二郎	大阪	812313	山田太郎	東京	648123	38000

表計算ソフト（Excel）やAccessなどにはRDBに接続する機能がある。Pythonなどにもコネクタ（バインダー）などが用意されている。

XMLとは

XML(extensible markup language)とは

csv形式には問題がある

	A	B	C	D	E	
1	Title	Author	Year	Price	Sold	
2	正義の法	大川隆法	2016	2160	5120	
3	あの日	小保方晴子	2016	1512	3123	
4	おやすみロジャー	エリーン	2015	1399	1282	
5	真田丸	三谷幸喜	2015	1188	503	
6						
7						

共著者が表示できない

特記事項がある場合

csvフォーマットには問題がある。要素数が固定していない場合、テーブルでは効率的にデータを格納できない。
(スポーツの記録で減点要素を記録する場合、本の共著者など)

タグの包含関係を使う

```
<booklist>
  <book isbn="617-28-31918237">
    <title>おやすみ、ロジャー</title>
    <author>エリーン</author>
    <author>カール＝ヨハン</author>
    <translator>三橋美穂</translator>
    <price unit="jp-yen">1399</price>
  </book>
  <book isbn="137-38-512416">
    <title>………</title>
  </booklist>
```

ブラウザで閲覧可/プログラム(parser)で読み取る

データの性質上、プログラム(parser)から読み取る必要がある。webで使う場合は、JSONという形式が使われる。

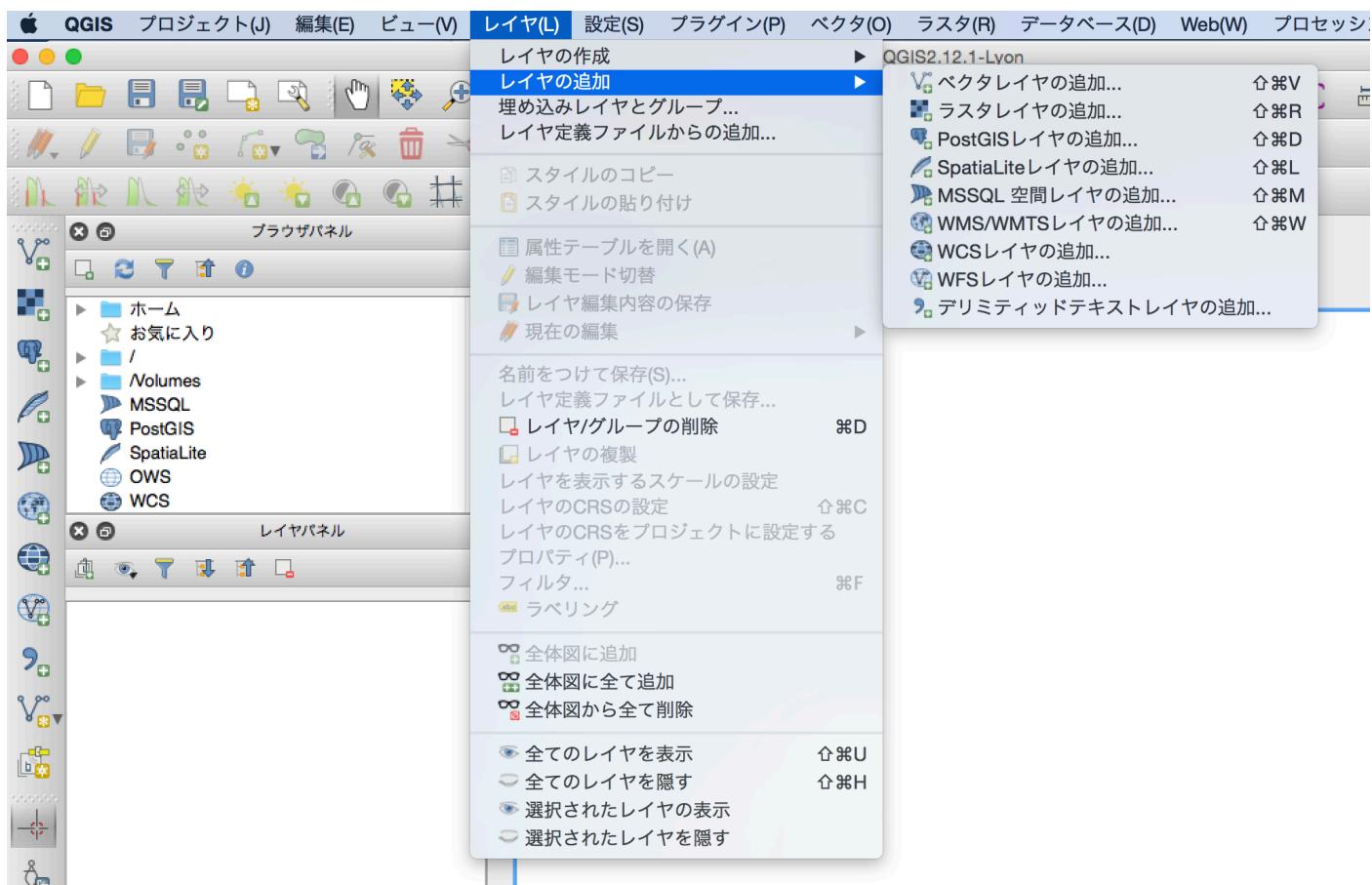
QGIS(デジタル地図)

表計算ソフトでは、「ある地域で」地価が変化したことに気付くことは難しい。このため、地図を使ってデータを見るためのGIS(Geographic Information System)というソフトウェアが開発されている。

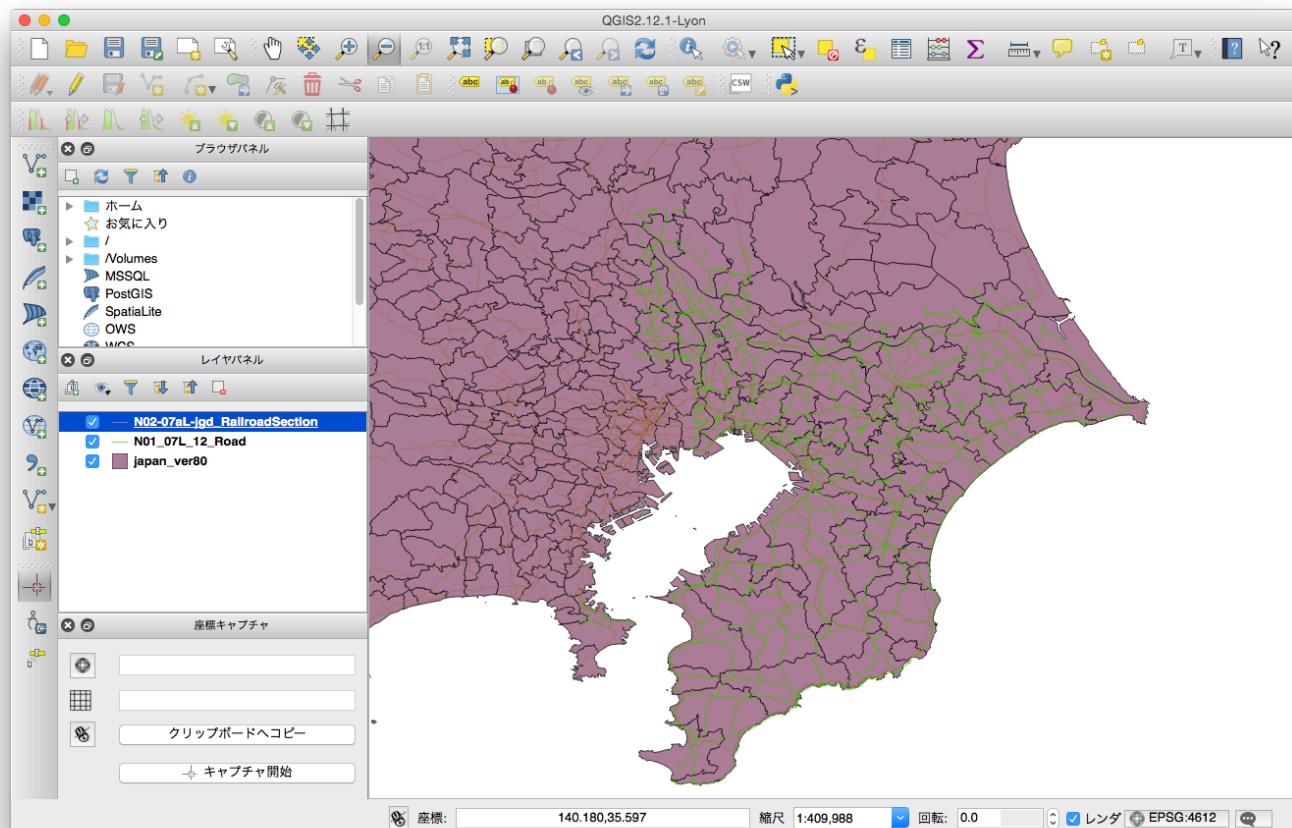
国土数値情報からSHP形式を選び、「道路」と「鉄道」を選び、ダウンロードする。また、ESRI Japanの全国市区町村界データをダウンロードする。(02フォルダにある)

- N01-07L-12-01/N01_07L_12_Road.shp 道路データ
- N02-07aL-jgd/N02-07aL-jgd_RailroadSection.shp 鉄道データ
- japan_ver80/japan_ver80.shp 市区町村データ

QGISを立ち上げ、「レイヤ」の「レイヤの追加」「ベクタレイヤの追加」からEPSG:4612で3つのshpファイルを開く。



このように、GISは地図データを表示するソフトウェアである。



地図データのフォーマットには様々なものがあるが、QGISは大抵の地図ファイルに対応している。

国土地理院や企業が作ったファイルを利用するだけでなく、自分独自のデータを作ることもできる。

CSVからShapefile

表計算ソフトで作った加工したファイルからシェープファイルを作ることにする。

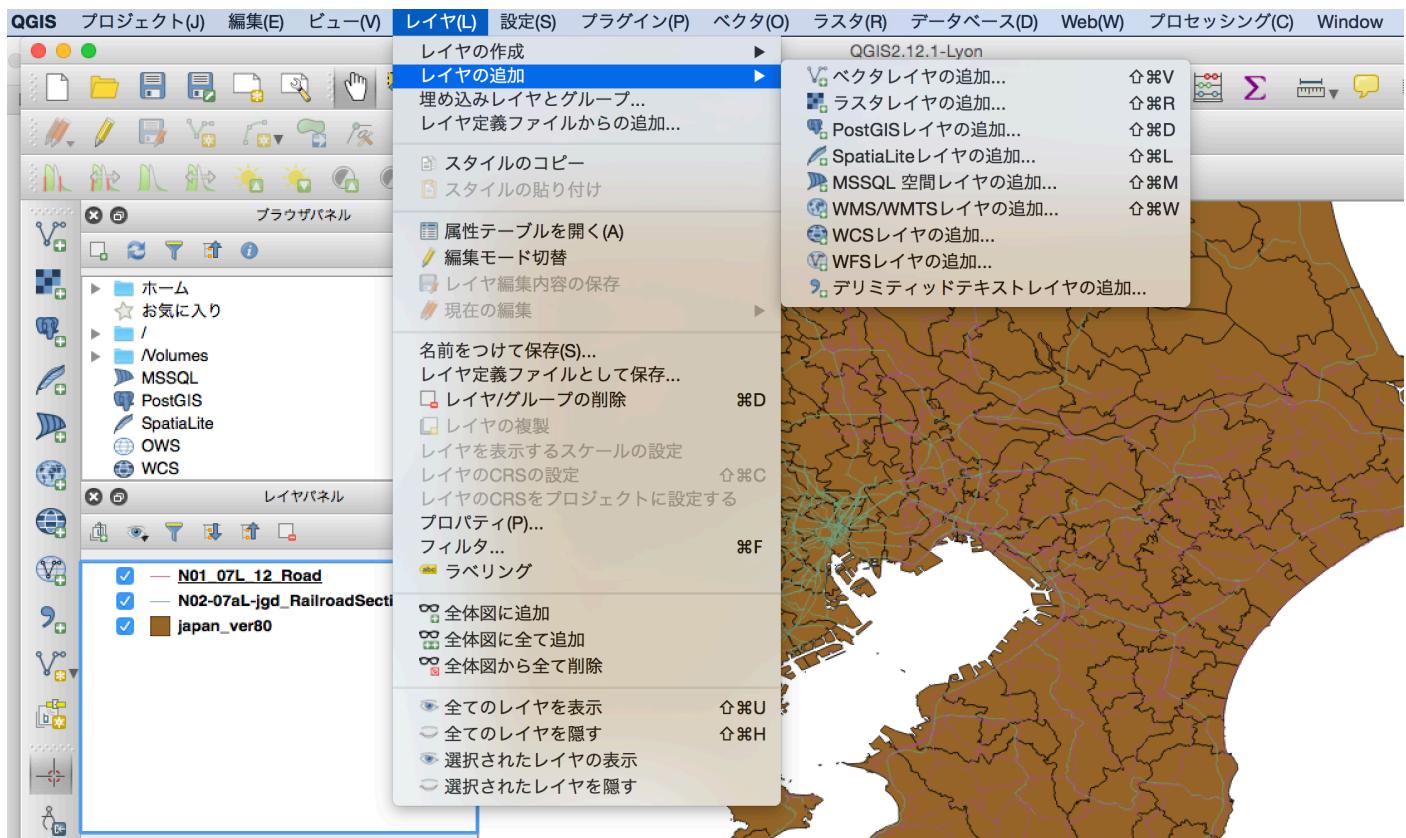
そのためには、データは、度数表示の緯度・経度を含んでいなければならない。シート「千葉継続住宅地」の3列目に「緯度（度）」と「経度（度）」を挿入し、秒単位の緯度・経度を3600（1度=3600秒）で割った数値を計算させる。

	A	B	C	D	E	F	G
1	経度	緯度	経度(度)	緯度(度)	所在地コード	用途	住居表示
2	503734.287	128618.358	139.9261908333	35.7273216667	12203	000	千葉県 市川市菅野1 - 7
3	503756.78	128607.62	139.9324388889	35.7243388889	12203	000	千葉県 市川市八幡4 - 1
4	503604.258	128288.743	139.8900716667	35.6357619444	12227	000	千葉県 浦安市舞浜3 - 2
5	503689.936	128360.884	139.9138711111	35.6558011111	12227	000	千葉県 浦安市美浜4 - 1
6	503679.395	128646.382	139.9109430556	35.7351061111	12203	000	千葉県 市川市真間2 - 2
7	503638.316	128401.911	139.8995322222	35.6671975	12227	000	千葉県 浦安市北栄3 - 3
8	503641.274	128390.202	139.9003538889	35.663945	12227	000	千葉県 浦安市北栄3 - 8
9	503693.809	128629.429	139.9149469444	35.7303969444	12203	000	千葉県 市川市菅野3 - 2
10	503730.511	128632.298	139.9251419444	35.7311938889	12203	000	千葉県 市川市菅野4 - 8
11	503755.326	128485.426	139.932035	35.6903961111	12203	000	千葉県 市川市妙典5 - 1
12	503939.354	128537.219	139.9831538889	35.7047830556	12204	000	千葉県 船橋市本町7 - 1
13	503685.981	128615.247	139.9127725	35.7264575	12203	000	千葉県 市川市新田2 - 2
14	503714.3	128608.276	139.9206388889	35.7245211111	12203	000	千葉県 市川市平田1 - 1
15	503664.602	128412.572	139.9068338889	35.6701588889	12203	000	千葉県 市川市南行徳2 -
16	503657.636	128652.84	139.9048988889	35.7369	12203	000	千葉県 市川市市川3 - 3
17	503644.439	128619.194	139.9012330556	35.7275538889	12203	000	千葉県 市川市市川南4 -
18	503633.826	128374.801	139.898285	35.6596669444	12227	000	千葉県 浦安市猫実2 - 2
19	504423.652	128179.055	140.1176811111	35.6052930556	12101	000	千葉県 千葉市中央区新宿
20	503720.781	128579.607	139.9959160111	35.7116380556	12203	000	千葉県 市川市市川大和田1

