空間情報を用いた社会・経済分析(第2回)

統計数理研究所　村上大輔

dmuraka@ism.ac.jp

担当回（前半）

内容

- 第2回(4/21 月）：空間データの処理・地図化
- 第3回(4/28 月）：探索的空間データ解析
- 第4回(5/8, **木**)　：空間計量経済モデルと応用

↑

各回で統計ソフトウェアRを用いた実例を紹介

Rコード置き場

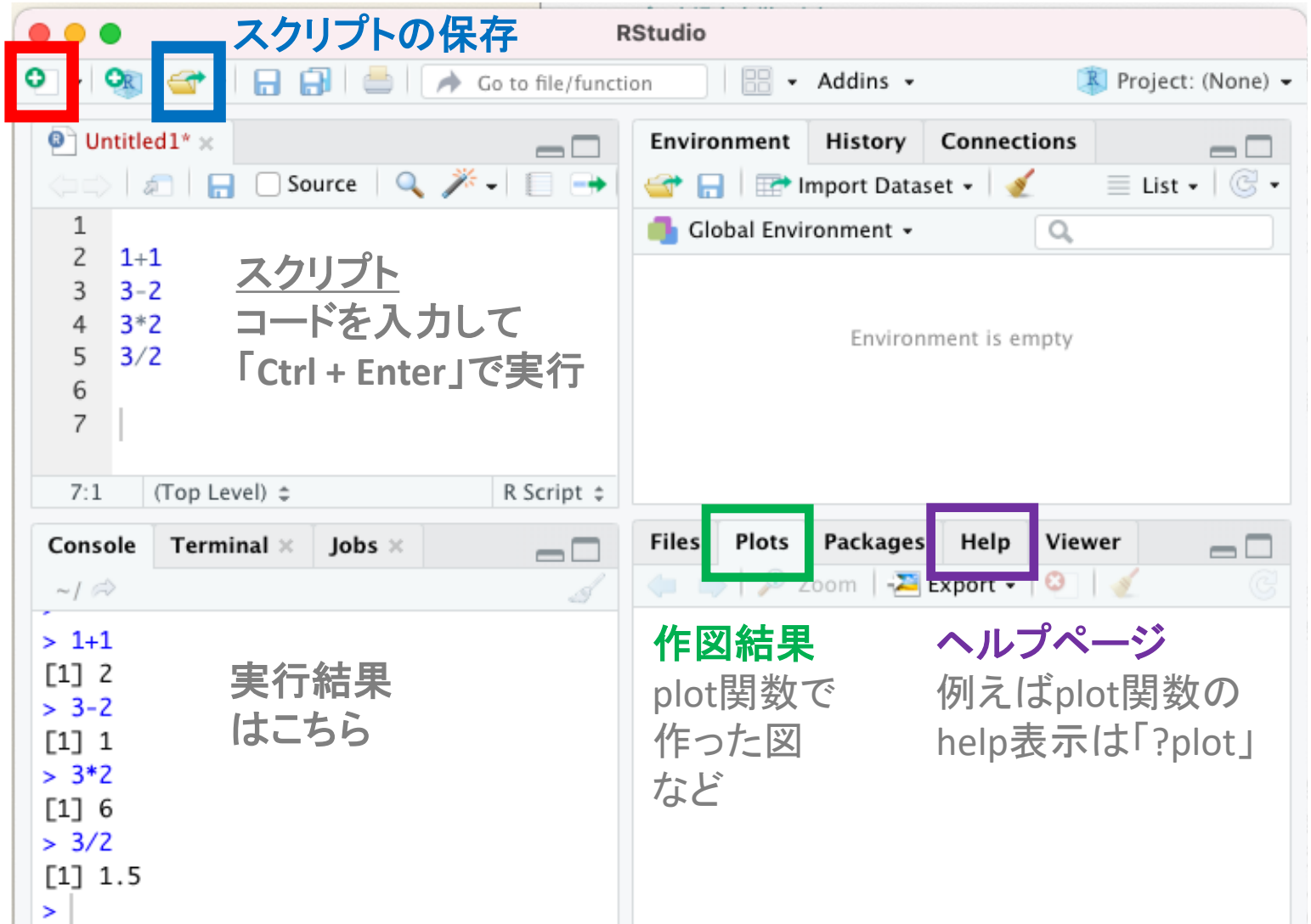
https://github.com/dmuraka/HIAS_class

- 質問等は村上(dmuraka@ism.ac.jp)までご連絡ください

RstudioでRを回します

- **R**
  - ✓統計解析に特化したフリーソフト。4/19時点で22,341個のパッケージ
  - ✓空間データの処理・視覚化のパッケージはpython等よりも充実した印象