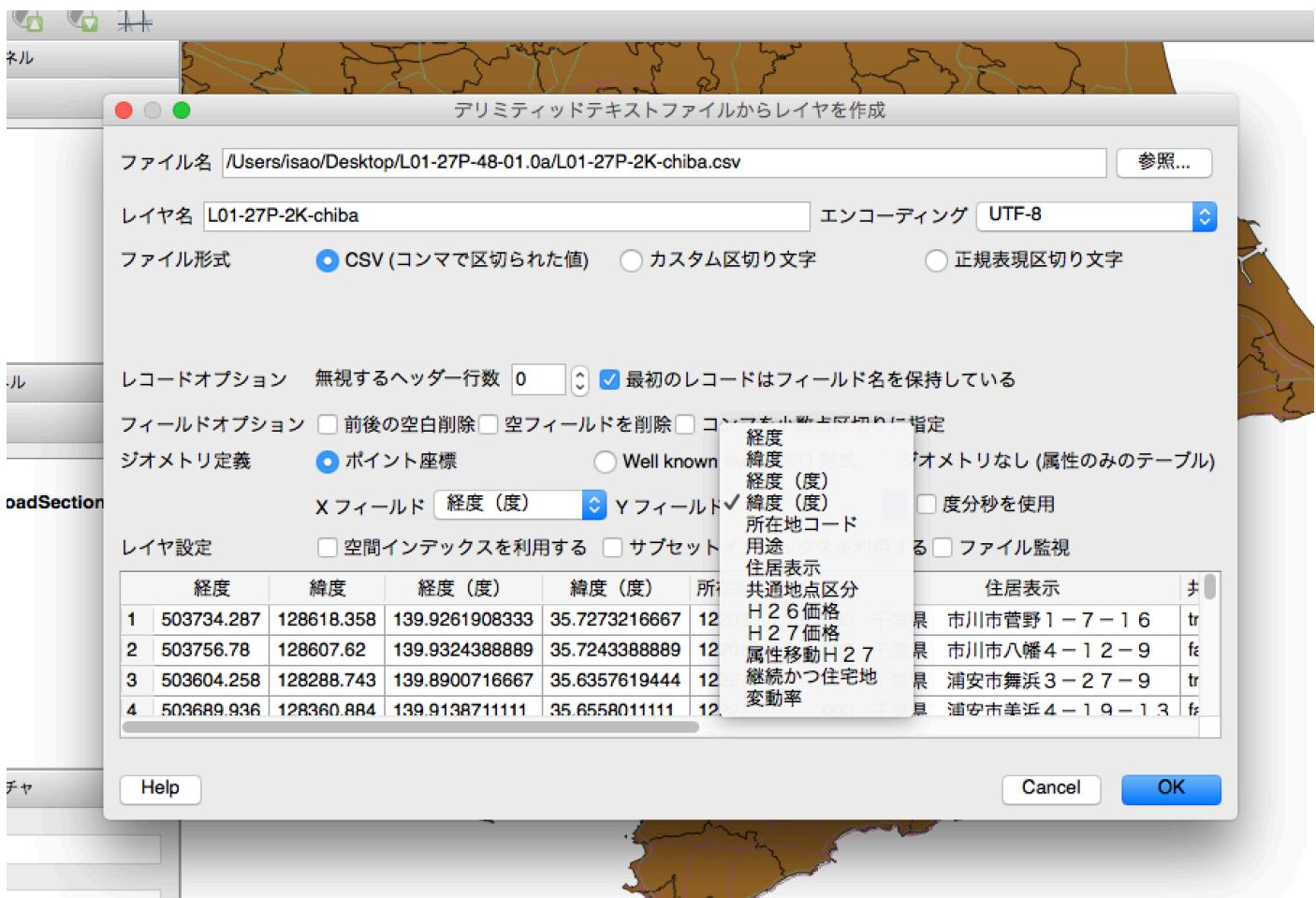
このシートをCSV形式で保存する。(ファイル名をL01-27P-2K-chiba.csvとする)

ただし、「変動率」の表示は(パーセント表示のままCSVで保存すると「1.2%」のように保存されてしまい、文字列と誤解されるため)小数表示に戻しておく。

QGISの「レイヤ」「レイヤの追加」「デリミテッドテキストレイヤの追加」を選び、CSVファイルを読み込む。



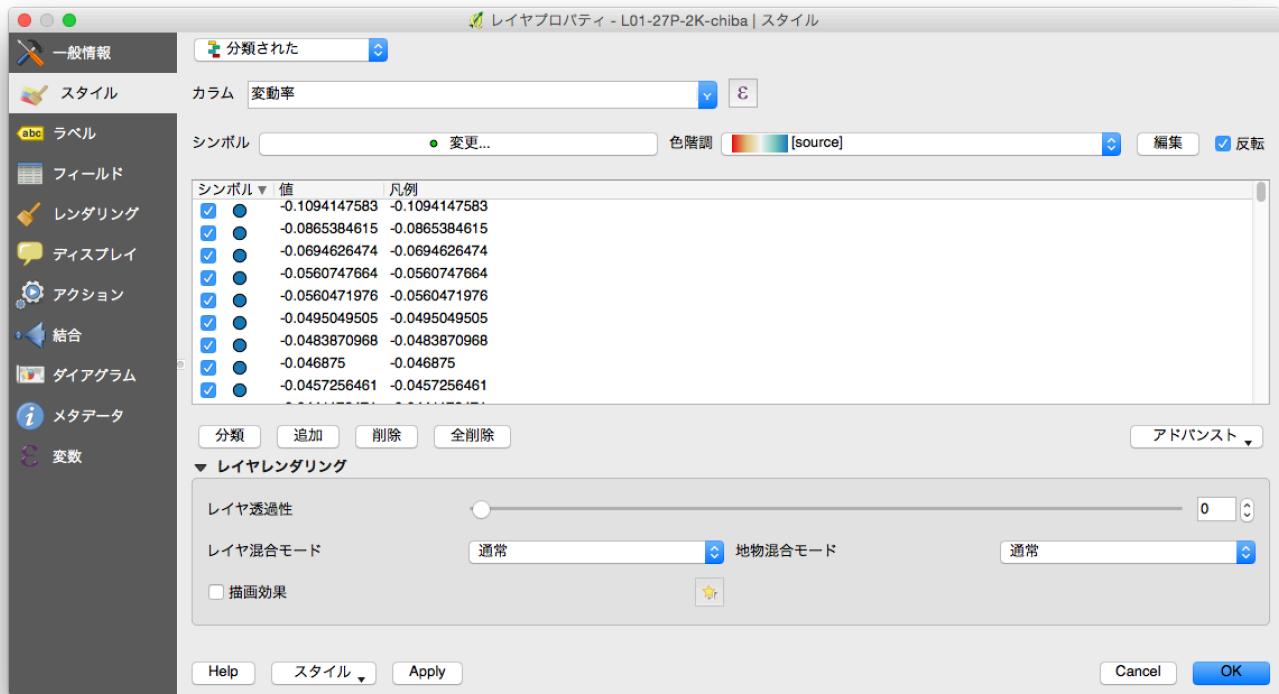
この時、どの列が経度（X座標）と緯度（Y座標）であるかをQGISに教えてやる必要がある。



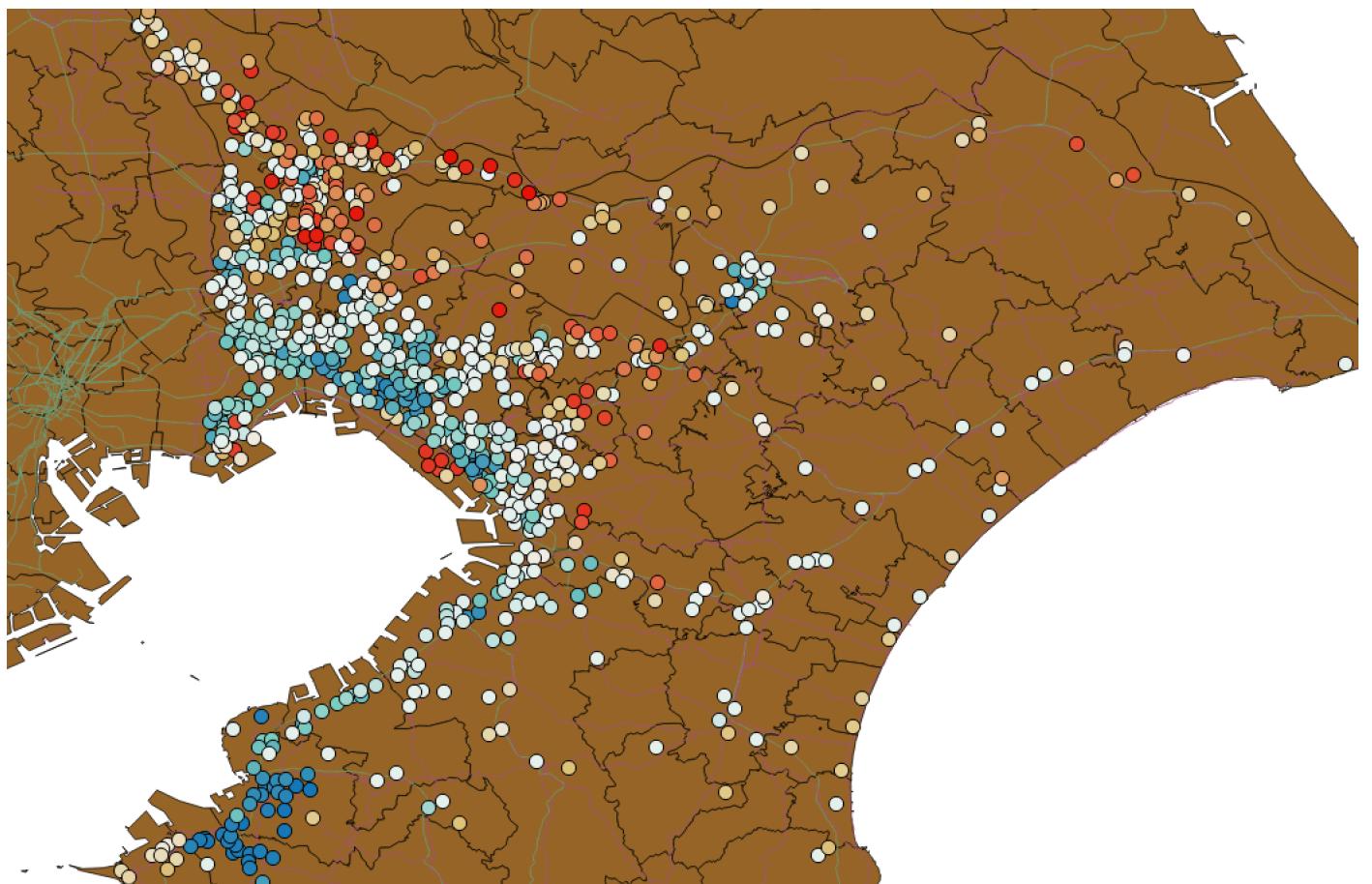
こうしてできたshapefileをchiba4326.shpとする。

マーカーの色分け

読み込んだレイヤー上で右クリックし、「プロパティ」を選び、左メニューの「スタイル」で上から「分類された」を選び、カラムに「変動率」を選び、シンボルの形と色階調を設定する。「分類」ボタンを押すと、データをどのように色分けするかが、テーブルに表示される。



すると、期待通り、データが色分けされる。



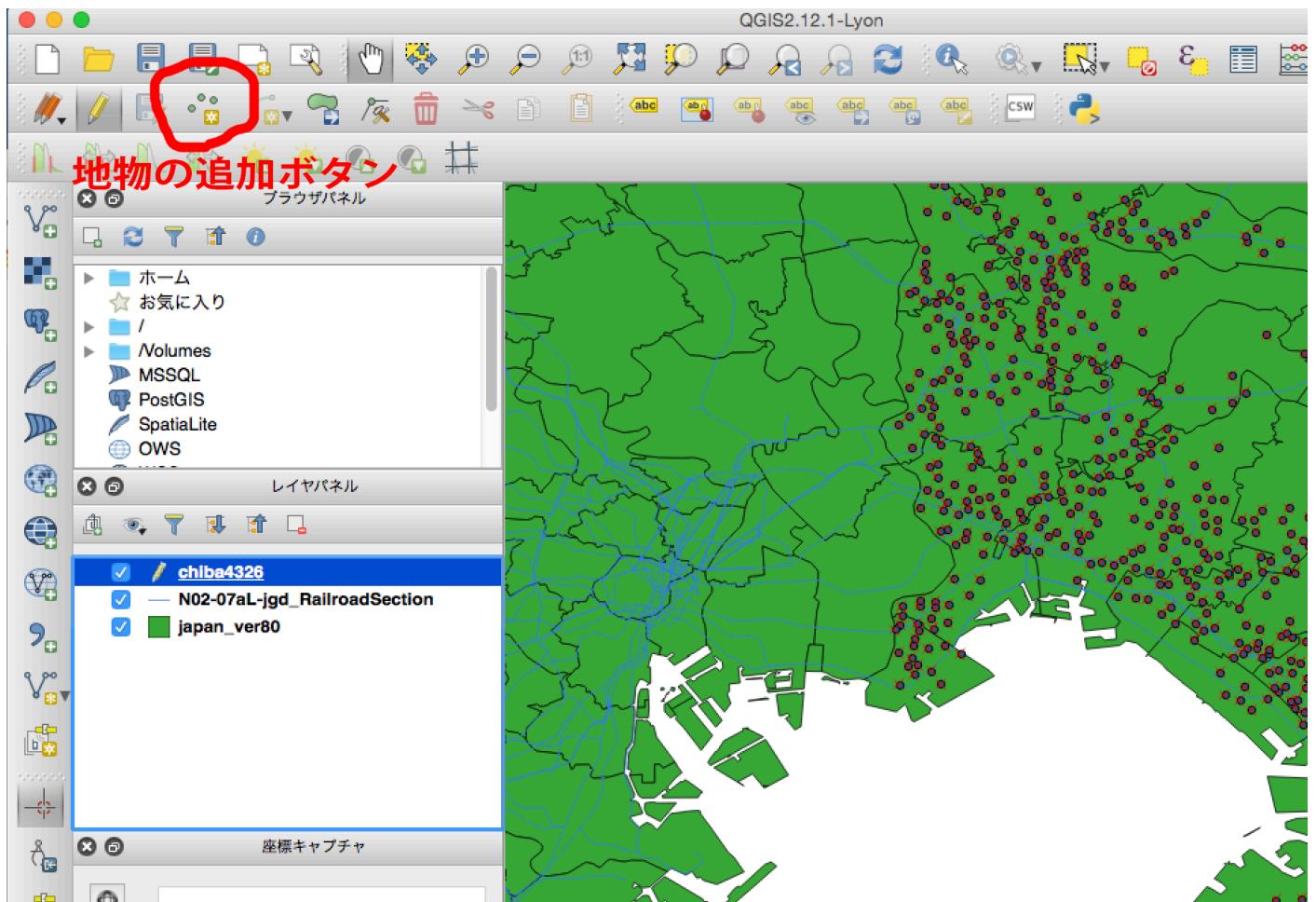
この例では、マイナスを赤、プラスを青に設定した。

木更津周辺が最も値上がりし、千葉市から東京方面の地価上昇地帯に、一部だけ下落地帯があることなどが一目瞭然だ。

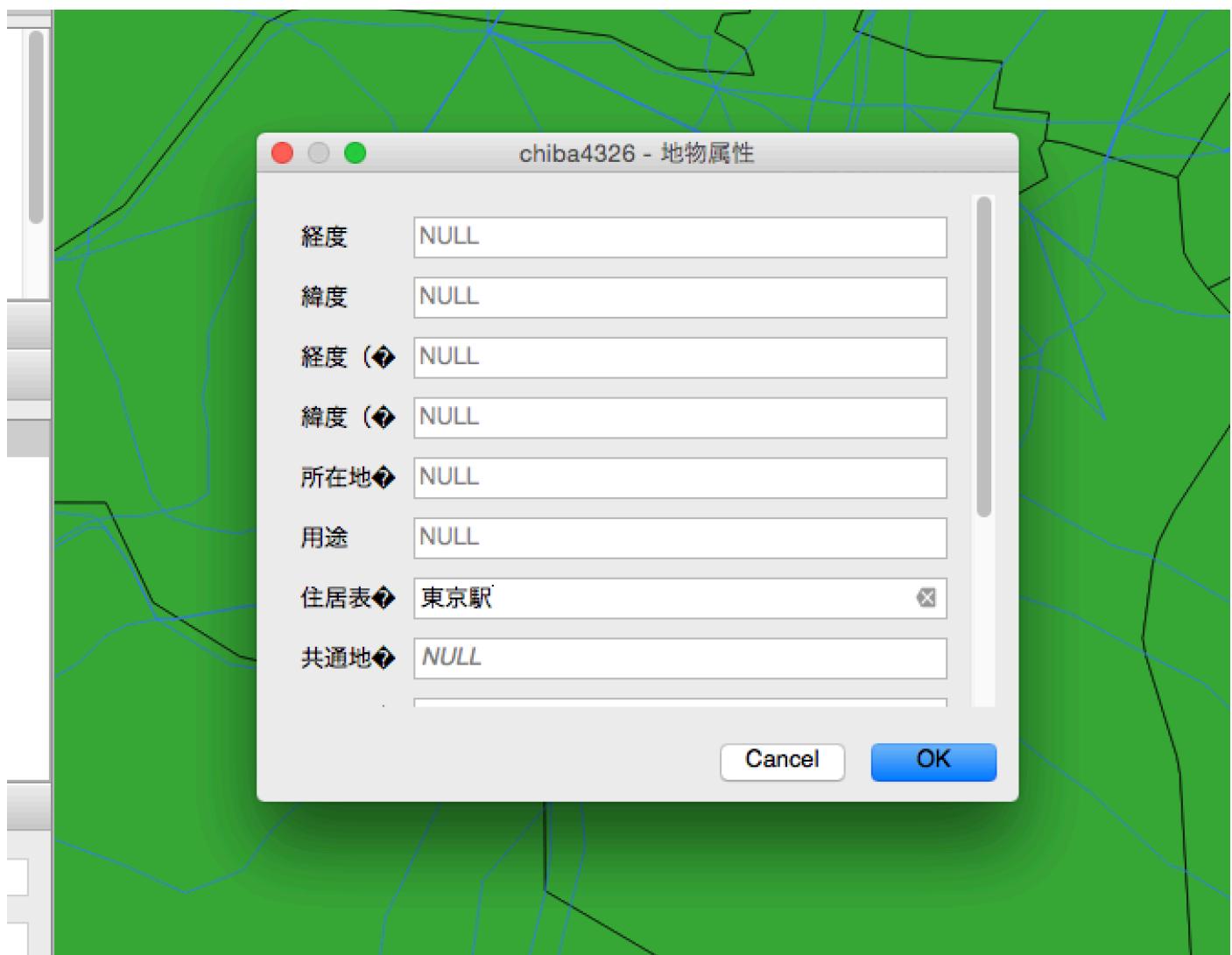
空間検索(spatial search)

GISはただの地図表示ソフトウェアではない。ここでは、東京駅から距離を計算してみる。

まず、東京駅を地図に追加する。レイヤー chiba4326 を選択し、「地物の追加」を選択する。



東京駅周辺の場所でダブルクリックすると、フィールドデータを入力するよう求められる。いま、必要なのは「東京駅」だけである。



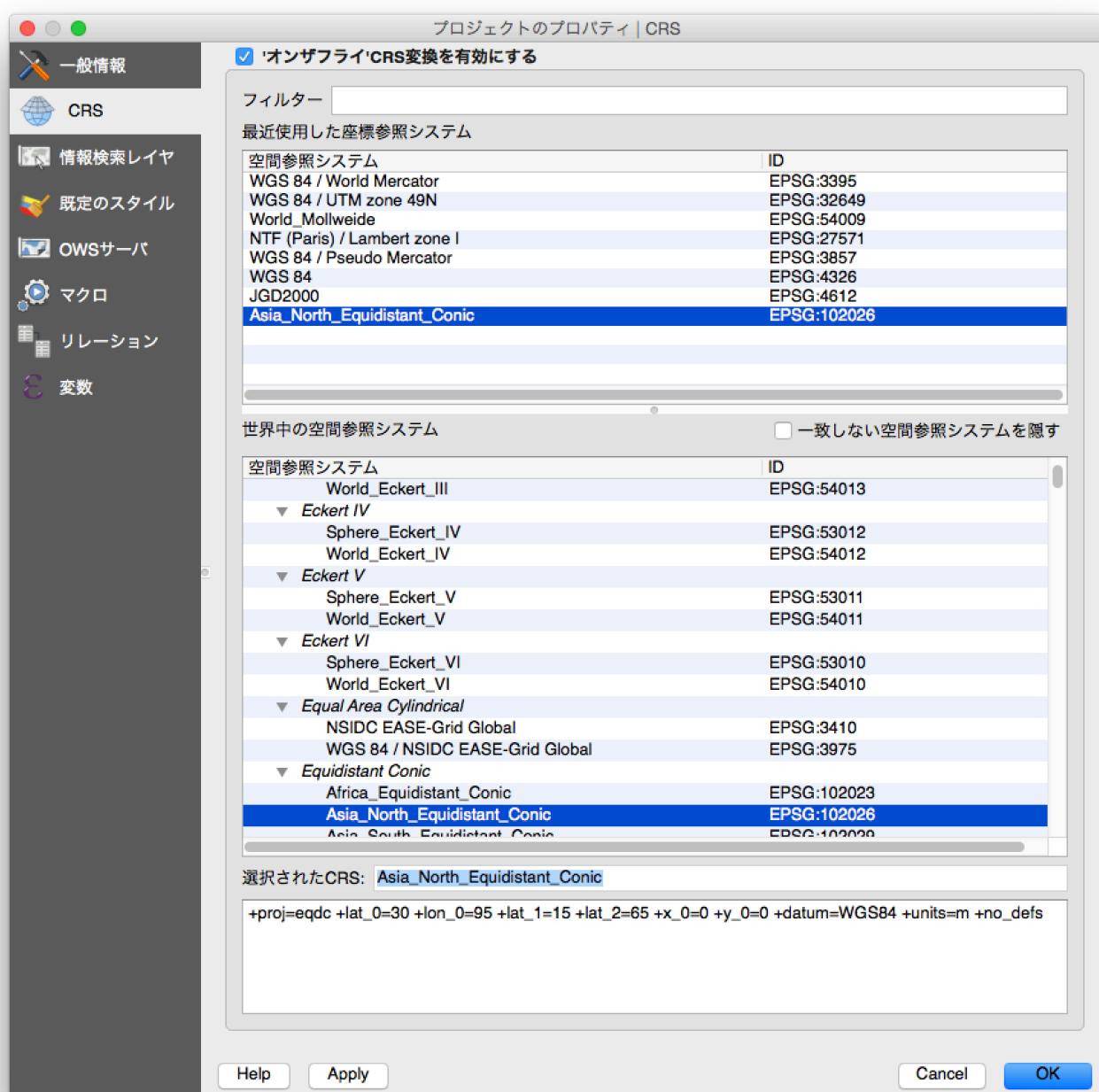
属性テーブルを開くと、確かに追加されている。

属性テーブル - chiba4326 :: 総地物数: 861, フィルター数: 861, 選択数: 0

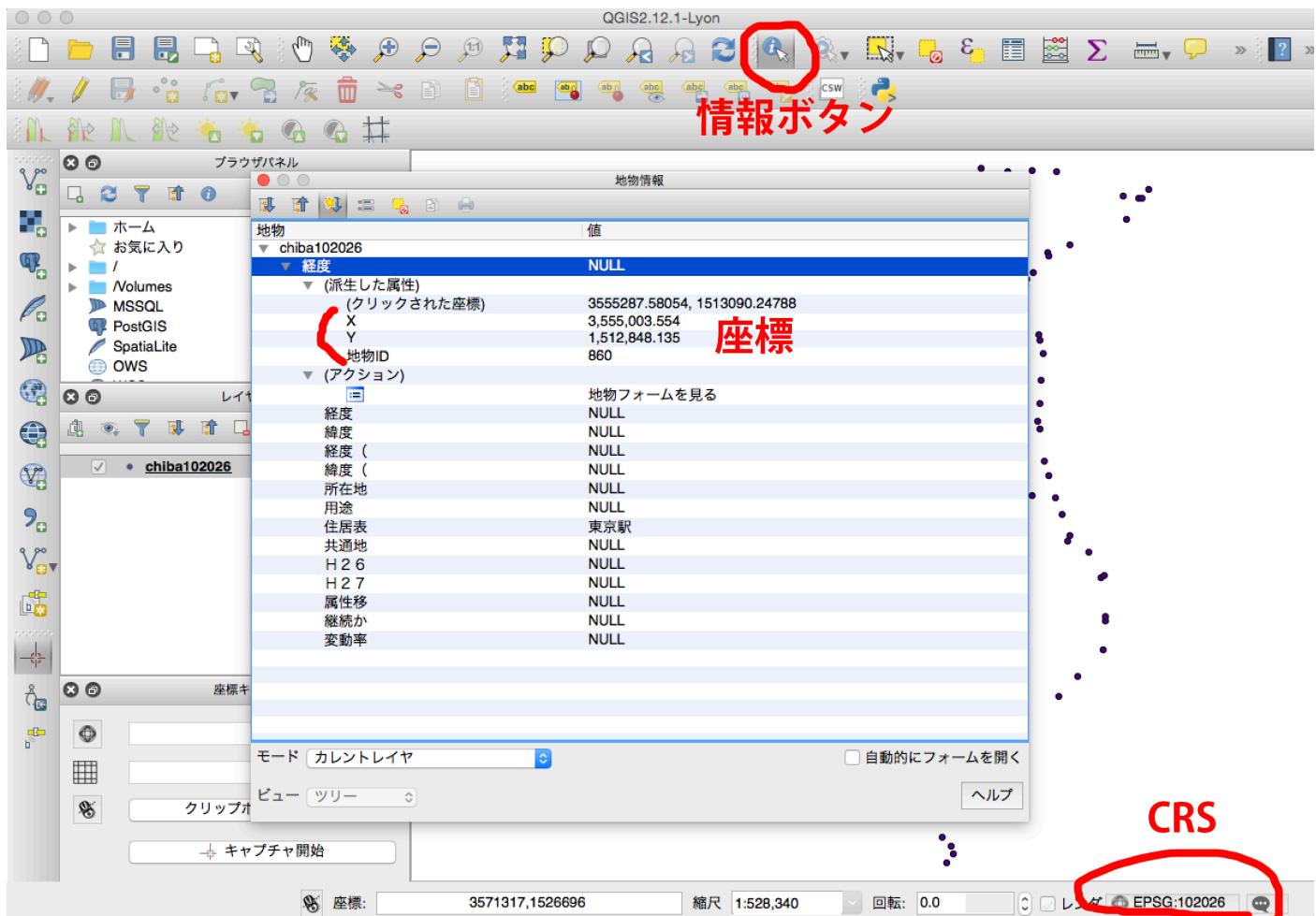
ID	経度	緯度	経度 (◆)	緯度 (◆)	所在地◆	用途	住居表◆	共通地◆	H 2 6 ◆	H 2 7 ◆	属性
843	504611.7770...	127414.2060...	140.1699380...	35.39283499...	12219	0	千葉県 市...	false	10300	10200	1000
844	505464.0729...	128933.2219...	140.4066869...	35.81478388...	12211	0	千葉県 成...	false	10200	10200	1000
845	504690.1599...	127723.7700...	140.1917111...	35.47882500...	12219	0	千葉県 市...	false	10100	10100	1000
846	505606.2880...	127898.4560...	140.4461911...	35.52734888...	12403	0	千葉県 山...	false	10100	10100	1000
847	505826.3379...	128373.0080...	140.5073161...	35.65916888...	12410	0	千葉県 山...	false	10100	10100	1000
848	505316.4299...	127168.0599...	140.3656750...	35.32446111...	12238	0	千葉県 い...	false	10000	10000	1000
849	505346.3750...	127186.9239...	140.3739930...	35.32970111...	12421	0	千葉県 長...	false	10100	10000	1000
850	505697.0109...	128010.3980...	140.4713919...	35.55844388...	12237	0	千葉県 山...	false	9950	9900	1000
851	503475.1560...	127125.5359...	139.8542099...	35.31264888...	12226	0	千葉県 富...	false	9900	9800	1000
852	505830.6500...	128203.0700...	140.5085138...	35.61196388...	12237	0	千葉県 山...	false	9400	9400	1000
853	505532.4079...	127820.1169...	140.4256688...	35.50558805...	12403	0	千葉県 山...	false	9100	9100	1000
854	505615.3200...	129039.2890...	140.4487000...	35.84424694...	12211	0	千葉県 成...	false	9200	9100	1110
855	505800.4459...	128128.2839...	140.5001238...	35.59118999...	12237	0	千葉県 山...	false	8550	8550	1000
856	505270.3969...	129156.0479...	140.3528880...	35.87668000...	12211	0	千葉県 成...	false	8550	8500	1000
857	505327.4280...	129058.7799...	140.3687299...	35.84966111...	12211	0	千葉県 成...	false	8050	8000	1000
858	505839.7760...	128231.6870...	140.5110488...	35.61991305...	12410	0	千葉県 山...	false	8100	8000	1000
859	504302.1010...	127256.3959...	140.0839169...	35.34899888...	12206	0	千葉県 木...	false	5150	5100	1000
860	NULL	NULL	NULL	NULL	NULL	NULL	東京駅	NULL	NULL	NULL	



距離を計算するためには、地図のCRS(地理参照系)を、等距離・メートル単位にしなければならない。ここでは、Asia_North_Equidistant_Conic(102026)に変換し、chiba102026.shpというファイル名で保存する。



再び開き、CRSが102026であることを確認。情報ボタンを押してから、東京駅をクリックすると、東京駅の情報が表示される。



CRSが変わったので、東京駅の座標はもやは緯度経度ではない。(3555003,1512848)であることをメモしておく。

レイヤを右クリックし、属性テーブルを開く。左上の鉛筆マークをクリックし、編集モードに移る（テーブルを変更可能にする）。

属性テーブル - 地価公示 :: 総地物数: 860, フィルター数: 860, 選択数: 1

	経度	緯度	経度 (◆)	緯度 (◆)	所在地◆	用途	住居表◆	共通地◆	H 2 6◆	H 2 7◆	属性
0	503734.2870...	128618.3579...	139.9261908...	35.72732166...	12203	0	千葉県 市...	true	326000	327000	1000
1	503756.7800...	128607.6199...	139.9324388...	35.72433888...	12203	0	千葉県 市...	false	310000	312000	1000
2	503604.2579...	128288.7430...	139.8900716...	35.63576194...	12227	0	千葉県 浦...	true	305000	306000	1000
3	503689.9359...	128360.8840...	139.9138711...	35.65580111...	12227	0	千葉県 浦...	false	304000	305000	1000
4	503679.3950...	128646.3819...	139.9109430...	35.73510611...	12203	0	千葉県 市...	false	298000	301000	1000
5	503638.3159...	128401.9109...	139.8995322...	35.66719750...	12227	0	千葉県 浦...	true	292000	296000	1000
6	503641.2739...	128390.2020...	139.9003538...	35.66394499...	12227	0	千葉県 浦...	false	292000	296000	1000
7	503693.8090...	128629.4290...	139.9149469...	35.73039694...	12203	0	千葉県 市...	false	293000	295000	1000
8	503730.5109...	128632.2979...	139.9251419...	35.73119388...	12203	0	千葉県 市...	false	288000	290000	1000
9	503755.3260...	128485.4260...	139.9320350...	35.69039611...	12203	0	千葉県 市...	false	288000	289000	1000
10	503939.3539...	128537.2189...	139.9831538...	35.70478305...	12204	0	千葉県 船...	false	280000	285000	1000
11	503685.9810...	128615.2470...	139.9127724...	35.72645750...	12203	0	千葉県 市...	true	280000	283000	1000
12	503714.2999...	128608.2759...	139.9206388...	35.72452111...	12203	0	千葉県 市...	false	282000	283000	1000
13	503664.6020...	128412.5720...	139.9068338...	35.67015888...	12203	0	千葉県 市...	false	280000	282000	1000
14	503657.6359...	128652.8399...	139.9048988...	35.73689999...	12203	0	千葉県 市...	false	279000	281000	1000
15	503644.4390...	128619.1940...	139.9012330...	35.72755388...	12203	0	千葉県 市...	false	276000	279000	1000
16	503633.8260...	128374.8010...	139.8982849...	35.65966694...	12227	0	千葉県 浦...	false	276000	278000	1000
<hr/>											
<hr/>											
<hr/>											

東京駅からの距離は、

```
distance(geom_from_wkt('POINT(3555003 1512848)'), $geometry)/1000
```

で計算されるので、「フィールド計算機」で新しい列「距離」を追加する。

フィールド計算機

選択されている0個の地物のみ更新する

新しいフィールドを作る

仮想フィールド作成

出力フィールド名

出力フィールドタイプ

出力フィールド幅 精度

式 関数エディタ

= + - / * ^ || ()