- **Rstudio**
  - ✓Rを使う環境

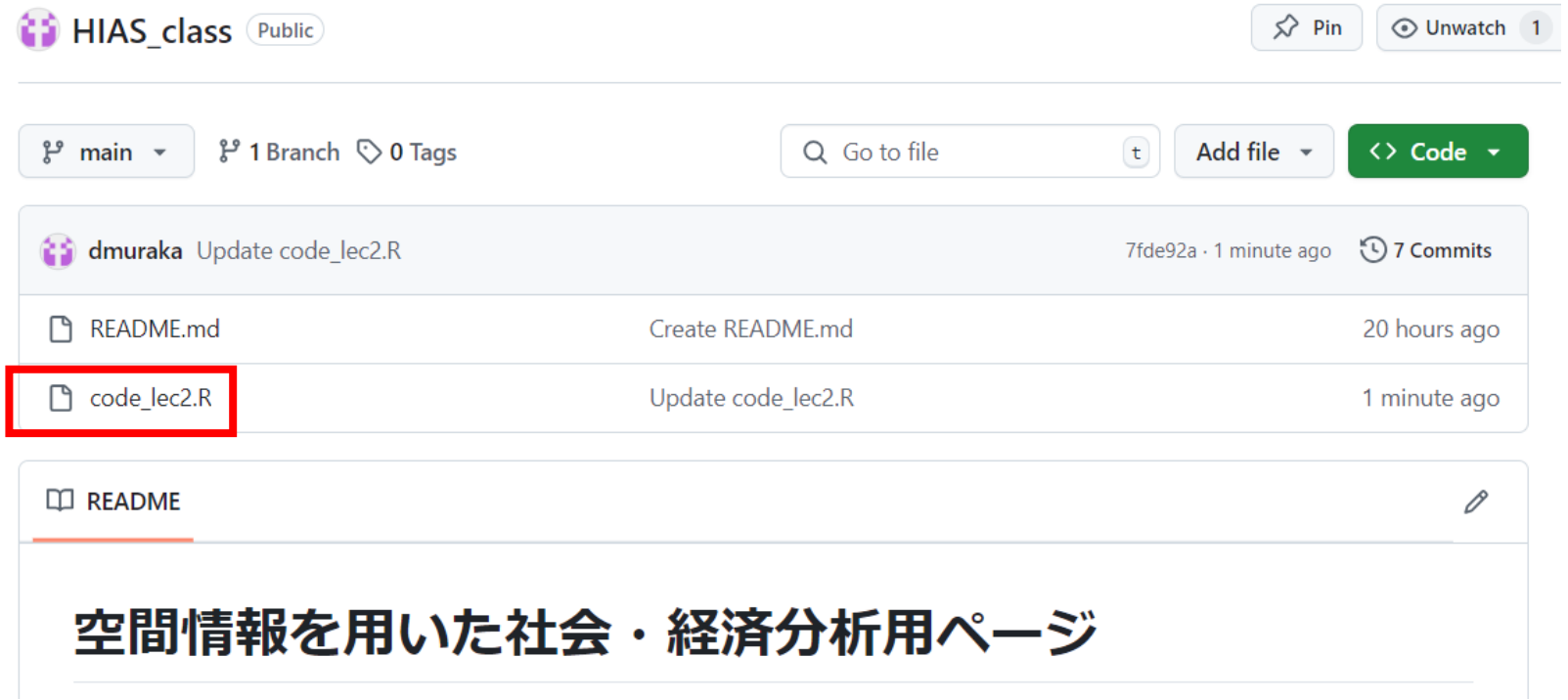


第2回: Rを用いた空間データの処理・地図化

コードは以下ページ。こちらも参照しながらお聞きください

https://github.com/dmuraka/HIAS_class/

ここをクリック→

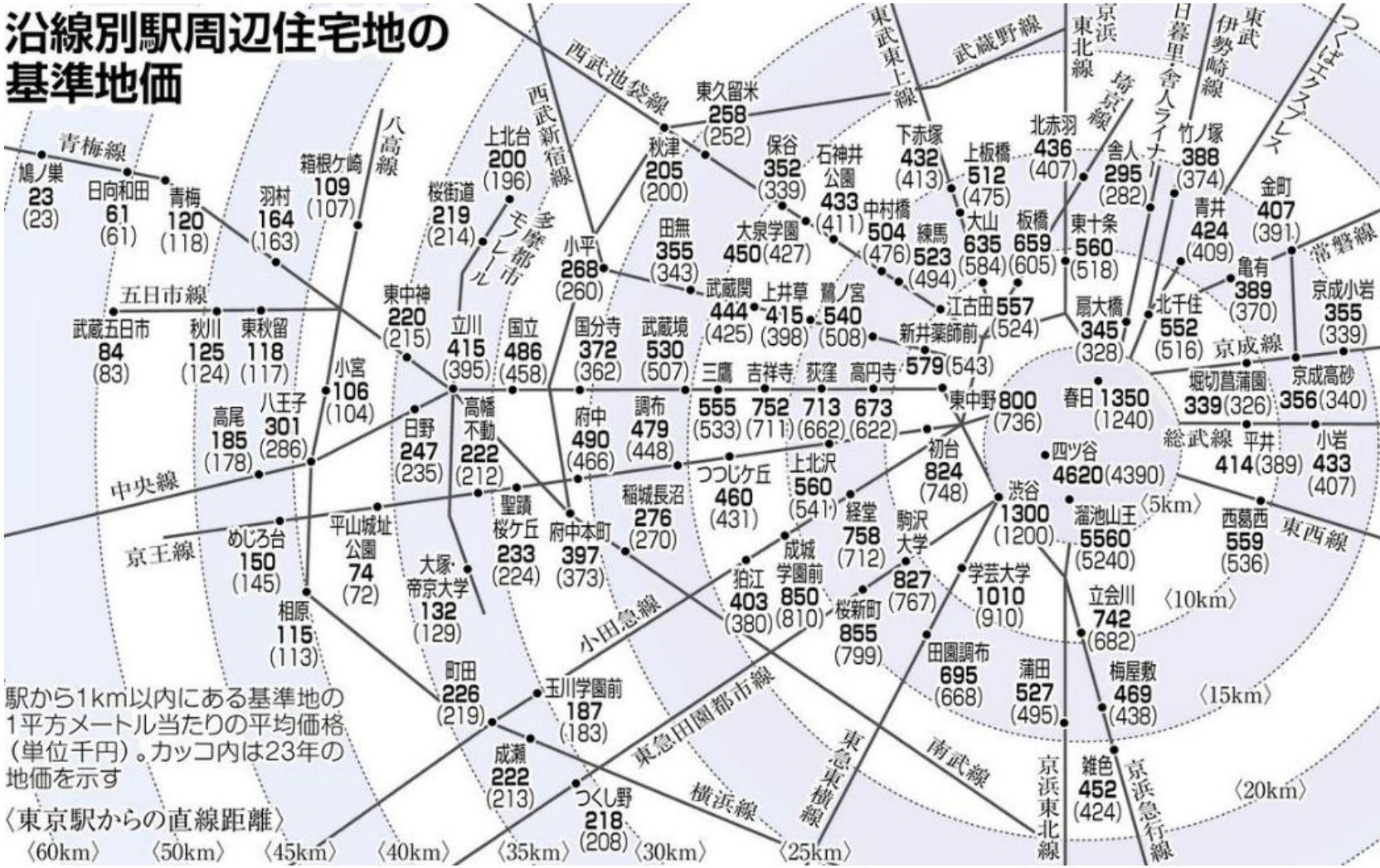


今日の題材：住宅地価分析

- **地価≒土地の魅力**

住宅地価の上位5位 (2024)

	順位	所在地	価格	変動率
区部	1	港区赤坂1-14-11	5560	6.1
	2	千代田区六番町6-1外	4620	5.2
	3	千代田区三番町9-4	3580	10.2
	4	千代田区麹町2-10-4外	2990	9.9
	5	千代田区二番町12-10	2650	10.0
多摩地域	1	武蔵野市吉祥寺本町4-21-6	810	6.6
	2	武蔵野市吉祥寺南町3-31-11	756	6.0
	3	三鷹市井の頭4-16-4	694	5.0
	4	武蔵野市東町4-10-9	590	5.2
	5	三鷹市下連雀4-4-19	555	4.1



出典: 東京新聞 https://www.tokyo-np.co.jp/article_photo/list?article_id=354790&pid=1608823

住宅地価分析のイメージ

要因分析

- 国立の住宅地価が高い理由は？
- 駅？大学？緑？ブランド力？

→幅広いデータを収集・整備
→回帰分析

予測

- 地価がわかるのは調査が行われた地点だけ
- 調査が行われていない地点の地価を予測するには？

→回帰分析の結果を元に予測

本日の地価分析の流れ

(1)データ収集

- 住宅地価, 土地利用, 鉄道駅,…

(2)データ整備

- 各地価調査地点の土地利用、最寄駅距離の計算,…

(3)分析

- 回帰分析, 予測,…

(4)地図化

- 地価の予測値の地図化

→ 典型的なワークフロー
→ 以上の一連の流れを
Rで行う方法を紹介

国土数値情報ダウンロードサイト(NLNI; https://nlftp.mlit.go.jp/ksj/)

地形、土地利用、公共施設などの国土の基礎的な情報を無償提供

余談

統計地理情報システムも社会経済分野でよく使われる

- https://www.e-stat.go.jp/gis
- 人口, 事業所数等の公的統計

データ提供サイトは分野毎に異なる

位置座標よ観測値を自前で収集する場合

- 例：生物、植物の分布データ

地価公示データのダウンロード(今回は不要)

- 地域・年を選択　→　アンケートに回答(スキップ可)　→　zipファイルをDL

↓

同じ地価データが2つの形式で格納されている

GeoJSON形式

L01-22_13.geojson　GEOJSON ファイル

Shapefile形式

L01-22_13.dbf	DBF ファイル	← dbf: 地点毎の属性データ(例:地価)
L01-22_13.prj	PRJ ファイル	← prj: 投影法(後述)の定義
L01-22_13.shp	SHP ファイル	← shp: ジオメトリ(形状)の情報
L01-22_13.shx	SHX ファイル	← shx:ジオメトリ(shp)と属性データ(dbf)の対応関係

空間データの形式

以下のいずれかの場合が多い(例: NLNI)

GeoJSON (.geojson)
- 単一ファイルからなる形式。中身が可読。ブラウザやアプリ連携に強い。

Shapefile (.shp)
- 複数ファイルからなる伝統的・業務用な形式。高速に扱える。

※複数データを格納可能な形式
GeoDatabase
GeoPackage

RではGeoJSON or Shapefile形式のデータを**sf**形式に変換

- sfパッケージのst_read()関数で、GeoJSONまたはShapefileデータを読み込んでsf形式で格納

```
> install.packages("sf")                    # sfパッケージのインストール(初回のみ; Web ->PC)
> library(sf)                               # sfパッケージの読込(PC -> R)
> dprice   <-st_read("landprice_R7.shp")    # 地価公示データの読込(shapefile → sf)
```

地価公示データ(sf形式)の中身の確認

```
> dprice[1:4, ]      # 最初4行だけ表示
Simple feature collection with 4 features and 146 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 139.7329 ymin: 35.6812 xmax: 139.7465 ymax: 35.69863
Geodetic CRS:  JGD2011
  L01_001 L01_002 L01_003 L01_004 L01_005 L01_006 L01_007 L01_008
1   13101     000     001   13101     000     001    2025 3960000
2   13101     000     002   13101     000     002    2025 2530000
3   13101     000     003   13101     000     003    2025 4830000
4   13101     000     004   13101     000     004    2025 1970000
                   L01_145         L01_146                    geometry
1  10000000000000 10000000000000 POINT (139.7448 35.69014)
2  10000000000000 10000000000000  POINT (139.7375 35.6812)
3  10000000000001 10000000000000 POINT (139.7329 35.68814)
4  10000000000000 10000000000000 POINT (139.7465 35.69863)
```