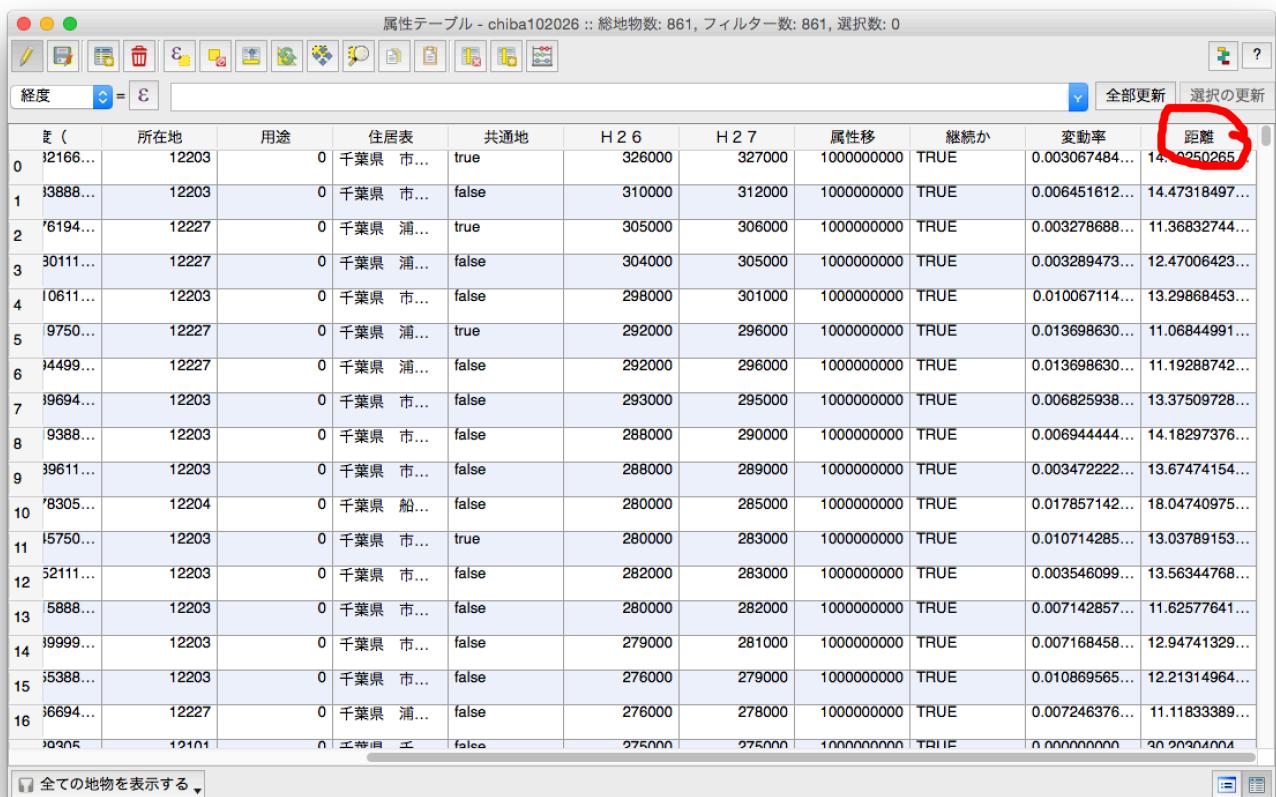
検索

row_number
Custom
▶ あいまい一致
▶ ジオメトリ
▶ フィールドと値
▶ 一
▶ 二

distance(geom_from_wkt('POINT(3555003 1512848)'), \$geometry)/1000

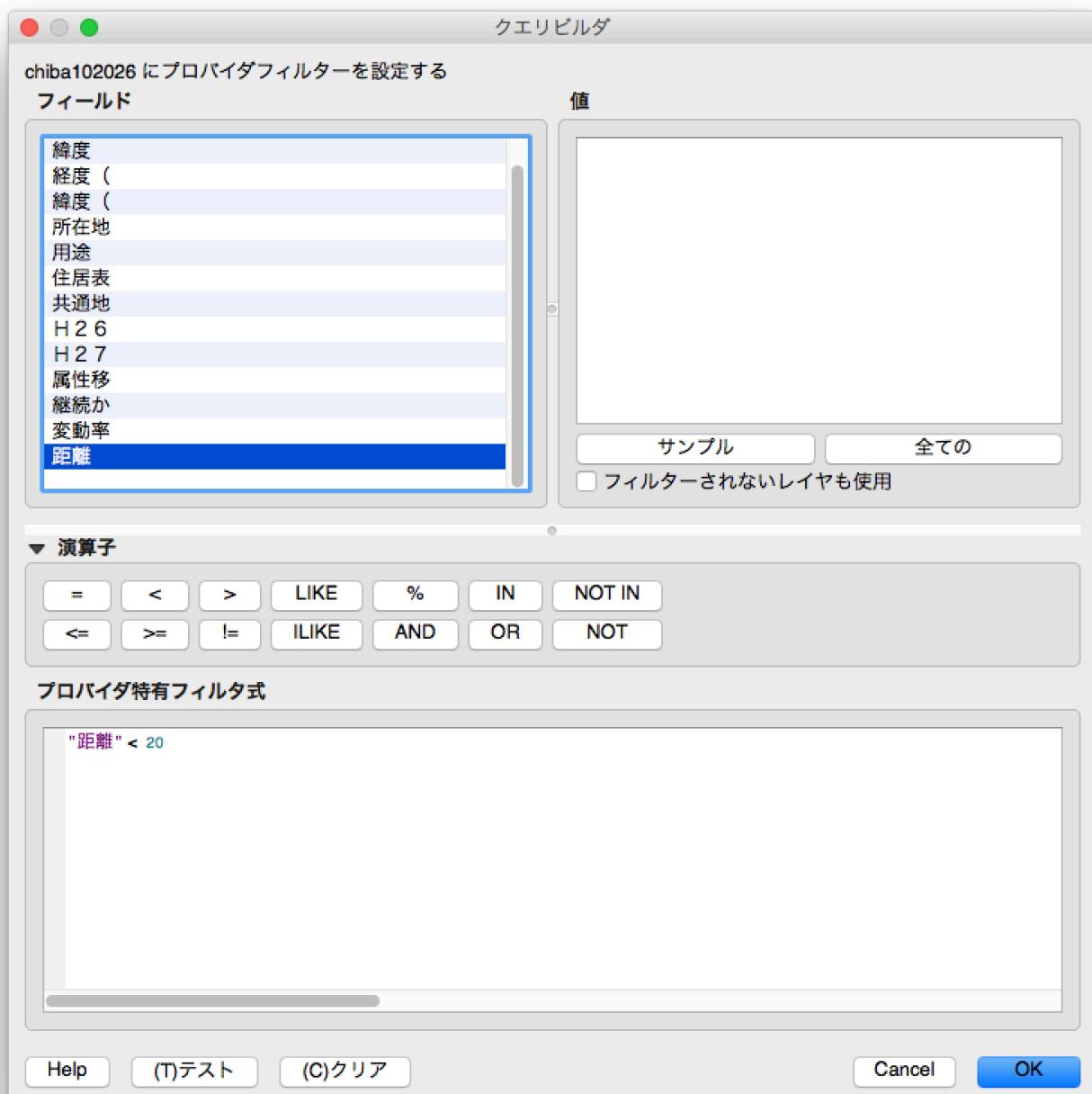
これで、すべての地点の東京駅距離(km)が表示される。

属性テーブル - chiba102026 :: 総地物数: 861, フィルター数: 861, 選択数: 0



レコード番号	所在地	用途	住居表示	共通地	H 2.6	H 2.7	属性移	継続か	変動率	距離
0	12166...	12203	0 千葉県 市...	true	326000	327000	1000000000	TRUE	0.003067484...	14.0250265...
1	13888...	12203	0 千葉県 市...	false	310000	312000	1000000000	TRUE	0.006451612...	14.47318497...
2	76194...	12227	0 千葉県 浦...	true	305000	306000	1000000000	TRUE	0.003278688...	11.36832744...
3	30111...	12227	0 千葉県 浦...	false	304000	305000	1000000000	TRUE	0.003289473...	12.47006423...
4	10611...	12203	0 千葉県 市...	false	298000	301000	1000000000	TRUE	0.010067114...	13.29868453...
5	9750...	12227	0 千葉県 浦...	true	292000	296000	1000000000	TRUE	0.013698630...	11.06844991...
6	14499...	12227	0 千葉県 浦...	false	292000	296000	1000000000	TRUE	0.013698630...	11.19288742...
7	19694...	12203	0 千葉県 市...	false	293000	295000	1000000000	TRUE	0.006825938...	13.37509728...
8	9388...	12203	0 千葉県 市...	false	288000	290000	1000000000	TRUE	0.006944444...	14.18297376...
9	39611...	12203	0 千葉県 市...	false	288000	289000	1000000000	TRUE	0.003472222...	13.67474154...
10	78305...	12204	0 千葉県 船...	false	280000	285000	1000000000	TRUE	0.017857142...	18.04740975...
11	15750...	12203	0 千葉県 市...	true	280000	283000	1000000000	TRUE	0.010714285...	13.03789153...
12	52111...	12203	0 千葉県 市...	false	282000	283000	1000000000	TRUE	0.003546099...	13.56344768...
13	5888...	12203	0 千葉県 市...	false	280000	282000	1000000000	TRUE	0.007142857...	11.62577641...
14	19999...	12203	0 千葉県 市...	false	279000	281000	1000000000	TRUE	0.007168458...	12.94741329...
15	55388...	12203	0 千葉県 市...	false	276000	279000	1000000000	TRUE	0.010869565...	12.21314964...
16	36694...	12227	0 千葉県 浦...	false	276000	278000	1000000000	TRUE	0.007246376...	11.11833389...
10000	19101	0 工業用 工	false	275000	275000	1000000000	TRUE	0.000000000...	20.20304004	

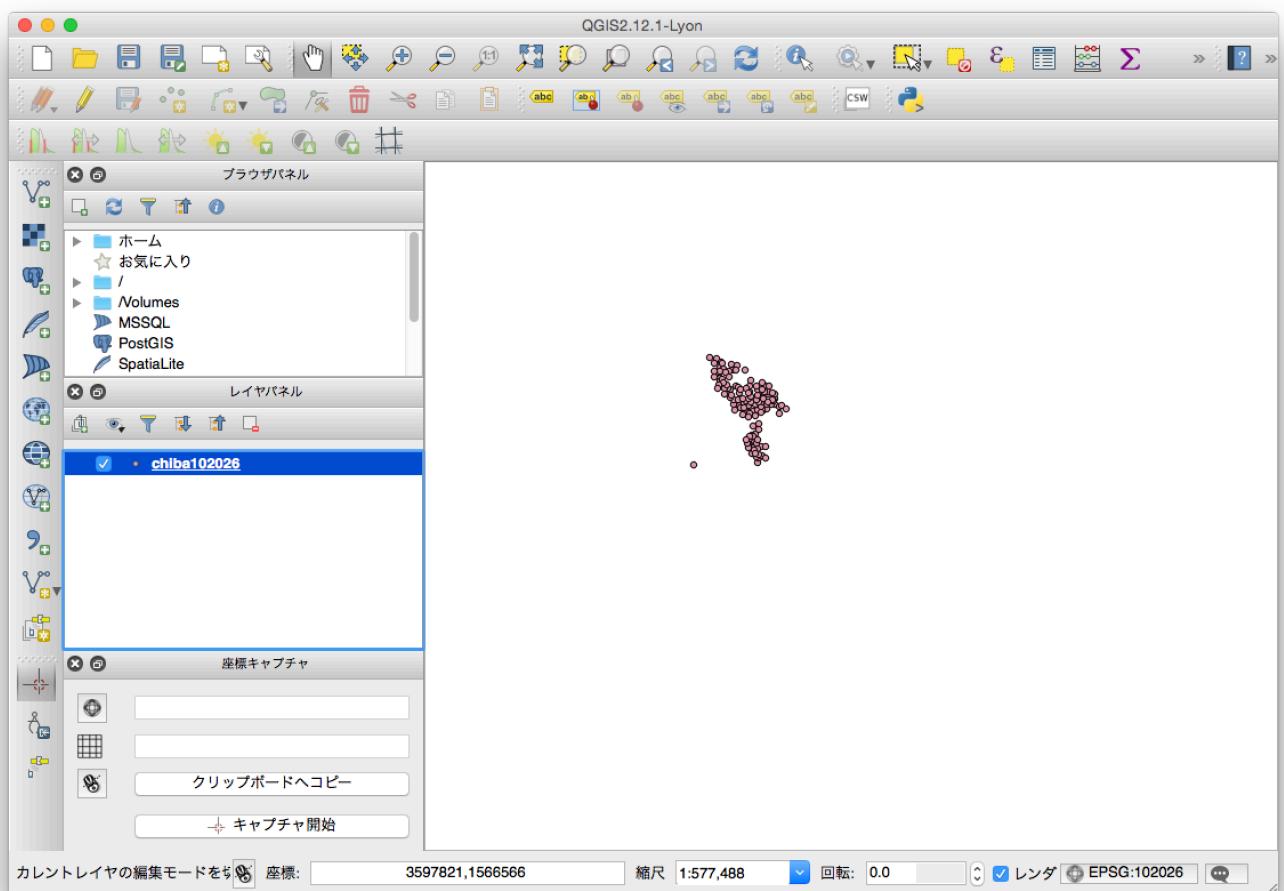
レイヤーを右クリックし、「フィルター」を選択すると、「クエリービルダー」が表示される。



東京駅から20キロ以内の地点だけに絞り込むには

'距離' < 20

を設定すればよい。たしかに、表示を絞りこむことができた。



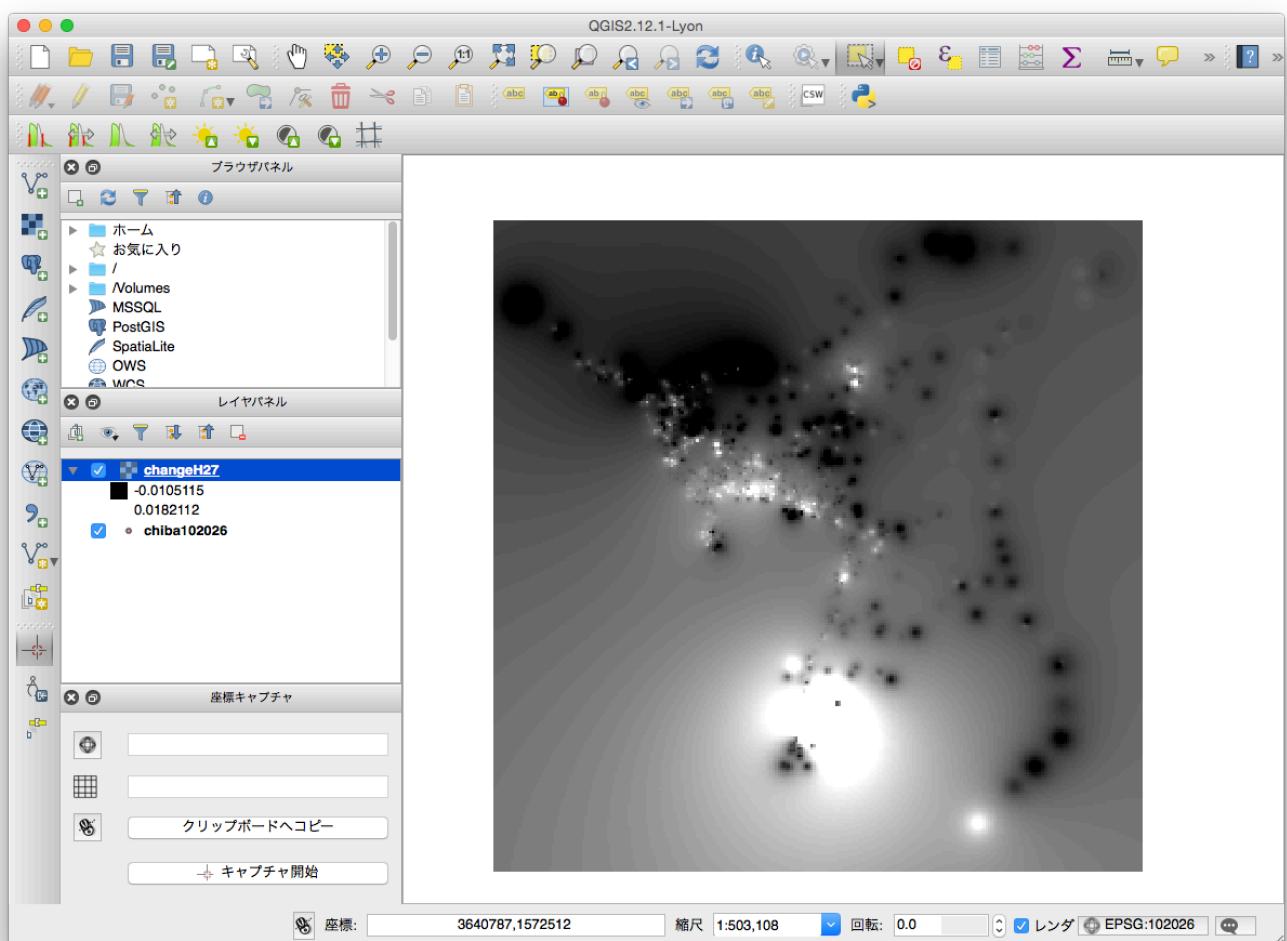
内挿(interpolation)

つぎに、東京駅の地点を削除し、ラスター▶解析▶グリッド（補間）を選ぶ。

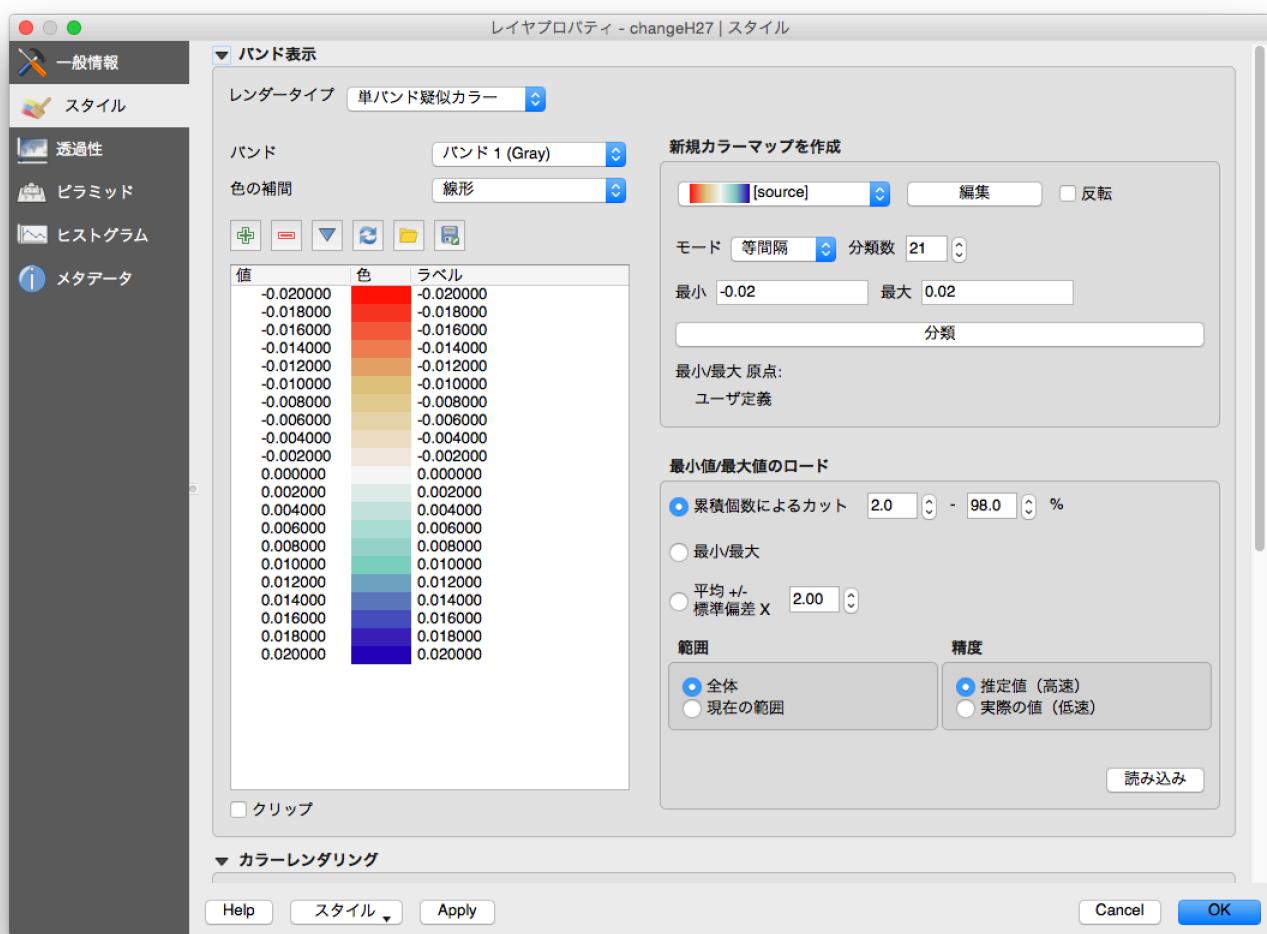
chiba102026.shpを選び、zフィールドを「変動率」にし、書き出すファイル名をchangeH27.tifにする。



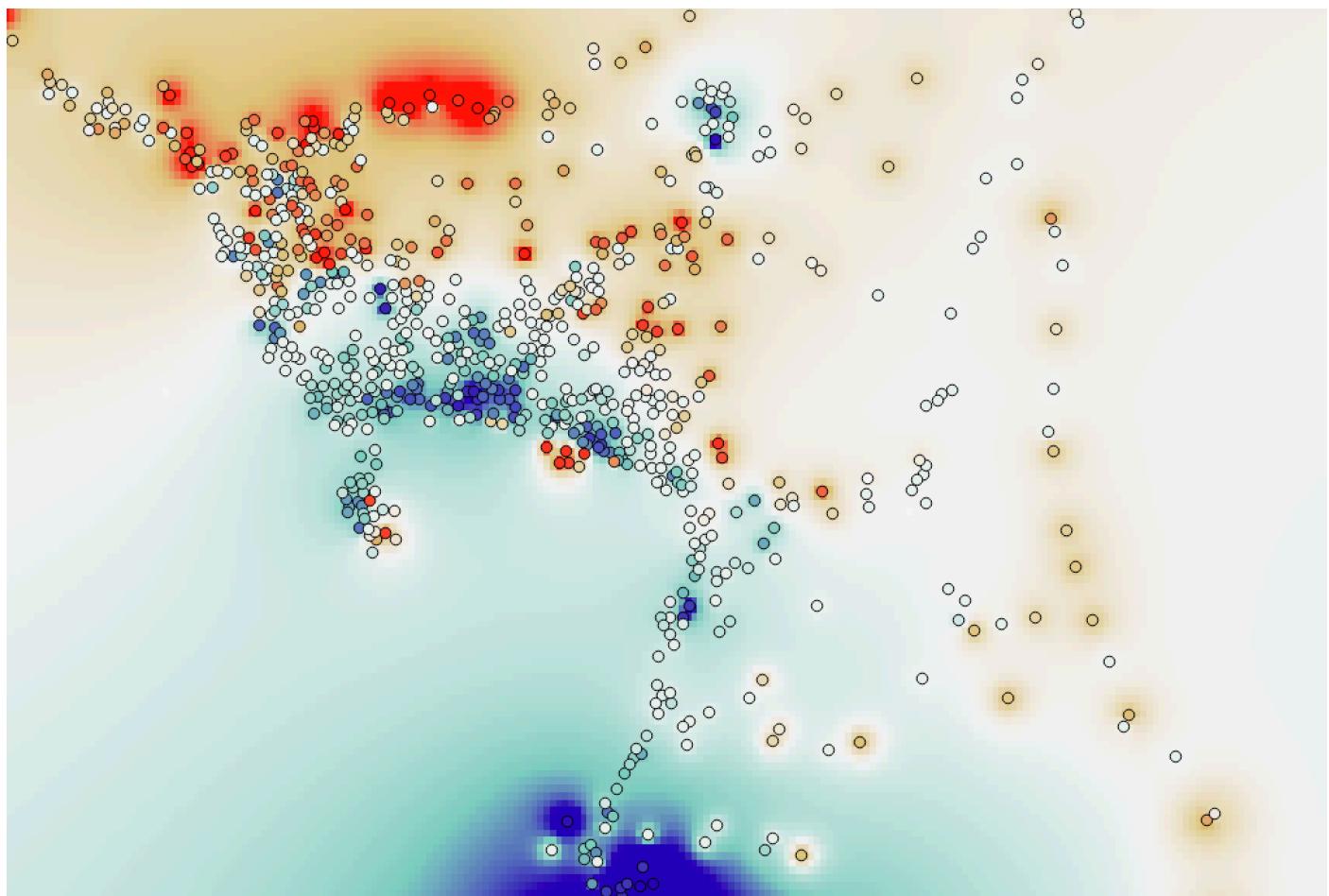
内挿された画像が表示される。



ラスター画像のプロパティで色分けをしてする。

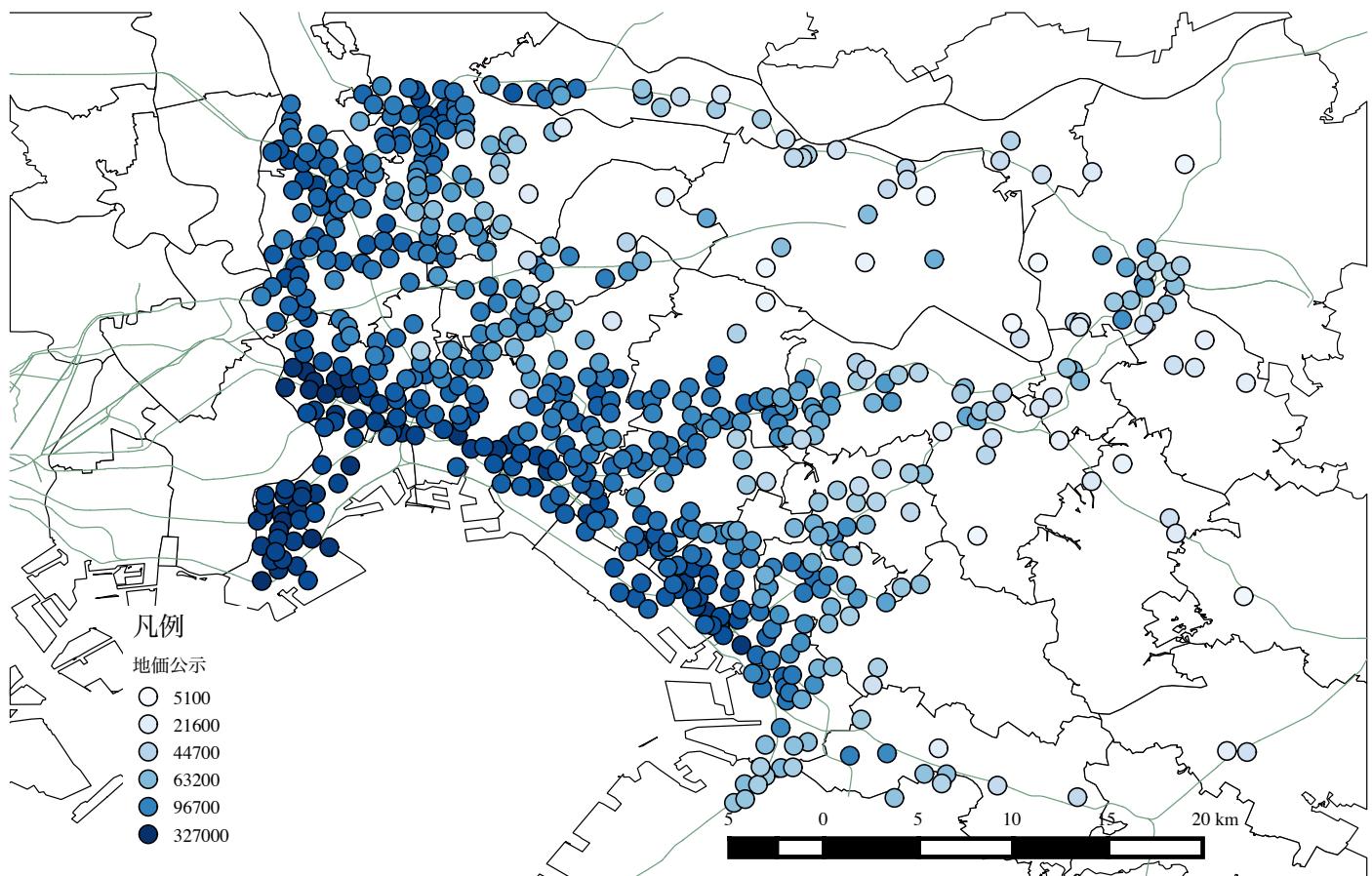


点データの色も揃えると、内挿の意図が分かりやすくなる。



プリントコンポーザー

プリントコンポーザーで地図、凡例、縮尺などを配置してPDF、SVGなどで書き出すことができる。



Python(プログラム)

公示データには、新設地点の情報は載っているが、廃止された地点の情報は欠落している。

廃止情報を得るために王道は、国交省に開示請求することだ。廃止情報がないのは、役所側の配慮である可能性がある。（公表データの背後にあるマスターデータを想像するセンスも必要になる）

廃止データが公表されない場合でも、通年データがあれば、前年のデータと比較し、欠落部分を抜き出すことで、廃止地点のリストができる。

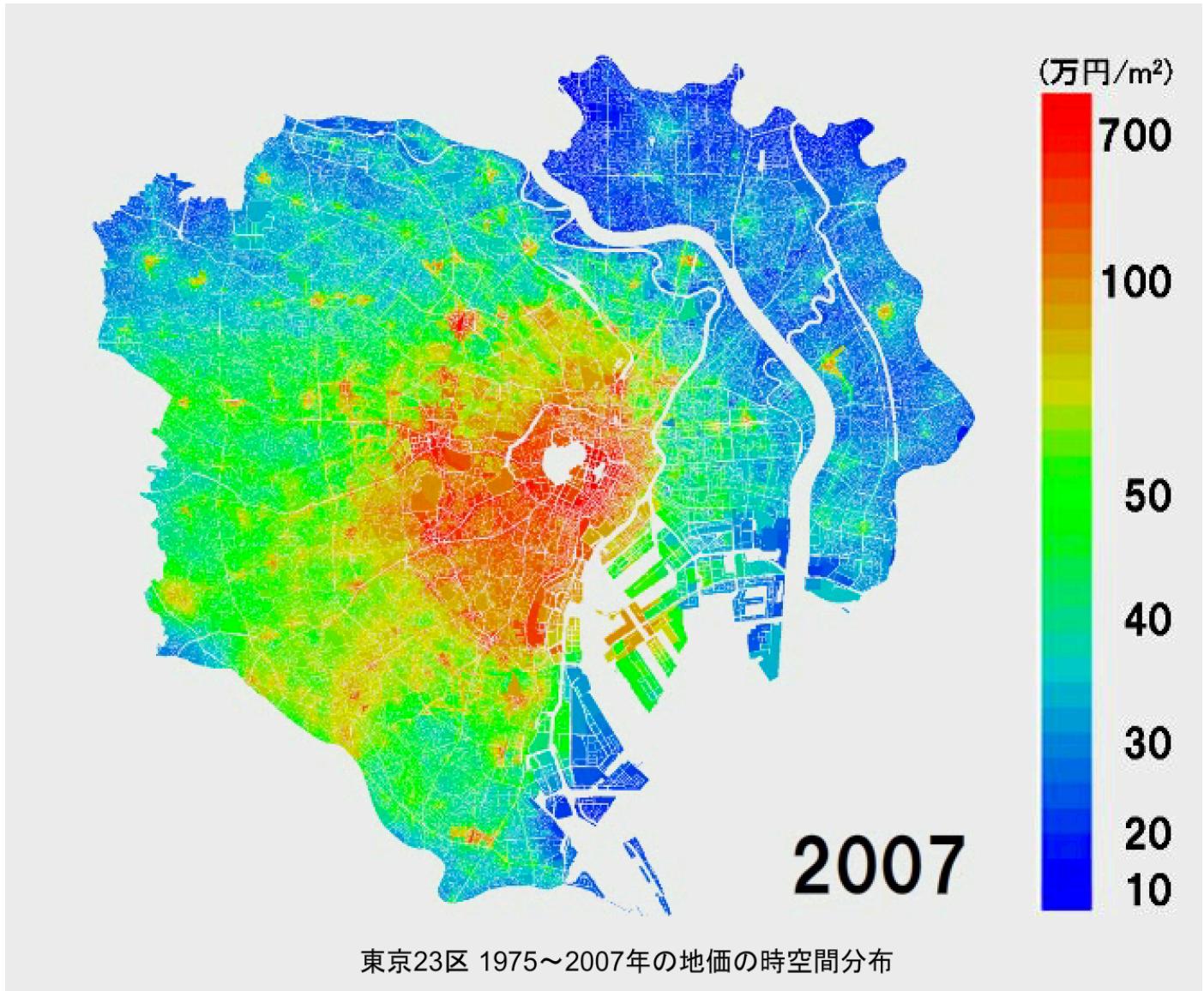
- もし、通年共通の地点識別子が設定されていれば、翌年データにはないものを抜き出せばよい
- この場合、識別子になりそうな要素は住所しかないが、市町村合併や地番変更に対応できない
- 緯度経度の情報から、同一地点かどうかを判断するほかない

この場合、プログラムで怪しいデータを抜き出し、最終的に人間の目で判断することにする。

(この結果、継続地点が新設地点としてコーディングされている例などが見つかるることを示す。03フォルダには、mySQL取り込みpythonコード例もある)

参考：高度な可視化

公示地価は、地理的な関係だけでなく、建ぺい率や道路に面しているかなどの個別事情によっても変わる。東北大気情報科学研究所の井上亮准教授は、それらの要素も考慮に入れた時空間内挿地図を作成している。大学院レベルの研究で、非常に難度が高い。<http://www.plan.civil.tohoku.ac.jp/inoue/research/kriging/>



デモ 1・空間検索

政府統計の総合窓口「eStat」<<http://e-stat.go.jp/SG2/eStatGIS/page/download.html#>>にはさまざまなデータがある。

国勢調査の男女別人口総数及び世帯総数を選ぶ。

The screenshot shows a web interface for selecting statistical tables from the e-Stat website. The URL is e-stat.go.jp/SG2/eStatGIS/page/download.html#.