ジオメトリのタイプ→

座標参照系→

地点毎の属性

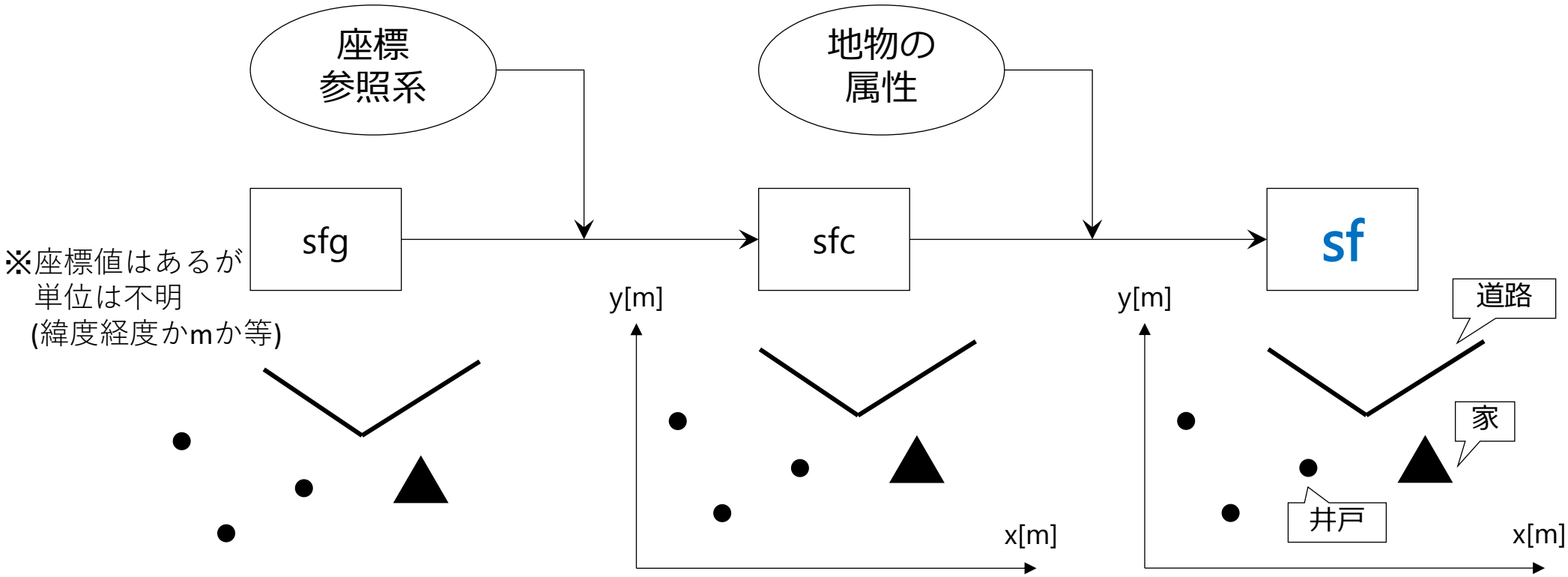
↑ジオメトリ

変数(L01_001, L01_002,…)の定義については国土数値情報ダウンロードサイト参照
https://nlftp.mlit.go.jp/ksj/gml/datalist/KsjTmplt-L01-2025.html

sf (simple feature)形式

・ジオメトリ・座標参照系・属性からなるR用の内部形式

- ✓ジオメトリ　：point (点), polygon (面), linestring (線),…
- ✓座標参照系　：位置座標の定義（後述）
- ✓属性　：地物毎の属性情報



本日使用するデータ

- **目的**
  - ✓回帰分析で最寄駅距離や土地利用が住宅地価に及ぼす影響を分析

- **国土数値情報ダウンロードサイトから以下をDL**
  - ✓地価公示データ(令和7)
  - ✓鉄道時系列(令和3)のうち、2021年に影響している鉄道駅
  - ✓土地利用３次メッシュ(令和3)

- **今回は以下のコマンドで各データをダウンロードください：**

```
dprice   <-st_read(dsn="https://www.dropbox.com/scl/fi/hllfqvyqna7eoj2y9rwem/landprice_R7.geojson?rlkey=c998opyiwbift11vmuls1f7tk&dl=1")
dstation<-st_read(dsn="https://www.dropbox.com/scl/fi/uqg0hzpqs6mya1hcgggcw/station_R3.geojson?rlkey=p0oixglii01xcgwuww3mcbw6x&dl=1")
dland    <-st_read(dsn="https://www.dropbox.com/scl/fi/eb9cbb9tt2g06wydqilnl/landuse_R3.geojson?rlkey=r4bh5aoemjf3lj5ur8df6crs5&dl=1")
```

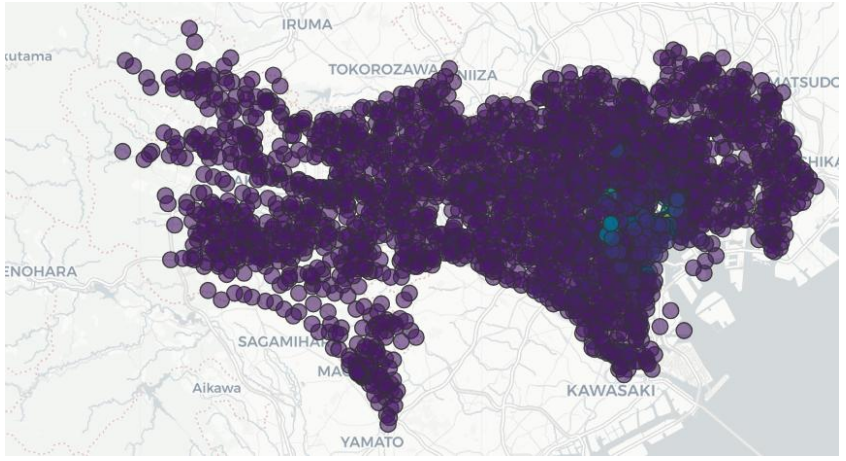
とりあえずプロット

mapviewパッケージが便利

- 移動、拡大などいろいろできる
- 細かな設定については後ほど

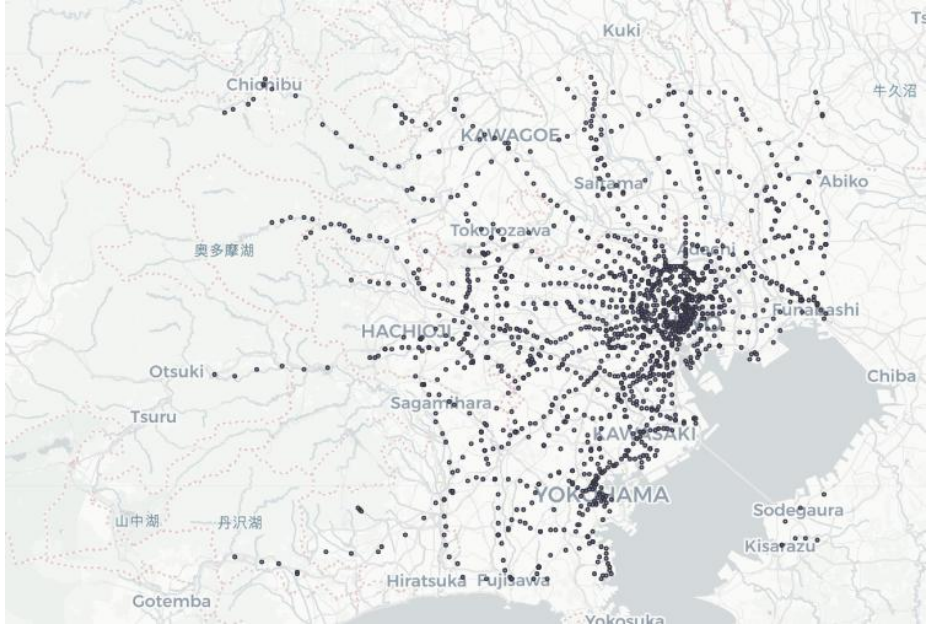
地価公示データ (point)

mapview(dprice[,"price"])



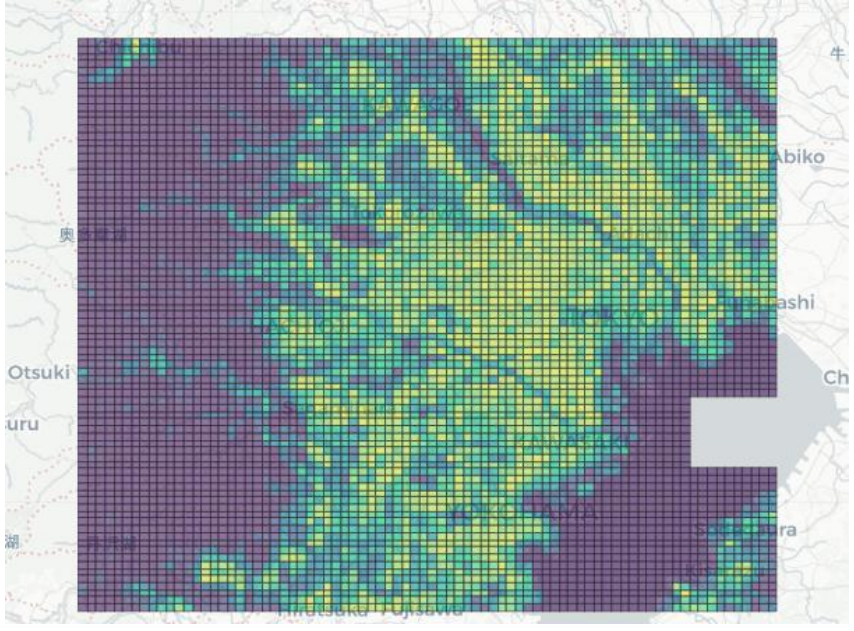
鉄道駅 (point)

mapview(dstation,cex=0.5)



土地利用3次メッシュ (polygon)

mapview(dland[,“building”]) #建物用地



位置座標について

```
> dprice[1:4, ]      # 最初4行だけ表示
Simple feature collection with 4 features and 146 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 139.7329 ymin: 35.6812 xmax: 139.7465 ymax: 35.69863
Geodetic CRS:  JGD2011

  L01_001 L01_002 L01_003 L01_004 L01_005 L01_006 L01_007 L01_008
1   13101     000     001   13101     000     001    2025 3960000
2   13101     000     002   13101     000     002    2025 2530000
3   13101     000     003   13101     000     003    2025 4830000
4   13101     000     004   13101     000     004    2025 1970000

                 L01_145         L01_146                    geometry
10000000000000 10000000000000 POINT (139.7448 35.69014)
10000000000000 10000000000000  POINT (139.7375 35.6812)
10000000000001 10000000000000 POINT (139.7329 35.68814)
10000000000000 10000000000000 POINT (139.7465 35.69863)
```

座標参照系→

. . .