Step1: 統計調査(集計)を選択

Category: 国勢調査

List of surveys:

- 平成22年国勢調査（小地域） 2010/10/01
- 平成22年国勢調査（国勢調査－世界測地系1kmメッシュ） 2010/10/01
- 平成22年国勢調査（国勢調査－世界測地系500mメッシュ） 2010/10/01
- 平成22年国勢調査（国勢調査－世界測地系250mメッシュ） 2010/10/01
- 平成17年国勢調査（小地域） 2005/10/01
- 平成17年国勢調査（国勢調査－世界測地系1kmメッシュ） 2005/10/01
- 平成17年国勢調査（国勢調査－世界測地系500mメッシュ） 2005/10/01
- 平成17年国勢調査（国勢調査－世界測地系250mメッシュ） 2005/10/01
- 平成12年国勢調査（小地域） 2000/10/01
- 平成12年国勢調査（国勢調査－世界測地系1kmメッシュ） 2000/10/01
- 平成12年国勢調査（国勢調査－世界測地系500mメッシュ） 2000/10/01

Step2: 統計表を選択(複数選択可能)

Selected items (indicated by a checked checkbox):

- 男女別人口総数及び世帯総数

Other available items (unchecked checkboxes):

- 年齢別(5歳階級、4区分)、男女別人口
- 世帯人員別一般世帯数
- 世帯の家族類型別一般世帯数
- 住宅の種類・所有の関係別一般世帯数
- 産業別(大分類)・従業上の地位別就業者数
- 職業別(大分類)就業者数
- 世帯の経済構成別一般世帯数

Buttons at the bottom:

- 統計表各種データダウンロードへ
- キャンセル

Page footer:

- GJ01060101
- Copyright(C) 2011 総務省 統計局 All rights reserved.
- ↑ このページのトップへ

千葉の全59市区町村のファイルと、定義書をダウンロードする。

Step4: データダウンロード

市区町村名をクリックして、統計データ、境界データをダウンロードして下さい。

名称	データ	定義書
千葉市中央区(3KB)		
千葉市花見川区(2KB)		
千葉市稲毛区(2KB)		
千葉市若葉区(3KB)		
千葉市緑区(2KB)		
千葉市美浜区(1KB)		
銚子市(3KB)		
市川市(5KB)		
船橋市(7KB)		
館山市(2KB)		

名称	データ	定義書
千葉市中央区(214KB)		
千葉市花見川区(118KB)		
千葉市稲毛区(60KB)		
千葉市若葉区(308KB)		
千葉市緑区(222KB)		
千葉市美浜区(47KB)		
銚子市(304KB)		
市川市(183KB)		
船橋市(367KB)		

[統計表検索へ戻る](#) [キャンセル](#)

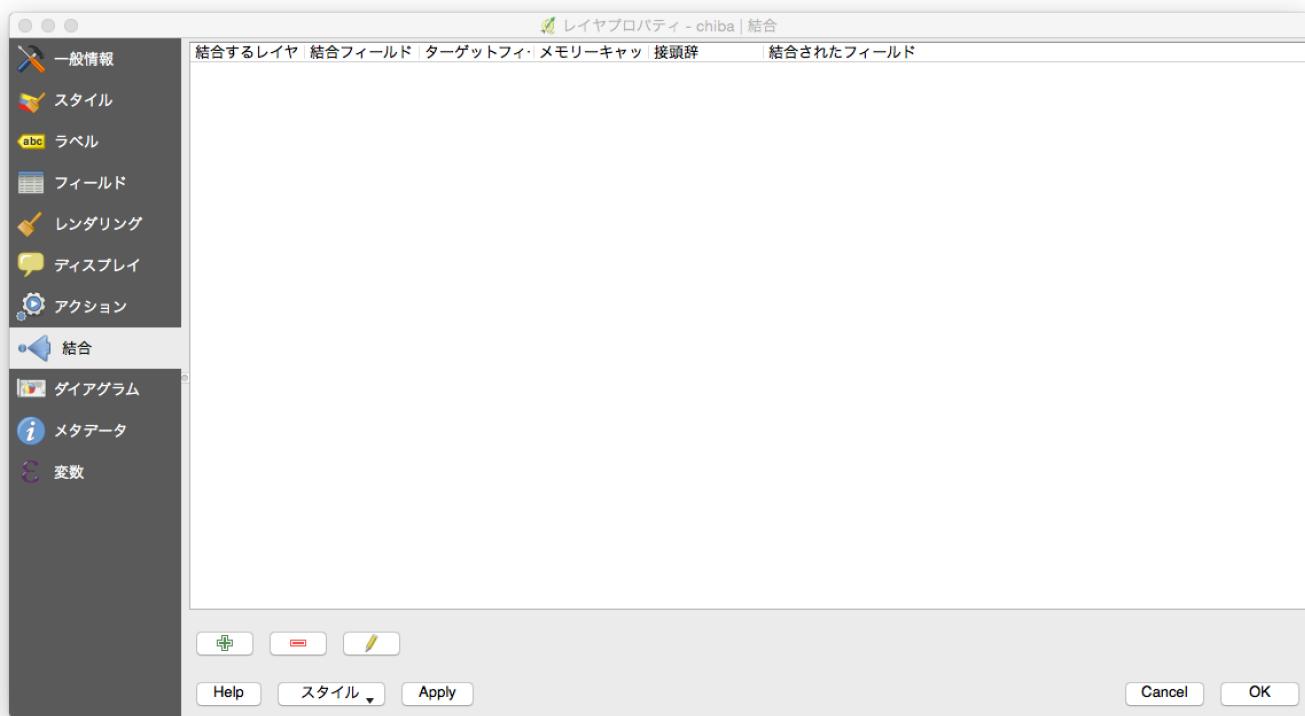
GJ01060102 [↑ このページのトップへ](#)

Copyright(C) 2011 総務省 統計局 All rights reserved.

(データは04/censusフォルダ、シェイプファイルは04/shapeフォルダにある)

Table Join

(中央区だけのデータを04/testフォルダに用意してある) 千葉市中央区のデータとシェープファイルを結びつける(Table join)。



シェル・スクリプト

千葉県の全市町村のデータをダウンロードすると、以下のような問題に突き当たる。

- データのテーブルを1本に統合したい
- シェープファイルを統合するため、同じディレクトリ（フォルダ）に集めたい

シェルスクリプトmergeCSV.shは、csvファイルを統合し、merged.csvをつくる。

```
targetFile="./census/merged.csv"
echo "KEY_CODE,HYOSY0,CITYNAME,NAME,HTKSYORI,HTKSAKI,GASSAN,T000572001,T000572002,T000572003,T
find ./census -name "*.txt" | while read -r f; do
    echo ${f}
    # 1から2行目までをd(無視)して、appendする
    sed 1,2d ${f} >> ${targetFile}
done
```

シェルスクリプトcollectShapefile.shは、ディレクトリallshpを作り、shapefileファイルを集める。

```
mkdir ./allshp
targetDirectory="./allshp/"
find ./shape -name "*.shp" | while read -r f; do
    this_path=$f
    # パスの分解
    string_filename=${this_path##*/}
    string_filename_without_extension=${string_filename%.*}
    string_path=${this_path%/*}
    string_extension=${this_path##*.}

    mv ${this_path} ${targetDirectory}${string_filename}
    mv ${string_path}/.${string_filename_without_extension}'.prj' ${targetDirectory}${string_filename}'.prj'
    mv ${string_path}/.${string_filename_without_extension}'.dbf' ${targetDirectory}${string_filename}'.dbf'
    mv ${string_path}/.${string_filename_without_extension}'.shx' ${targetDirectory}${string_filename}'.shx'

done
```

QGISでポリゴンを統合したchiba.shpを作り、フィールド計算機で面積を追加する。このファイルのCRSは30169である。

スクリーピング

TSUTAYAのホームページには、店舗情報がある。千葉県は4ページだけなので、手作業でコピーすればよい。しかし、もし400ページに分かれていたらどうするか？

店舗検索結果

都道府県[**千葉県**] を含む検索結果

選択した住所でヒットした地図を表示します。
周辺の地図を表示

下記の店舗の詳細情報を表示します。 市区町村を絞り込む 市区町村を選んでください ▾

21~40件/全72件 •前へ | 1 | 2 | 3 | 4 次へ

TSUTAYA 市川店 お店登録に追加する

住所 : 〒272-0026
千葉県市川市東大和田1-26-19
営業時間 : 深夜04:00~深夜04:00(24時間営業)

TSUTAYA 花見川店 お店登録に追加する

Elements Console Sources Network Timeline Profiles Resources Audits

View: XHR JS CSS Img Media Font Doc WS Other

Filter Hide data URLs All XHR JS CSS Img Media Font Doc WS Other

Name	Path
<input type="checkbox"/>	view_interface.php?classNa...
<input type="checkbox"/>	articleList?account=tsutay... as.chizumaru.com/tsutaya

2 / 139 requests | 25.0 KB / 4...

Console Emulation Rendering

webページからプログラムでデータを抜き出すことをscrapingという。以下は、scrapeTsutaya.pyの一部。

```
def main():
    ...
    店舗リストは以下のAPIでHTMLで送信されている
    http://as.chizumaru.com/tsutaya/articleList
    ?account=tsutaya&accmd=0&adr=12&searchType=True&pg=2&pageSize=20&pageLimit=10000&temp...
    このうち、千葉はpg=1から4まである
    ...
    # 出力ファイルはこのファイルと同じディレクトリに作成
    result_directory = os.path.dirname(os.path.abspath(__file__))
    fout_name = '%s/stores.csv' % (result_directory)
    fout = codecs.open(fout_name, 'w', 'utf_8')
```

```
print "Result file: %s" % fout_name
fout.write(u"Store,Address\n")

opener = urllib2.build_opener()
opener.addheaders = [
    ('User-Agent', 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.109 Safari/537.36'),
    ('Accept-Language', 'ja,en-us;q=0.7,en;q=0.3'),
    ('Origin', 'http://store-tsutaya.tsuite.jp'),
    ('Host', 'http://as.chizumaru.com'),
    ('DNT', '1'),
    ('Accept', 'application/html, text/html, */*; q=0.01'),
    ('Referer', 'http://store-tsutaya.tsuite.jp/storelocator/result_ad.html'),]

for nPage in range(1,5):
    targetURL = 'http://as.chizumaru.com/tsutaya/articleList?account=tsutaya&accmd=1&category=&sort=&page=' + str(nPage)

    try:
        remotef = opener.open( targetURL )
    except urllib2.URLError, e:
        if hasattr(e, 'reason'):
            print 'Fail to reach a server: ', e.reason
        elif hasattr(e, 'code'):
            print 'Request Error: ', e.code
    else:
        html = unicode(remotef.read(), "utf-8")
        d = BeautifulSoup(html, "lxml")
        for iTable, Table in enumerate( d.find_all("table") ):
            if not Table.has_attr('summary'):
                continue
            Store = unicode(Table.find('td', class_="fb").a.string)
            print type(Store), Store
            TdAddress = Table.find('td', class_="wh")
            if TdAddress:
                content = TdAddress.get_text()
                # print type(Address)
                # print Address
                Address = content.split( u"\n" )[2]
                print type(Address), Address
                fout.write( u"\r\n".format( Store, Address.strip() ) )
            else:
                print "not found td", nPage, Store
fout.close()
```

これによって、stores.csvが生成される。

	A	B
1	Store	Address
2	TSUTAYA 松戸駅前店	千葉県松戸市本町 4 - 18
3	TSUTAYA 市川大野店	千葉県市川市南大野2-4-26
4	TSUTAYA 土気店	千葉県千葉市緑区あすみが丘東2丁目26番地7
5	TSUTAYA 都賀店	千葉県千葉市若葉区都賀3丁目31番5号
6	TSUTAYA 行徳店	千葉県市川市閑ヶ島16-2
7	TSUTAYA 幕張本郷店	千葉県千葉市花見川区幕張本郷6-22-22
8	すばる書店 TSUTAYA 六高台店	千葉県松戸市六実1-14-1
9	TSUTAYA 稲毛海岸駅前店	千葉県千葉市美浜区高洲3-10-1
10	TSUTAYA 夏見台店	千葉県船橋市夏見台5丁目16-1
11	すばる書店 TSUTAYA 松戸栄町店	千葉県松戸市栄町西2丁目865
12	すばる書店 TSUTAYA 二十世紀が丘店	千葉県市川市堀之内3-31-16
13	すばる書店 TSUTAYA 南行徳店	千葉県市川市相之川3-13-23 丸伝小川ビル
14	TSUTAYA 千葉寺店	千葉県千葉市中央区千葉寺町971-7
15	TSUTAYA 東千葉祐光店	千葉県千葉市中央区祐光3-2
16	TSUTAYA 稲毛店	千葉県千葉市稲毛区小仲台6-15-3 フルールビル1階
17	TSUTAYA 本八幡駅前店	千葉県市川市八幡2丁目16-6 ハタビル2階
18	TSUTAYA 市川オリンピック店	千葉県市川市市川1-5-17 B1F

ジオ・エンコーディング

店舗の住所情報に、緯度経度情報を付加したい。Google/YahooにはGeoEncodingサービスがある。以下は、locateTsutaya.pyの一部。

```
def getLonLat(local_address):
    opener = urllib2.build_opener()
    opener.addheaders = [
        ('User-agent', 'Python2.4.3'),
        ('Accept-Language', 'ja,en-us;q=0.7,en;q=0.3'),
        ('Content-Type', 'application/json') ]
    query = [
        ( "address", local_address.encode('utf-8') ),
        ( "key", GOOGLE_ID ),
    ]
    targetURL = GOOGLE_API + urllib.urlencode(query)
    try:
        response = opener.open( targetURL )
    except urllib2.URLError, e:
        if hasattr(e, 'reason'):
            print 'Fail to reach a server: ', e.reason
        elif hasattr(e, 'code'):
            print 'Request Error: ', e.code
    else:
        res = json.loads( response.read() )
        if res[u'status'] != u'OK':
            print "RESPONSE ERROR", res
            return (0, 0)
        else:
            geo = res[u'results'][0][u'geometry'][u'location']
            print "geoLonLat:",local_address.encode('utf-8'), geo[u'lng'], geo[u'lat']
```

```

        return (geo[u'lng'], geo[u'lat'])

def main():
    # 出力ファイルはこのファイルと同じディレクトリに作成
    result_directory = os.path.dirname(os.path.abspath(__file__))
    fin_name = '%s/stores.csv' % (result_directory)
    fout_name = '%s/stores_geolocated.csv' % (result_directory)
    fin = codecs.open(fin_name, 'r', 'utf_8')
    fout = codecs.open(fout_name, 'w', 'utf_8')
    print "Result file: %s" % fout_name

    fout.write(u"Store,Address,Lon,Lat\n")
    for nLine, Line in enumerate(fin):
        time.sleep(0.5)
        if nLine == 0:
            continue
        else:
            items = Line.split(u",")
            Store, Address = items[0], items[1]
            Lon, Lat = getLonLat(Address.strip())
            fout.write(u"\t{0:s},{1:s},{2:.6f},{3:.6f}\n".format(Store.strip(), Address, Lon, Lat))

    fin.close()
    fout.close()

```

これによって、stores_geolocated.csvという緯度経度情報付きcsvファイルが生成される。

	A	B	C	D
1	Store	Address	Lon	Lat
2	TSUTAYA 松戸駅前店	千葉県松戸市本町 4 - 1 8	139.899018	35.784397
3	TSUTAYA 市川大野店	千葉県市川市南大野2-4-26	139.952599	35.744575
4	TSUTAYA 土気店	千葉県千葉市緑区あすみが丘東2丁目26番地7	140.273772	35.52252
5	TSUTAYA 都賀店	千葉県千葉市若葉区都賀3丁目31番5号	140.153033	35.637016
6	TSUTAYA 行徳店	千葉県市川市関ケ島16-2	139.917906	35.687558
7	TSUTAYA 幕張本郷店	千葉県千葉市花見川区幕張本郷6-22-22	140.045583	35.675159
8	すばる書店 TSUTAYA 六高台店	千葉県松戸市六実1-14-1	139.987044	35.793479
9	TSUTAYA 稲毛海岸駅前店	千葉県千葉市美浜区高洲3-10-1	140.071701	35.628315
10	TSUTAYA 夏見台店	千葉県船橋市夏見台5丁目16-1	139.992414	35.724916
11	すばる書店 TSUTAYA 松戸栄町店	千葉県松戸市栄町西2丁目865	139.895398	35.809544
12	すばる書店 TSUTAYA 二十世紀が丘店	千葉県市川市堀之内3-31-16	139.912816	35.763883
13	すばる書店 TSUTAYA 南行徳店	千葉県市川市相之川3-13-23 丸伝小川ビル	139.898296	35.676873
14	TSUTAYA 千葉寺店	千葉県千葉市中央区千葉寺町971-7	140.132217	35.590003
15	TSUTAYA 東千葉祐光店	千葉県千葉市中央区祐光3 - 2	140.127144	35.619922
16	TSUTAYA 稲毛店	千葉県千葉市稲毛区小仲台6 - 1 5 - 3 フルールビル↑	140.092962	35.639337
17	TSUTAYA 本八幡駅前店	千葉県市川市八幡2丁目16 - 6 ハタビル2階	139.926774	35.721553
18	TSUTAYA 市川オリンピック店	千葉県市川市市川1 - 5 - 1 7 B 1 F	139.909331	35.729757
19	TSUTAYA 船橋南口駅前店	千葉県船橋市本町 4 - 4 1 - 2 5 ステージ船橋 2 F	139.986451	35.698952

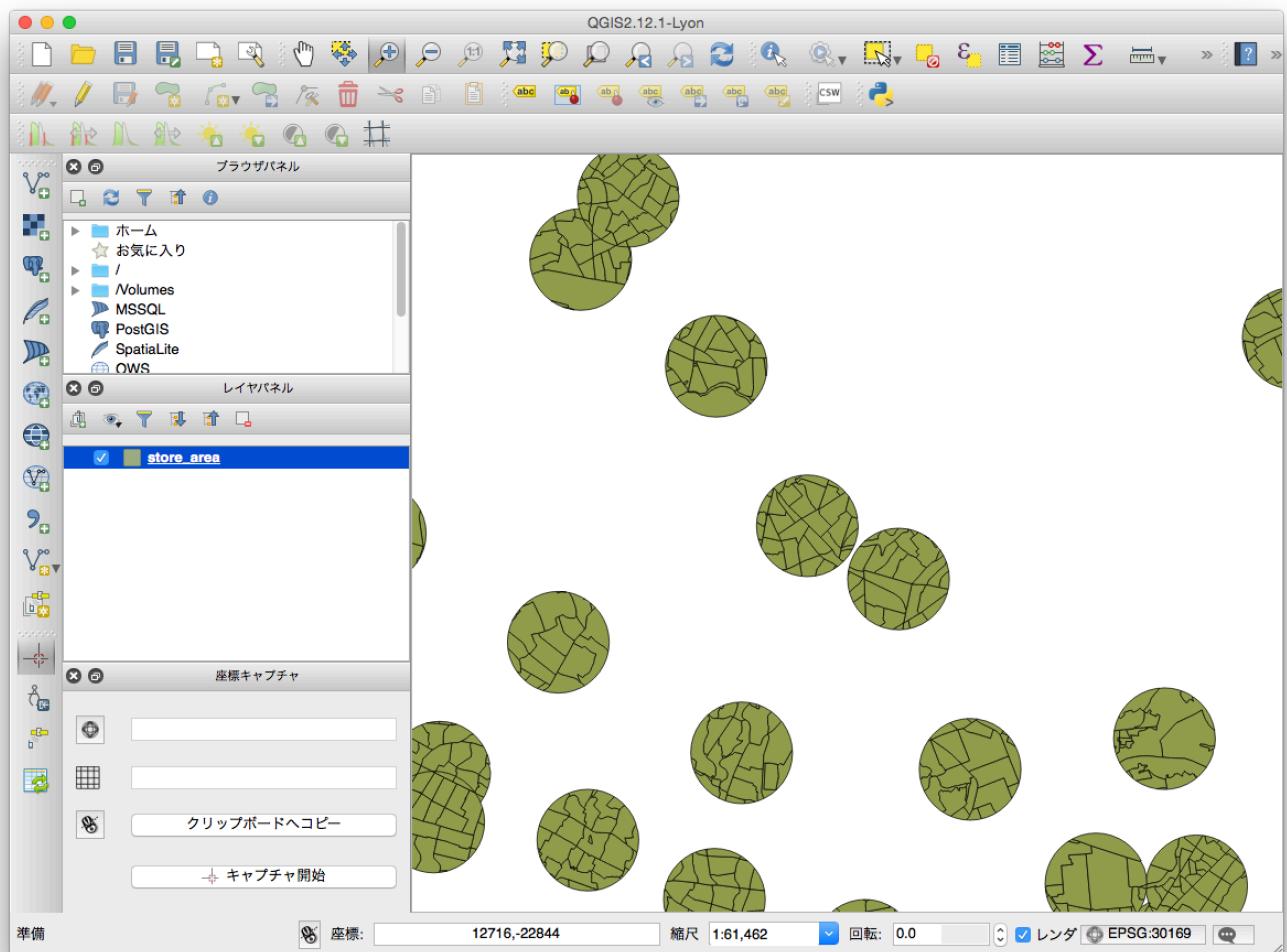
QGISで点情報のシェープファイルstore.shpをCRS=4326(WGS84)で作り、さらにCRS=30169でstore_30169.shpを作る。

バッファーの生成

ベクタ▶空間演算ツール▶バッファで直径1000=1kmのベクターファイルstore_buffer.shpを生成する。

交差ポリゴンの生成

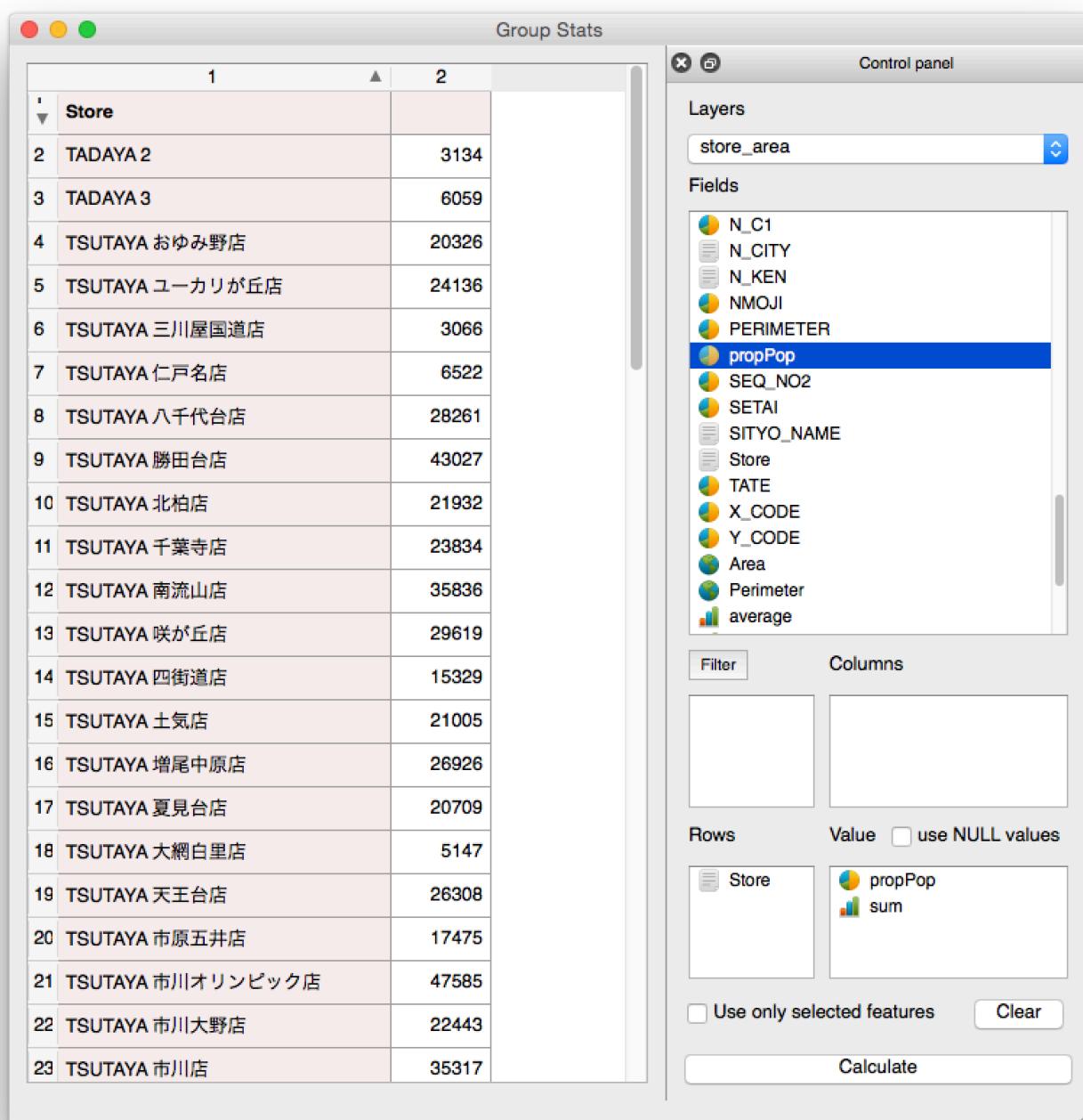
ベクタ▶空間演算ツール▶交差でベクターファイルstore_area.shpを作る。



フィールド計算機で、推定人口（面積比×人口）を計算したものpropPopを追加する。

プラグインGroup Stat

ベクタ▶GroupStatで、以下のように、店舗単位で推定人口を集計する。



ファイルfinal_result.csvを書き出すと以下のようになる。

	A	B	C	D	E
13	TSUTAYA 咲が丘店	29619			
14	TSUTAYA 四街道店	15329			
15	TSUTAYA 土気店	21005			
16	TSUTAYA 増尾中原店	26926			
17	TSUTAYA 夏見台店	20709			
18	TSUTAYA 大網白里店	5147			
19	TSUTAYA 天王台店	26308			
20	TSUTAYA 市原五井店	17475			
21	TSUTAYA 市川オリンピック店	47585			
22	TSUTAYA 市川大野店	22443			
23	TSUTAYA 市川店	35317			
24	TSUTAYA 幕張本郷店	32971			
25	TSUTAYA 成田空港第1ターミナル店	16			
26	TSUTAYA 成田赤坂店	20060			
27	TSUTAYA 木下店	4234			
28	TSUTAYA 木更津店	13233			
29	TSUTAYA 本八幡駅前店	45404			
30	TSUTAYA 東千葉祐光店	27166			

デモ 2・統計学

人口動態統計には、主要死因の市町村別(保健所別)データがある。

人口動態保健所・市区町村別統計: <http://www.e-stat.go.jp/SG1/estat/GL08020103.do?_toGL08020103_&tclassID=000001052136&requestSender=search>

e-Stat 政府統計の総合窓口

ホーム | お問い合わせ | ヘルプ | English | 文字拡大・読み上げ

統計データを探す 地図や図表で見る 調査項目を調べる 統計サイト検索・リンク集 ログイン

トップページ > 統計データを探す > 統計表一覧

統計表一覧

各行にある Excel CSV PDF DB のボタンを押すと該当データが表示されます。

平成20～24年 人口動態保健所・市区町村別統計

作成機関

表番号	統計表	
概要		
1	人口動態保健所・市区町村別統計の概要	PDF
2	用語の解説等	PDF
統計表		
1	人口動態総覧（数・率）・人口、都道府県・保健所・市区町村別	Excel
2	合計特殊出生率・母の年齢階級別出生率、都道府県・保健所・市区町村別	Excel
3	死亡数、主要死因・性・都道府県・保健所・市区町村別	Excel
4	死亡率（男性・女性人口10万対）、主要死因・性・都道府県・保健所・市区町村別	Excel
5	標準化死亡比、主要死因・性・都道府県・保健所・市区町村別	Excel
sankou	東日本大震災による死亡を除いた場合の参考値（死亡数・死亡率（男性・女性人口10万対）・標準化死亡比・標準化死亡比（ペイズ推定値）・都道府県・保健所・市区町村別）	Excel

GL08020103

[↑ このページのトップへ](#)

(05フォルダにh20-24_第3表.xlsxがある)

Open Officeで開き、以下のような「カスタム関数」を定義し、市町村コードを抜き出す。また、「女性の死亡数」と「女性の肝疾患」を抜き出す。

```
Function getCode(strTarget, strReg as String) as String
    rem 文字列の中のCODE(半角数字だけで表現される)だけを抜き出すカスタム関数
    rem 全角空白があるとエラーが出るので、事前に半角空白に置換しておくこと
    rem 参考：https://wiki.openoffice.org/wiki/JA/Documentation/BASIC\_Guide/Strings\_\(Runtime\_Language\)
    Dim oTextSearch as Object
    Dim oOption as Object
    oTextSearch = CreateUnoService("com.sun.star.util.TextSearch")
    oOption = CreateUnoStruct("com.sun.star.util.SearchOptions")
    ' 正規表現を使う
    With oOption
        .algorithmType = com.sun.star.util.SearchAlgorithms.REGEXP
        .searchFlag = com.sun.star.util.SearchFlags.REG_EXTENDED
        .searchString = strReg
    End With
    ' 検索の実行
    Dim oStartPos as Long
    Dim oLength as Long
    Dim oResult as Object
    oResult = oTextSearch.searchForward(strTarget, 0, Len(strTarget) -1)
    ' 返値の定義：https://www.openoffice.org/api/docs/common/ref/com/sun/star/util/SearchResults.html
    If oResult.subRegExpressions = 1 Then
        oStartPos = oResult.startOffset(0)
        oLength = oResult.endOffset(0) - oResult.startOffset(0)
        ' offsetは0からスタート、Midは1からスタート
        getCode = Mid(strTarget, oStartPos+1, oLength)
    Else

```

```

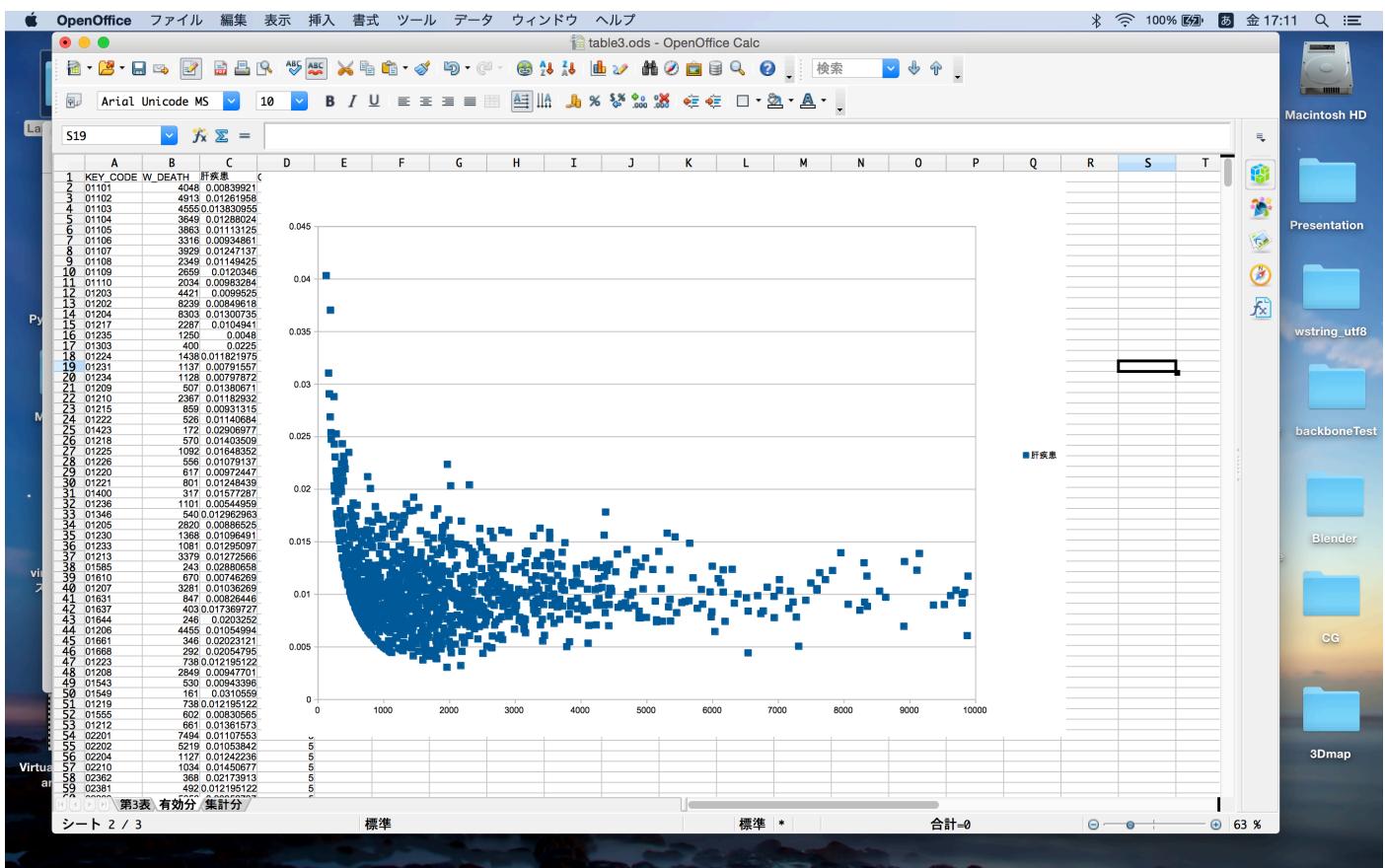
        getCode = "N.A."
End If
rem 関数の戻り値は関数名に値を設定する方法をとる (returnという命令がない)
End Function

```

table3.ods -

	A	B	C	D	E	F	G	H	I
1	平成20年～平成24年				人口動態保健所・市町村別統計				
2	第3表				死亡数、主要死因・性・都道府県・保健所・市区町村別(平)				
3									
4					死亡総数		悪性新生物		
5							総数	胃	
6		KEY CODE	W DEATH	肝疾患	男性	女性	男性	女性	男性
7	全国				3155767	2825402	1051468	706246	163508
8	01北海道				148558	128058	52338	35902	7193
9	10 札幌市保健所				41193	35315	14863	10737	2008
10	札幌市				41193	35315	14863	10737	2008
11	01101 中央区	01101	4048	0.0083992	4303	4048	1582	1330	202
12	01102 北区	01102	4913	0.0126196	5994	4913	2207	1514	309
13	01103 東区	01103	4555	0.013831	5473	4555	1921	1336	270
14	01104 白石区	01104	3649	0.0128802	4590	3649	1581	1141	235
15	01105 豊平区	01105	3863	0.0111312	4441	3863	1584	1182	195
16	01106 南区	01106	3316	0.0093486	3794	3316	1361	962	160
17	01107 西区	01107	3929	0.0124714	4572	3929	1690	1238	230
18	01108 厚別区	01108	2349	0.0114943	2680	2349	998	695	129
19	01109 手稲区	01109	2659	0.0120346	3108	2659	1131	750	156
20	01110 清田区	01110	2034	0.0098328	2238	2034	808	589	122

このままOpenOfficeで分析を続けると、以下のようなグラフまでたどり着ける。（欠損値の市町村は除外する）



R(統計処理ソフト)