. . .

値はどのように評価されたもの？
単位は？(度?メートル?)

→座標参照系

位置座標について

- 地球上の位置を2次元座標で表す方法について、これから説明

EPSGコード: CRS毎のID
- 投影法の定義：EPSGの指定
- 投影変換　　：EPSGの変更

座標参照系（CRS）：測地系と座標系のペア

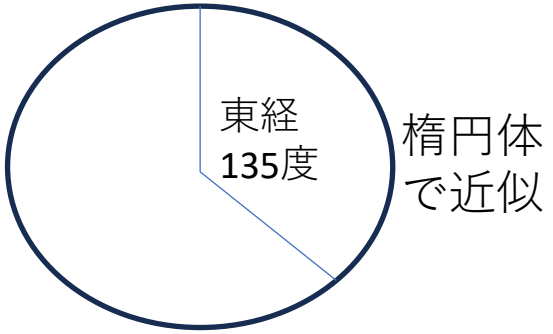
測地系
球面上の緯度経度を定義
- 世界測地系
  - WGS84, JGD2011,...
- 局所測地系
  - 日本測地系,…
  ⋮

座標系
球面を**投影**することで与えられる
2次元平面上の座標を定義
- 地理座標系
- 投影座標系
  - ✓ UTM座標系
  - ✓ 平面直角座標系
  ⋮

投影法
投影のための手法
- 円筒図法
- 平面図法
⋮

測地系 (geodetic system)

✓ 地球を楕円体で近似。**地球(楕円体)上の**位置を緯度経度で表すシステム



世界測地系：世界共通

✓World Geodetic System 1984 (WGS84): 米国が構築。今日の GPS の位置基準

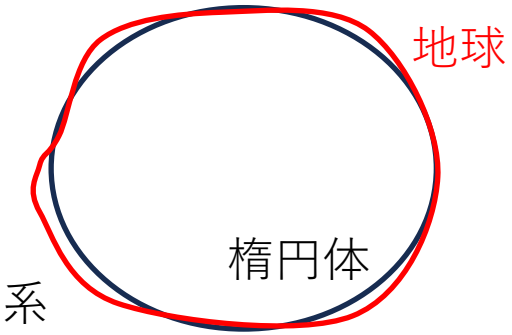
局所測地系：地域独自の局所測地系。日本は以下

✓旧日本測地系　　：2002/3まで。世界測地系とは数百mのずれ

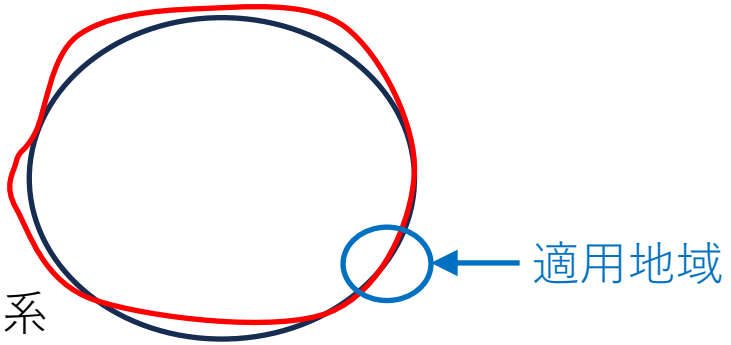
✓日本測地系 (JGD2000)　：2011/10まで。 WGS84と互換するよう再設計

✓日本測地系 (JGD2011)　：現行。東日本大震災の影響を考慮して補正。WGS84とのずれは数十cm

今日では、WGS84との違いを無視しても影響は小さい場合が多い

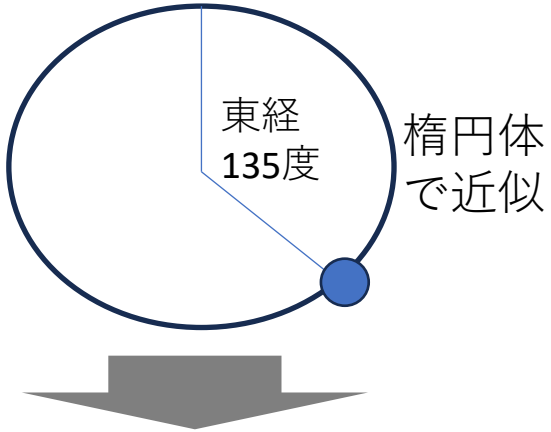


世界測地系



局所測地系

座標系 (coordinate system)

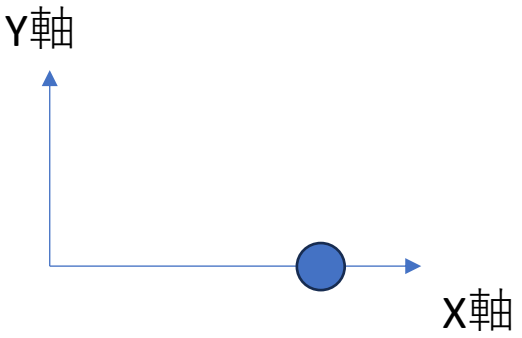


✓ 地球(楕円体)上の緯度経度を、**<u>平面上の2次元座標で表す</u>**するシステム

地理座標系

✓ (X, Y) = (経度, 緯度)

✓ 問題：緯度によって見た目の位置関係が実際と異なる(例:グリーンランド)



局所の精度を高める

投影座標系(座標をm単位等で与えられる)

✓ **UTM 座標系**: 地球を経度 6 度毎の縦長のゾーンに分割。ゾーン毎に(X, Y)=(赤道, 中央経線)

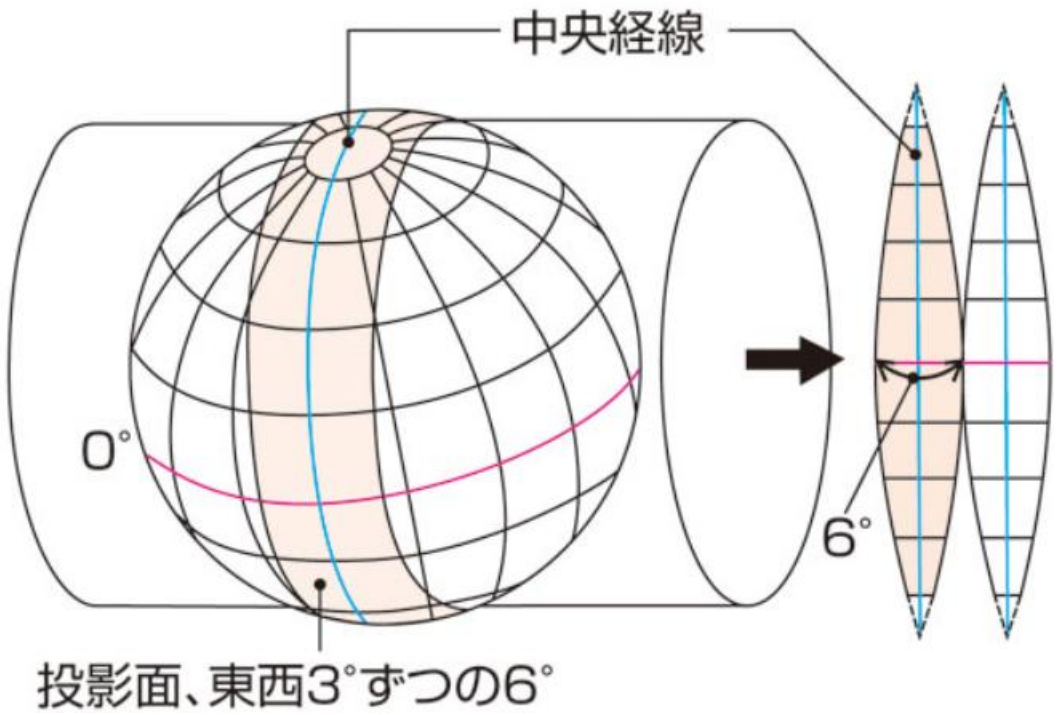
- 日本はゾーン 51～56 （例：東京はゾーン 54、大阪はゾーン 53 ）

✓ **平面直角座標系**: 原点座標の周辺に対する座標。(X, Y)=(原点を通る緯線(子午線)、原点を通る経線)