ここでは、結果をcsv形式で書き出し(table3.csv)、R(統計処理ソフト)で読み込むことにする。

RにはR専用のスクリプト言語がある。様々な統計手法パッケージがあり、グラフィックスも描きやすいのでデータ処理に利用されている。

(05フォルダにR_demo.Rがある。Rのカレントディレクトリを05にする必要がある)

```
d <- read.table("result/table3.csv", header=TRUE, sep=",")
total <- d$W_TOTAL[!(is.na(d$DEATH))&!(is.na(d$W_TOTAL))&(d$LEN > 3)]
death <- d$DEATH[!(is.na(d$DEATH))&!(is.na(d$W_TOTAL))&(d$LEN > 3)]
ratio <- d$RATIO[!(is.na(d$DEATH))&!(is.na(d$W_TOTAL))&(d$LEN > 3)]

plot(total,ratio, xlim = c(0, 15000), ylim = c(0, max(ratio)),
     main = "death by liver disease", xlab = "death", ylab = "ratio")

# 二項分布による想定
averageRate <- sum(death)/sum(total)
# 関数はベクトルを受けとり、ベクトルを返さなければならない
error95upper <- function(sampleSize){
  return (sqrt( averageRate*(1-averageRate)/sampleSize )*1.96 + averageRate)
}
error99upper <- function(sampleSize){
  return (sqrt( averageRate*(1-averageRate)/sampleSize )*2.56 + averageRate)
}
error95lower <- function(sampleSize){
  return (sqrt( averageRate*(1-averageRate)/sampleSize )*-1.96 + averageRate)
}
error99lower <- function(sampleSize){
  return (sqrt( averageRate*(1-averageRate)/sampleSize )*-2.56 + averageRate)
}
```

```

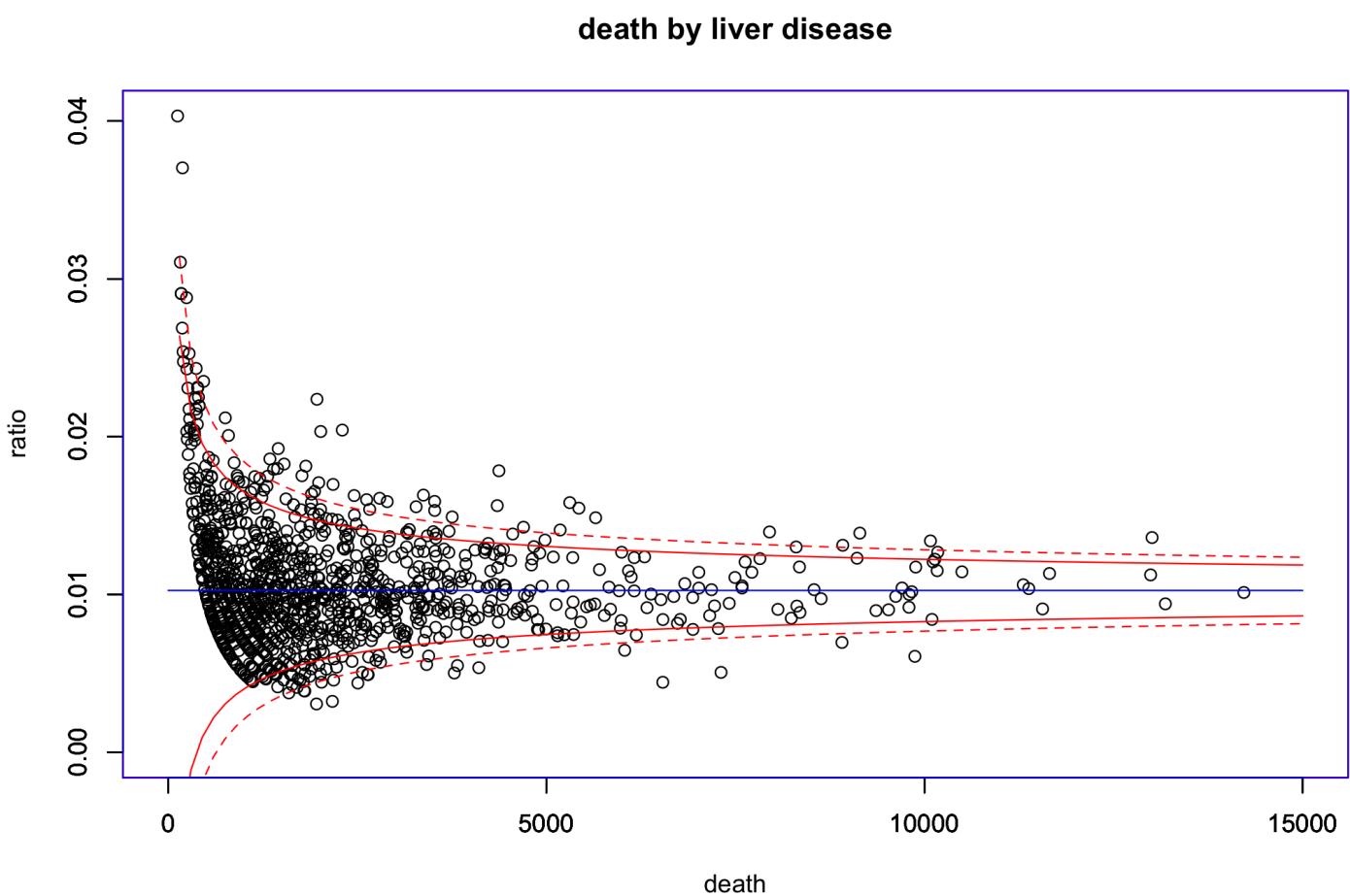
averageLine <- function(sampleSize){
  # ベクトルを返すように細工する
  return (averageRate + sampleSize*0.0)
}

par(new=T,col=2,lty="solid")
plot( error95upper, xlim = c(0, 15000), ylim = c(0, max(ratio)), main = "", xlab = "", ylab =
par(new=T,lty="dashed")
plot( error99upper, xlim = c(0, 15000), ylim = c(0, max(ratio)), main = "", xlab = "", ylab =
par(new=T,lty="solid")
plot( error95lower, xlim = c(0, 15000), ylim = c(0, max(ratio)), main = "", xlab = "", ylab =
par(new=T,lty="dashed")
plot( error99lower, xlim = c(0, 15000), ylim = c(0, max(ratio)), main = "", xlab = "", ylab =
par(new=T,col=4,lty="solid")
plot( averageLine, xlim = c(0, 15000), ylim = c(0, max(ratio)), main = "", xlab = "", ylab =

# eps出力の場合
# dev.copy2eps( file="table.eps", width=6 )
# dev.off()

```

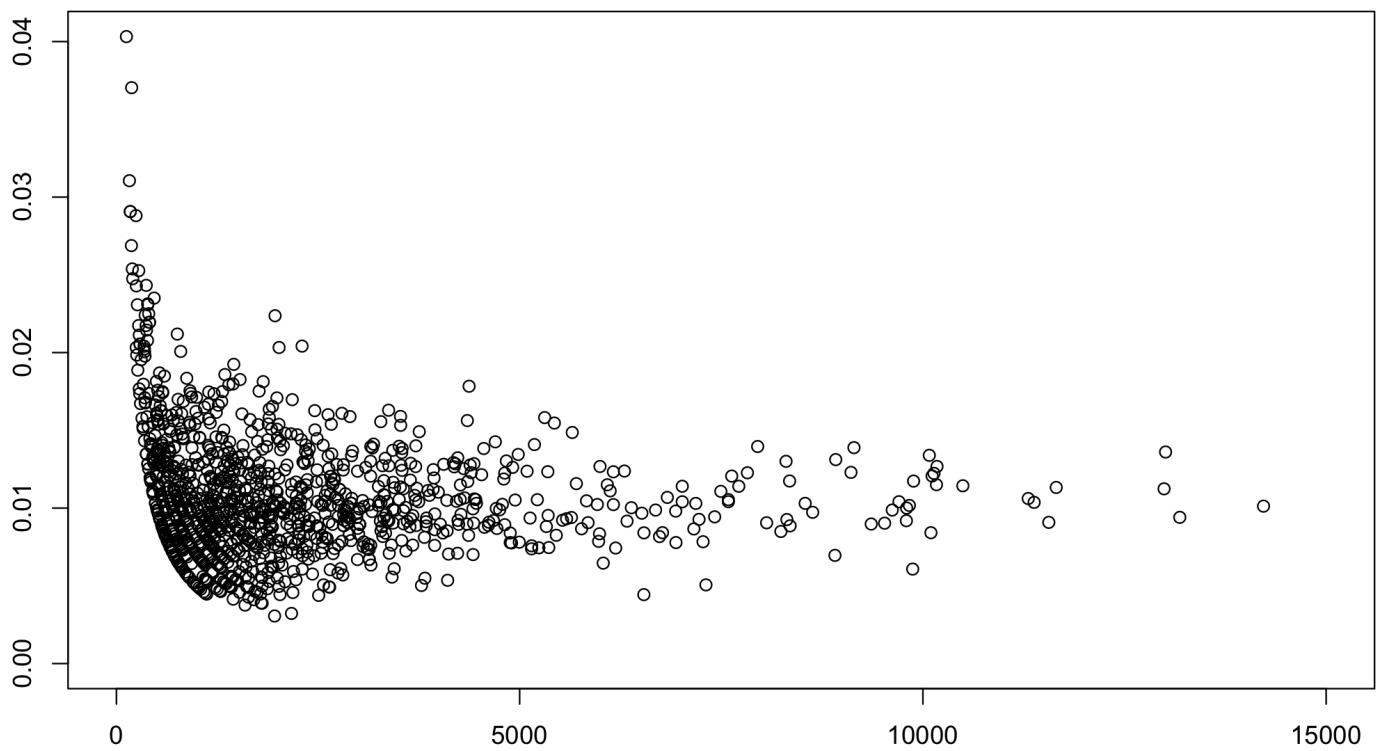
これによって、以下のようなグラフが出力される。曲線は全体の平均と、95/99%信頼区間を表している。



統計学の基本

データの可視化は簡単にできるになったが、データの解釈が簡単になったわけではない。

death by liver disease



上の例では、女性の肺疾患による死亡の比率を単純に比べた場合、表計算ソフトの並べ替えを使えば、すぐにワースト記録が分かる。肝臓に悪い風土病、食習慣がある地域なのだろうか？

A screenshot of a spreadsheet application showing a table of data and a corresponding bar chart. The table lists various locations with their names, key codes, total populations, deaths, and death ratios. The bar chart on the right shows the death ratio for each location, with values ranging from approximately 0.025 to 0.045.

	A	B	C	D	E	F	G	H	I
1	NAME	KEY_CODE	W_TOTAL	DEATH	RATIO	LEN			
2	40609 赤村	40609	124	5	0.0403226		5		
3	21382 輪之内町	21382	189	7	0.037037		5		
4	01549 訓子府町	01549	161	5	0.0310559		5		
5	01423 南幌町	01423	172	5	0.0290698		5		
6	29362 三宅町	29362	172	5	0.0290698		5		
7	01585 安平町	01585	243	7	0.0288066		5		
8	20349 青木村	20349	186	5	0.0268817		5		
9	41345 上峰町	41345	197	5	0.0253807		5		
10	47325 嘉手納町	47325	277	7	0.0252708		5		
11	40642 吉富町	40642	202	5	0.0247525		5		
12	20481 池田町	20481	370	9	0.0243243		5		
13	47314 金武町	47314	247	6	0.0242915		5		
14	11342 嵐山町	11342	468	11	0.0235043		5		
15	37403 琴平町	37403	389	9	0.0231362		5		
16	07342 鏡石町	07342	260	6	0.0230769		5		
17	40500 篠之町	40500	200	0	0.0230769		-		

試しに、ベスト市町村は以下の通り。これらの市町村には肝臓によい食習慣があるのだろうか？

The screenshot shows a spreadsheet application with the following data:

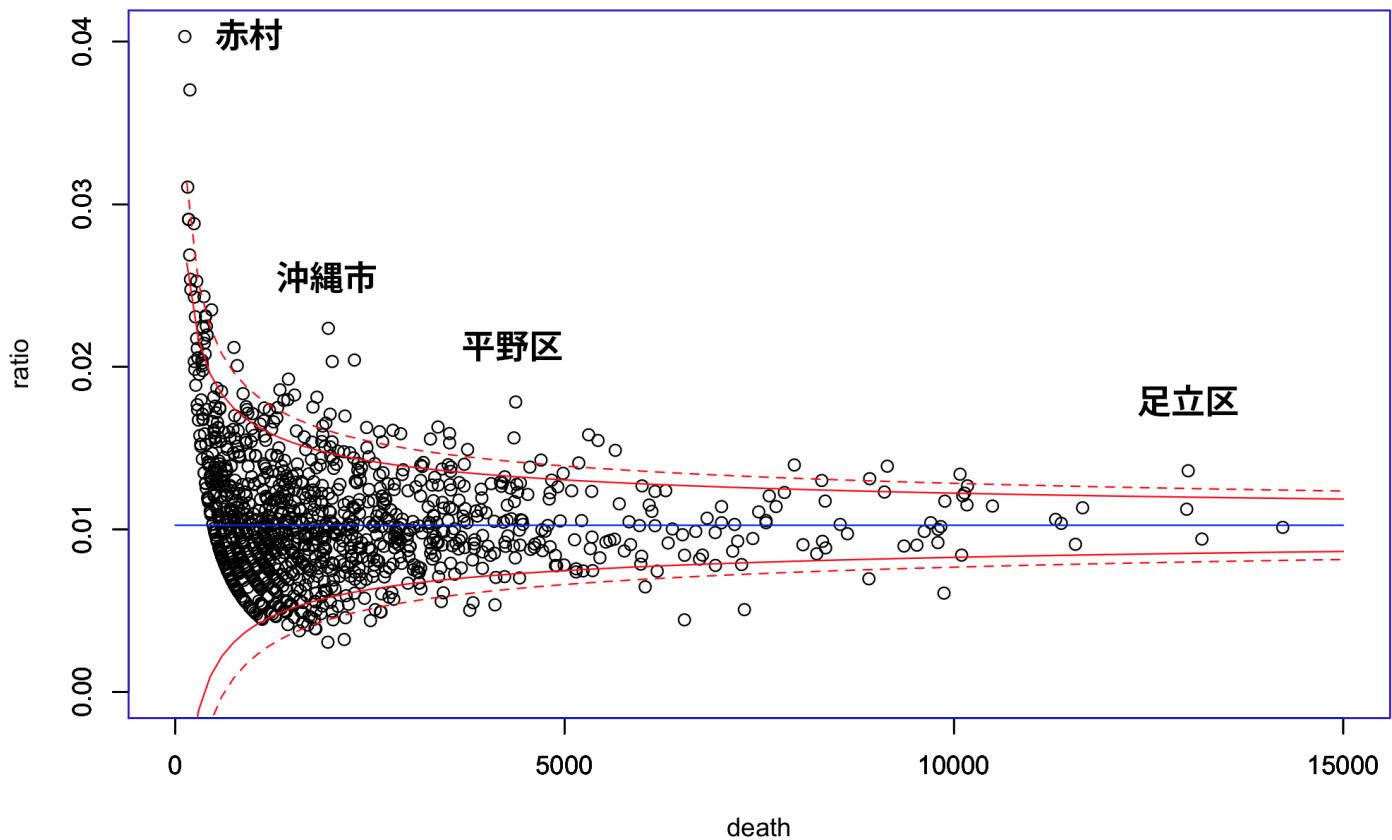
	A	B	C	D	E	F	G	H	I
1155	35204 萩市	35204	2188	10	0.0045704	5			
1156	39210 四万十市	39210	1103	5	0.0045331	5			
1157	04207 名取市	04207	1787	8	0.0044768	5			
1158	33216 浅口市	33216	1119	5	0.0044683	5			
1159	03211 釜石市	03211	2028	9	0.0044379	5			
1160	04202 石巻市	04202	6540	29	0.0044343	5			
1161	22133 西区	22133	2508	11	0.004386	5			
1162	03210 陸前高田市	03210	1638	7	0.0042735	5			
1163	04203 塩竈市	04203	1448	6	0.0041436	5			
1164	38214 西予市	38214	1703	7	0.0041104	5			
1165	16210 南砺市	16210	1796	7	0.0038976	5			
1166	33205 笠岡市	33205	1806	7	0.003876	5			
1167	24208 名張市	24208	1595	6	0.0037618	5			
1168	40207 柳川市	40207	2171	7	0.0032243	5			
1169	15105 秋葉区	15105	1961	6	0.0030597	5			
1170									
1171									
1172									
1173									
1174									

統計学の教科書によると、標本調査で分かる比率 p の分散は、「 $p(1-p)/標本数$ 」になる。つまり、標本数が多ければ多いほど、ブレが少なくなる。

例えば、6個のサイコロを振り、1が出たサイコロの比率を考える場合、1が3つある場合(50%)も時にはあるだろうことは直感的に分かる。一方、100個のサイコロを振り、1が50個ある場合(50%)はほとんど考えられない。(一生サイコロを降り続けても起きない)

つまり、この種のデータの異常性は、分散（その平方根が標準偏差）を基準にして考えなければならない。4.0%の福岡県赤村（筑豊）よりも、13000人付近にある東京都足立区(1.36%, 3.79std)、4300人付近にある大阪府平野区(1.78%, 4.98std)、1900人付近の沖縄市(2.23%, 5.3std)の方が異常である。

death by liver disease



この問題は、現場にはかなり過酷な要求である。

アメリカのように、全国学力テストの学校別データが公表されたとする。上の理由で、優秀校ベスト10は間違いなく小規模校になる。

たまたま賢い子供がいた影響がダイレクトに反映されてしまうからだ。

この時、教育担当記者は「小規模クラスの方が成績がよい（というデータの裏付けがある）」という結論に陥ってしまう危険性がある。（ワースト校も小規模校ばかりであることに気づくべきだ。そのためには、表計算ソフトを苦もなく使いこなせなければならない。また、優秀校には「たまたま賢い子が多い」とは別の要因がないという意味ではない）

一方、政党の市町村別得票率にも同じ現象が起きる。

この時、統計に半可通の政治担当記者が「この特異な得票率は人口が少ないからだ」という結論を出す可能性もある。

有権者数は小さな町村でも大きな数字なので分散はかなり小さい。だから、特異な得票率は、ほとんどの場合、地域の事情を反映している。そもそも、選挙結果は、「母集団」を想定することが適切なデータではない。

一般的に、初級レベルの統計学は、キャッチャーなネタを潰す方向に働く。数字に惑わされないと、統計学を学ぼうという意思を削がれかねない。