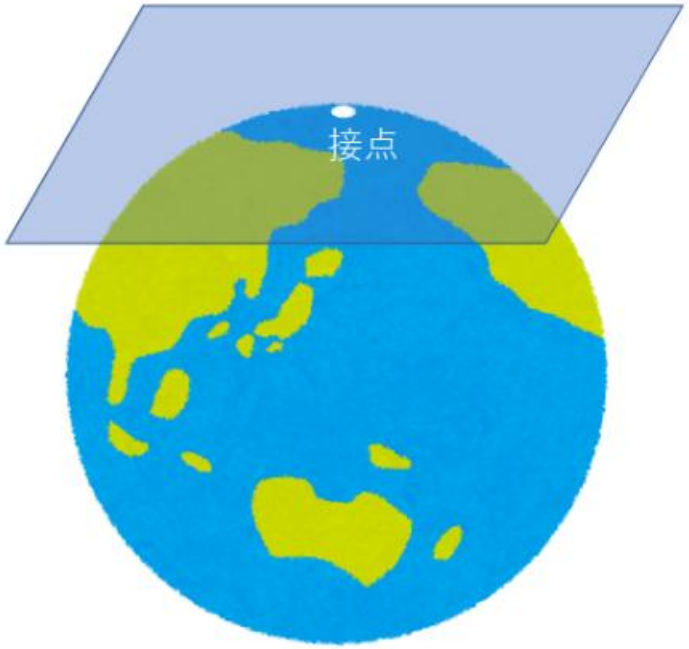
- 日本は 19 の原点座標（例：東京は第 9 系、大阪は第 6 系）

投影座標系における2次元平面への投影法

UTM 座標系：円筒図法(メルカトル図法)



平面直角座標系：平面図法



出典: GIS基礎：座標系を理解する https://sk-lb.net/gis-crs/
コトバンク ユニバーサル横メルカトル図法

日本で用いられる投影座標系

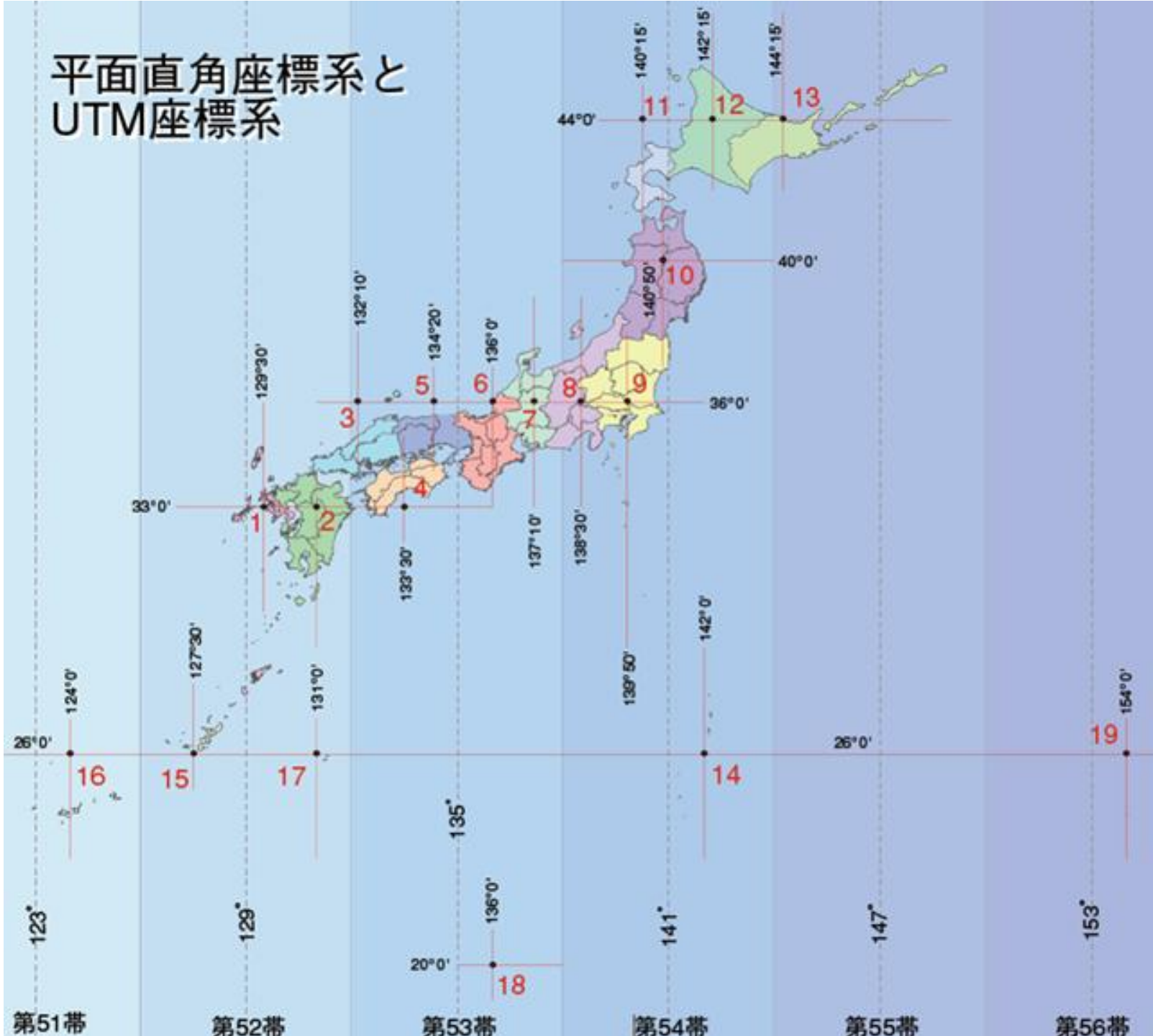
UTM座標系

- 第51-56帯

平面直角座標系

- **1 – 19**を原点とする第1~19系

出典
https://www.esrij.com/gis-guide/coordinate-and-spatial/coordinate-system-japan/



座標参照系（CRS: Coordinates Reference System）

✓**測地系と座標系のペア。座標参照系が与えられると2次元座標が決まる**

EPSG コード: 座標参照系毎のID

✓NLNLのデータにはEPSGが予め付与されている。自力で収集したデータなどの場合、ESGGを自分で付与する必要。緯度経度であれば4326で基本的にはOK

表 1.1: 我が国でよく使う EPSG コードの例

EPSG	測地系	座標系	単位	備考
6669 ~6687	日本測地系 2011 (JGD2011)	平面直角座標系	m, km, 緯度経度	日本の地域毎 (1～19 系) の座標系。例えば東京は第 9 系（EPSG:6677 大阪は第 6 系 (EPSG:6674)
6688 ~6692	日本測地系 2011 (JGD2011)	UTM 座標系	m, km, 緯度経度	地球表面を経度 6 度毎で 60 分割した領域毎の座標系。例えば東京を含む東経 138-144 は EPSG:6691、大阪を含む東経 132-138 は EPSG:6690
4326	世界測地系 (WGS84)	地理座標系	緯度経度	地球全体の位置を経度と緯度で表現

座標参照系(CRS)を理解する必要性

- 座標参照系(EPSGコード)が定義されていないデータも多い
  - ✓座標値の単位が不明なため、地図上でずれた位置に表示されたり、距離・面積が正しく評価されなかったり、重ね合わせができなかったりする
    - 表形式のデータの場合（CSV等）
    - 自分で収集したデータの場合（GPS等で位置座標を収集）
  - ✓データを重ね合わせなどをする際は、座標参照系の統一が必要

- 定義されていない場合　→　座標参照系(EPSGコード)を定義
  - 緯度経度の場合はEPSG=4326（WGS84）でOK

- 座標参照系を統一/変換したい場合　→　変換
  - 一方のEPSGを他方に合わせる

座標参照系の定義の例

```
> dprice[1:4, ]      # CSVをRに読み込んだものとする
         X        Y   price
1 139.7448 35.69014 3960000
2 139.7375 35.68120 2530000
3 139.7329 35.68814 4830000
4 139.7465 35.69863 1970000
```

```
> dprice2c <- st_as_sf(dprice2b, coords=c("X","Y"))     # sf形式に変換
> dprice2c[1:4,]
Simple feature collection with 4 features and 1 field
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 139.7329 ymin: 35.6812 xmax: 139.7465 ymax: 35.69863
CRS:           NA
    price                    geometry
1 3960000 POINT (139.7448 35.69014)
2 2530000  POINT (139.7375 35.6812)
```

座標参照系が定義されていない →

```
> st_crs( dprice2c ) <- 4326  ←世界測地系WGS84 (EPSG: 4326)で定義
> dprice2c[1:4,]
Simple feature collection with 4 features and 1 field
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 139.7329 ymin: 35.6812 xmax: 139.7465 ymax: 35.69863
Geodetic CRS:  WGS 84
    price                    geometry
1 3960000 POINT (139.7448 35.69014)
2 2530000  POINT (139.7375 35.6812)
3 4830000 POINT (139.7329 35.68814)
4 1970000 POINT (139.7465 35.69863)
```

座標参照系が世界測地系(WGS84)になった →

```
> st_crs( dprice2c ) <- 6677  ←平面直角座標系第9系(EPSG: 6677)への変換
```

本日の地価分析の流れ

(1)データ収集
- 住宅地価, 土地利用, 鉄道駅,…

(2)データ整備
- 各地価調査地点の土地利用、最寄駅距離の計算,…

(3)分析
- 回帰分析, 予測,…

(4)地図化
- 地価の予測値の地図化

→ 典型的なワークフロー
→ 以上の一連の流れを
Rで行う方法を紹介

よくやる処理

列名の変更(日本語はバグりやすい)

```
> names(dprice)[ names(dprice) ==“L01_008” ]<-“price”   # 地価(L01_008)の列名        → price
> names(dprice)[ names(dprice) ==“L01_028” ]<-“status”  # 利用現況(L01_028)の列名 → status

> dprice[1:10,c(“price”,“status”)]          # 最初10行の確認
```

```
Simple feature collection with 10 features and 2 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 139.7329 ymin: 35.68065 xmax: 139.7639 ymax: 35.69863
Geodetic CRS:  JGD2011
      price      status                    geometry
1   3960000        住宅 POINT (139.7448 35.69014)
2   2530000        住宅  POINT (139.7375 35.6812)
3   4830000        住宅 POINT (139.7329 35.68814)
4   1970000        住宅 POINT (139.7465 35.69863)
5   3680000        住宅 POINT (139.7462 35.69608)
6   2270000        住宅 POINT (139.7408 35.68065)
7   3740000        住宅 POINT (139.7408 35.68827)
8  14600000      事務所 POINT (139.7597 35.69063)
9  37100000 店舗,事務所 POINT (139.7639 35.68109)
10  1760000 店舗,事務所  POINT (139.7628 35.6917)
```

↑
住宅以外も含まれているので除きたい

よくやる処理

文字列の処理 + 行・列の選択

```
> table(dprice$status)          # statusの値毎のカウント
  住宅       住宅, 事業所      店舗,....
  1563       84                71 …
> is_residual     <- str_detect(dprice$status, “住宅”)    # “住宅”を含む場合はTRUE, さもなくばFALSE
> is_residual2    <- is.element(dprice$status, “住宅”)    # “住宅”に完全一致はTRUE, さもなくばFALSE
> cbind(is_residual,is_residual2,dprice$status)[1:10,]    # statusと並べて表示（最初10行だけ)
```

```
      is_residual is_residual2
 [1,] "TRUE"      "TRUE"       "住宅"
 [2,] "TRUE"      "TRUE"       "住宅"
 [3,] "TRUE"      "TRUE"       "住宅"
 [4,] "TRUE"      "TRUE"       "住宅"
 [5,] "TRUE"      "TRUE"       "住宅"
 [6,] "TRUE"      "TRUE"       "住宅"
 [7,] "TRUE"      "TRUE"       "住宅"
 [8,] "FALSE"     "FALSE"      "事務所"
 [9,] "FALSE"     "FALSE"      "店舗,事務所"
[10,] "FALSE"     "FALSE"      "店舗,事務所"
```

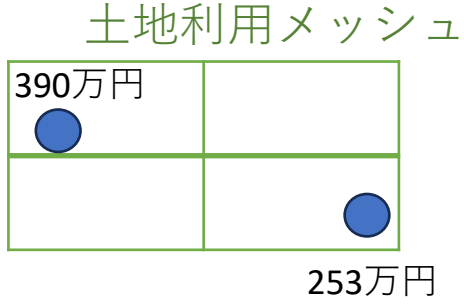
←非住宅系

```
> dprice2    <- dprice[is_residual, c(“price”,“status”) ]    # 今回はis_residualがTRUEの行のみ使用
> dprice2[1:10,]
```

```
    price          status                    geometry
1  3960000            住宅 POINT (139.7448 35.69014)
2  2530000            住宅  POINT (139.7375 35.6812)
3  4830000            住宅 POINT (139.7329 35.68814)
4  1970000            住宅 POINT (139.7465 35.69863)
5  3680000            住宅 POINT (139.7462 35.69608)
6  2270000            住宅 POINT (139.7408 35.68065)
7  3740000            住宅 POINT (139.7408 35.68827)
11 3790000 住宅,店舗,事務所 POINT (139.7749 35.69587)
12 3200000 住宅,店舗,事務所  POINT (139.771 35.70278)
13 1580000      住宅,事務所  POINT (139.769 35.70212)
```

↑
住宅系のみが残った

空間結合



地価データに、地点を含む土地利用メッシュの属性値を結合

```
> dprice3 <- st_join(dprice2, dland, join = st_within)
```

dprice2

	price	status
1	3960000	住宅
2	2530000	住宅
3	4830000	住宅
4	1970000	住宅
5	3680000	住宅

1km²メッシュに占める森林面積(m²)
↓

dland

	mesh	rice	agri	forest	wild	building	road	rail	other	river	beach	ocean	golf	out
1	53395000	0	0	1045315	0	0	0	0	0	0	0	0	0	0
2	53395001	0	0	1045305	0	0	0	0	0	0	0	0	0	0
3	53395002	0	0	1045294	0	0	0	0	0	0	0	0	0	0
4	53395003	0	0	1045284	0	0	0	0	0	0	0	0	0	0
5	53395004	0	0	1045274	0	0	0	0	0	0	0	0	0	0

各地価調査地点を含むメッシュの属性データを結合

dprice3

	price	status	mesh	rice	agri	forest	wild	building	road	rail	other	river	beach	ocean	golf	out
1	3960000	住宅	53394529	0	0	209134	0	575119	62740	0	62740	135937	0	0	0	0
2	2530000	住宅	53394519	0	0	94120	0	564720	188240	0	83662	115036	0	0	0	0
3	4830000	住宅	53394528	0	0	31370	0	784257	125481	20914	62741	20914	0	0	0	0
4	1970000	住宅	53394539	0	0	62734	0	690072	62734	52278	73189	104556	0	0	0	0
5	3680000	住宅	53394539	0	0	62734	0	690072	62734	52278	73189	104556	0	0	0	0

距離の計算 (st_distance関数)

- デフォルトではメートル単位の距離を評価

地価調査地点(dprice2)から最寄駅(dstation)

```
> dmat                <- st_distance(dprice2, dstation)          # 距離行列 (地価調査地点×駅)
> dprice3$st_dist <- apply(dmat, 1, min)                         # 最寄駅距離[m] → dprice3のst_dist列
```

地価調査地点(dprice2)から東京駅

地点が路線毎。簡単のため1地点目を選択
↓

```
> dstation_tokyo  <- dstation[dstation$N05_011=="東京",][1, ]# 東京駅
> dprice3$tk_dist <- st_distance(dprice2, dstation_tokyo)      # 東京駅距離[m] → dprice3のtk_dist列
```

土地利用3次メッシュ(dland)の重心から東京駅

```
> dland$tk_dist   <- st_distance(st_centroid(dland), dstation_tokyo) # 東京駅距離[m] → dlandのtk_dist列
```

↑
ポリゴンの重心

完成した
住宅地公示地価データ

列名	説明
price	住宅地公示地価(円/m²)
rice, agri, …, out	1km²メッシュ内の土地利用面積(m²)
st_dist	最寄駅距離(m)
tk_dist	東京駅までの距離(m)

```
> price[1:10, ]
Simple feature collection with 2129 features and 18 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 139.1364 ymin: 27.0949 xmax: 142.2034 ymax: 35.83208
Geodetic CRS:  JGD2011
First 10 features:
     price           status     mesh rice agri forest wild building   road   rail
1  3960000             住宅 53394529    0    0 209134    0   575119  62740      0
2  2530000             住宅 53394519    0    0  94120    0   564720 188240      0
3  4830000             住宅 53394528    0    0  31370    0   784257 125481  20914
4  1970000             住宅 53394539    0    0  62734    0   690072  62734  52278
5  3680000             住宅 53394539    0    0  62734    0   690072  62734  52278
6  2270000             住宅 53394519    0    0  94120    0   564720 188240      0
7  3740000             住宅 53394529    0    0 209134    0   575119  62740      0
11 3790000 住宅,店舗,事務所 53394631    0    0      0    0   763251 135922 125466
12 3200000 住宅,店舗,事務所 53394641    0    0      0    0   773627 188180  62727
13 1580000       住宅,事務所 53394641    0    0      0    0   773627 188180  62727
   other  river beach ocean golf out                   geometry   st_dist      tk_dist
1  62740 135937     0     0    0   0 POINT (139.7448 35.69014) 599.62116 2331.644 [m]
2  83662 115036     0     0    0   0  POINT (139.7375 35.6812) 333.27568 2762.764 [m]
3  62741  20914     0     0    0   0 POINT (139.7329 35.68814) 348.31769 3277.265 [m]
4  73189 104556     0     0    0   0 POINT (139.7465 35.69863) 303.54010 2761.092 [m]
5  73189 104556     0     0    0   0 POINT (139.7462 35.69608) 453.15420 2589.836 [m]
6  83662 115036     0     0    0   0 POINT (139.7408 35.68065) 285.30953 2468.656 [m]
7  62740 135937     0     0    0   0 POINT (139.7408 35.68827) 332.41378 2594.560 [m]
11     0  20911     0     0    0   0 POINT (139.7749 35.69587)  42.66372 1752.159 [m]
12 10454  10454     0     0    0   0  POINT (139.771 35.70278)  72.51797 2424.979 [m]
13 10454  10454     0     0    0   0  POINT (139.769 35.70212) 260.74456 2339.240 [m]
```

本日の地価分析の流れ

(1)データ収集
- 住宅地価, 土地利用, 鉄道駅,...

(2)データ整備
- 各地価調査地点の土地利用、最寄駅距離の計算,...

(3)分析
- 回帰分析, 予測,...

(4)地図化
- 地価の予測値の地図化

→ 典型的なワークフロー
→ 以上の一連の流れを
Rで行う方法を紹介

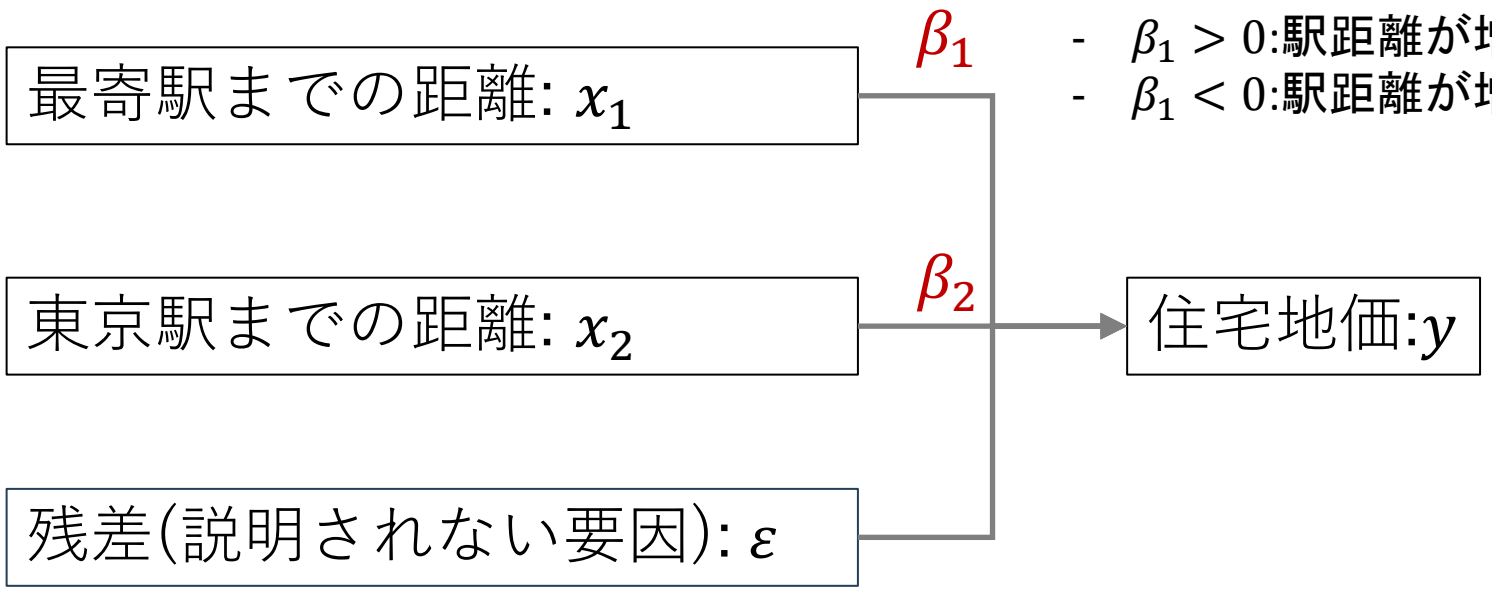
線形回帰モデル(今日はさわりだけ)

説明変数($x_1, \ldots, x_K$)が被説明変数(y; 住宅地価)に及ぼす影響の強さを推定

$$y = \beta_0 + \sum_{k=1}^{K} x_k \beta_k + \varepsilon$$

説明変数: x_k(例:最寄駅距離、東京駅までの距離)
回帰係数: β_k(k番目の説明変数からの影響の強さを表す)

住宅地価の場合



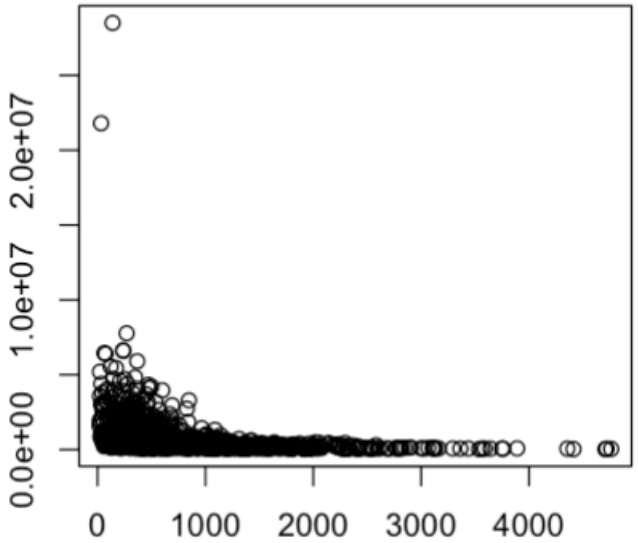
駅距離が1単位増えたら
地価はいくつ増えるか
- $\beta_1 > 0$:駅距離が増えるほど高騰
- $\beta_1 < 0$:駅距離が増えるほど低下

被説明変数と説明変数の関係の確認

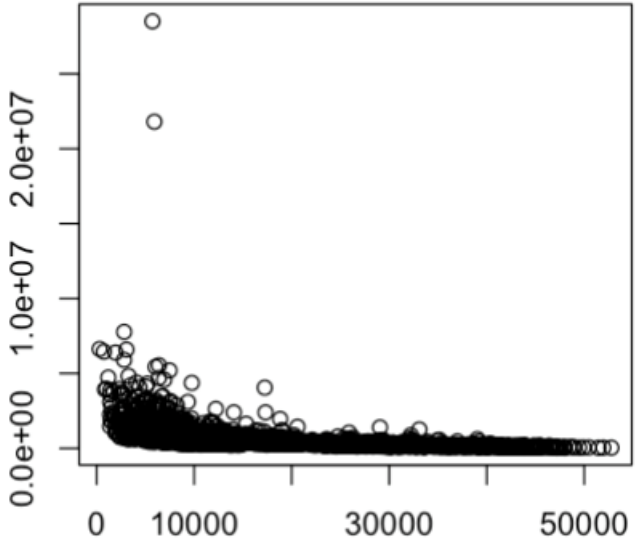
- 片対数: yだけ対数
- 両対数: yもxも対数

→st_distとtk_distに関しては、
今回は両対数が良さそう

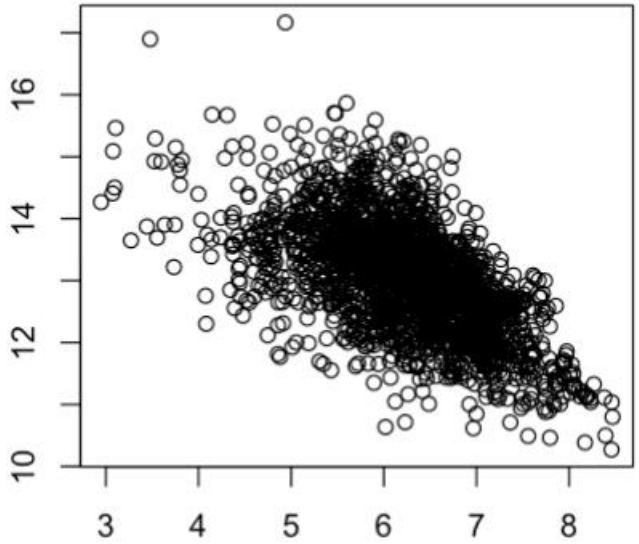
> plot(st_dist, price)



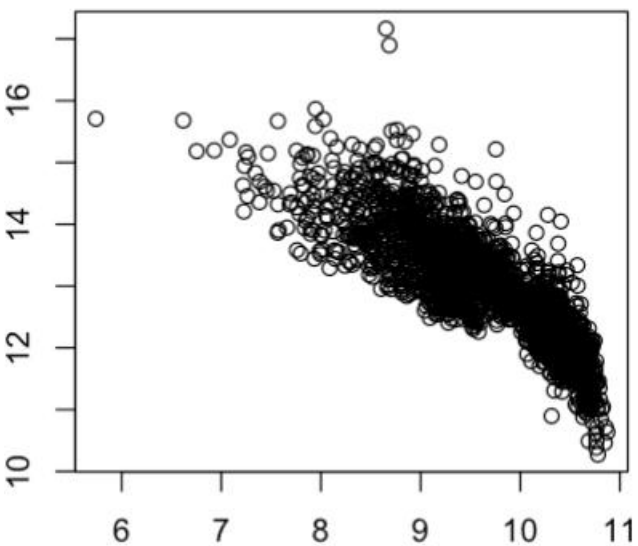
> plot(tk_dist, price)



> plot(log(st_dist), log(price))



> plot(log(tk_dist), log(price))



線形回帰モデルの推定結果

住宅地価(対数)　最寄駅距離(対数)　東京駅距離(対数)　1km²メッシュ内の各土地利用の面積(m²)　データ名

```
> mod <- lm(log(price) ~ log(st_dist) + log(tk_dist) + rice + agri + forest + wild + building + road + rail + river, data = dprice3)
> summary(mod)
```

回帰係数

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.282e+01  2.131e-01 107.100  < 2e-16 ***
log(st_dist) -2.863e-01  1.282e-02 -22.332  < 2e-16 ***
log(tk_dist) -8.208e-01  1.601e-02 -51.258  < 2e-16 ***
rice         -1.542e-06  1.173e-06  -1.315  0.18874
agri         -1.303e-06  2.328e-07  -5.600 2.43e-08 ***
forest       -1.046e-06  1.771e-07  -5.907 4.06e-09 ***
wild         -3.198e-06  2.045e-06  -1.564  0.11803
building     -1.903e-09  1.355e-07  -0.014  0.98879
road         -7.711e-07  2.856e-07  -2.700  0.00699 **
rail          2.186e-07  3.145e-07   0.695  0.48716
river        -1.113e-06  1.696e-07  -6.562 6.65e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4045 on 2100 degrees of freedom
Multiple R-squared:  0.7956,    Adjusted R-squared:  0.7946
F-statistic: 817.2 on 10 and 2100 DF,  p-value: < 2.2e-16
```

有意性(効果の有無)

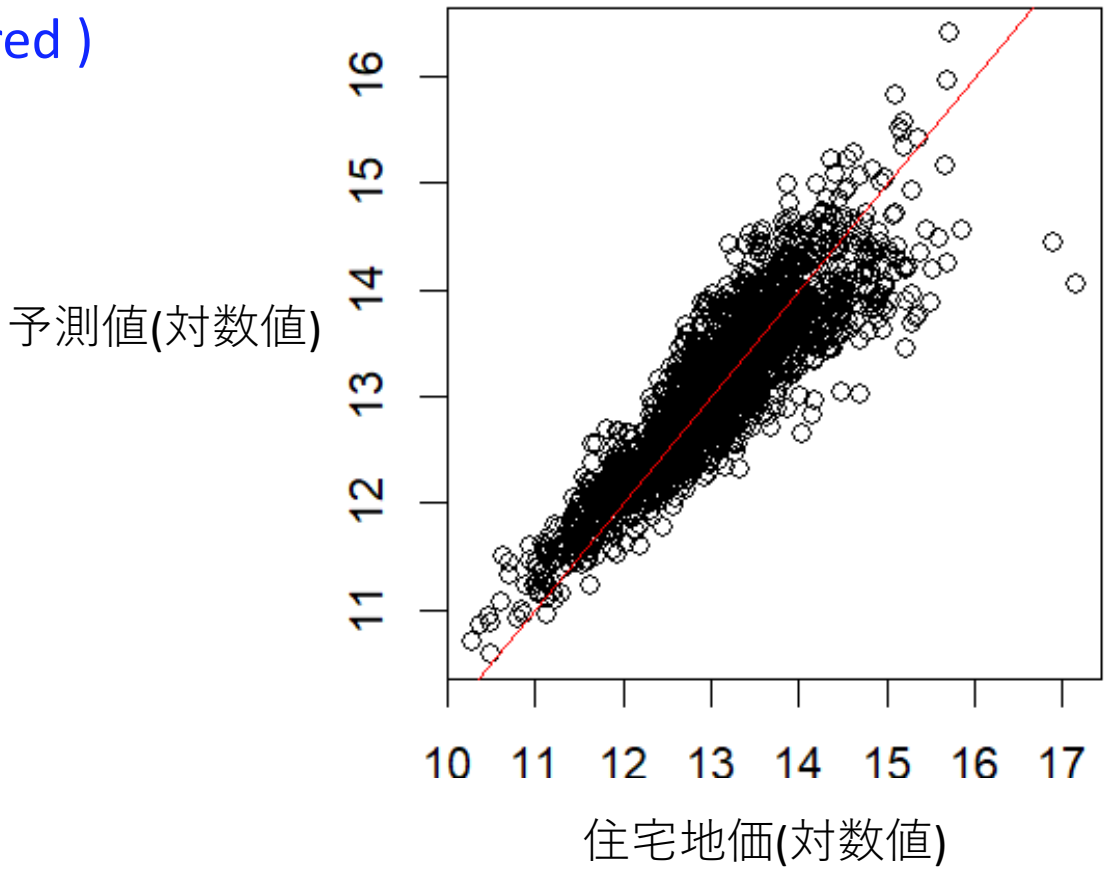
***	: 0.1%水準で有意
**	: 1%水準で有意
*	: 5%水準で有意
	: 0.1%水準で有意

自由度調整済み決定係数

地価の変動の79.46%が説明された

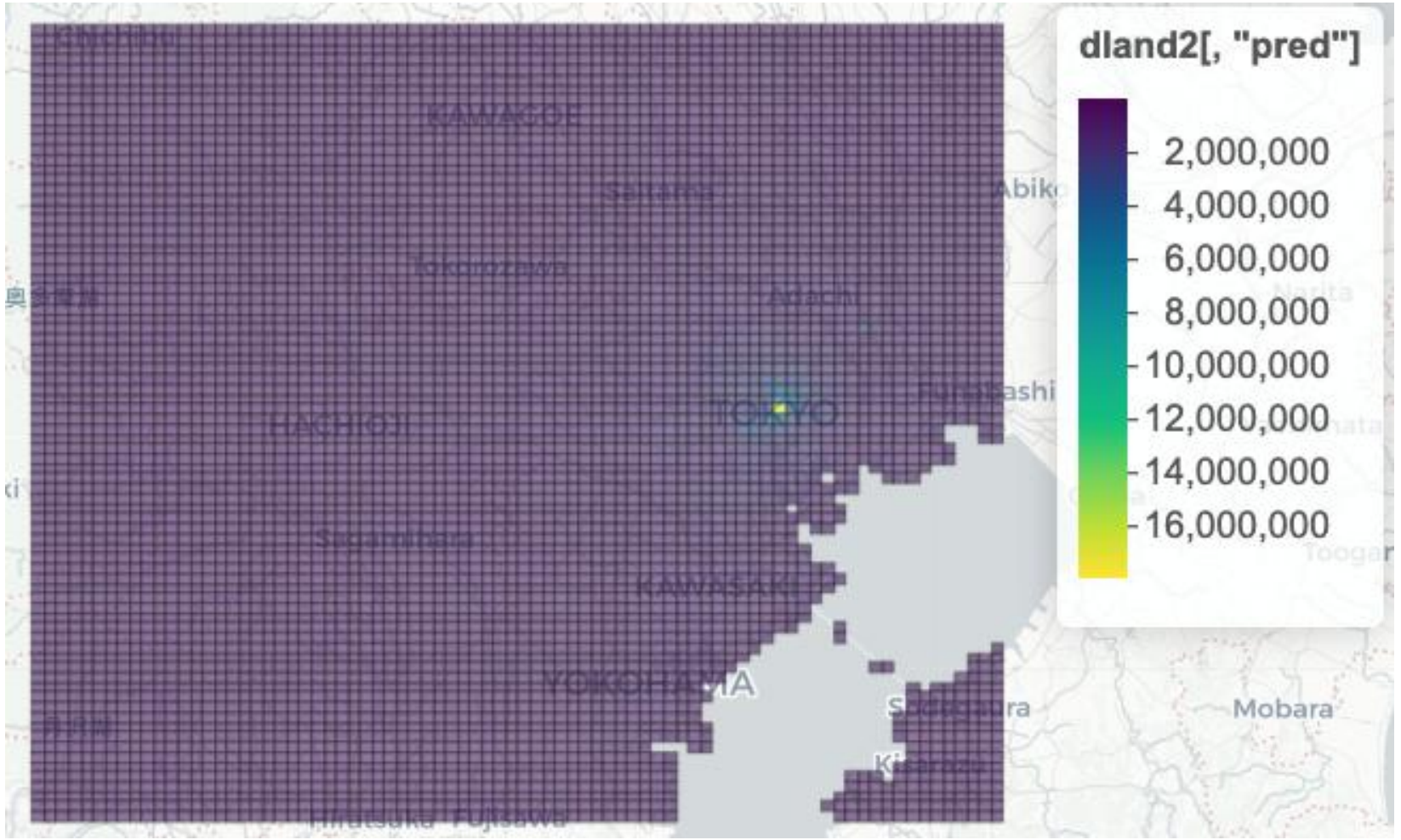
実際の住宅地価と予測値の比較

```
> log_pred        <- predict(mod)
> plot( log(dprice3$price), log_pred )
> abline(0,1,col="red")
```



地価の予測

```
> log_pred0     <- predict(mod,newdata=dland2) # 土地利用3次メッシュ毎の住宅地価(対数値)の予測値
> dland2$pred   <- exp(log_pred0)              # 住宅地価(実数値)の予測値
> mapview(dland2[,“pred”])                    # プロット
```



本日の地価分析の流れ

(1)データ収集
- 住宅地価, 土地利用, 鉄道駅,...

(2)データ整備
- 各地価調査地点の土地利用、最寄駅距離の計算,...

(3)分析
- 回帰分析, 予測,...

(4)地図化
- 地価の予測値の地図化

→ 典型的なワークフロー
→ 以上の一連の流れを
Rで行う方法を紹介

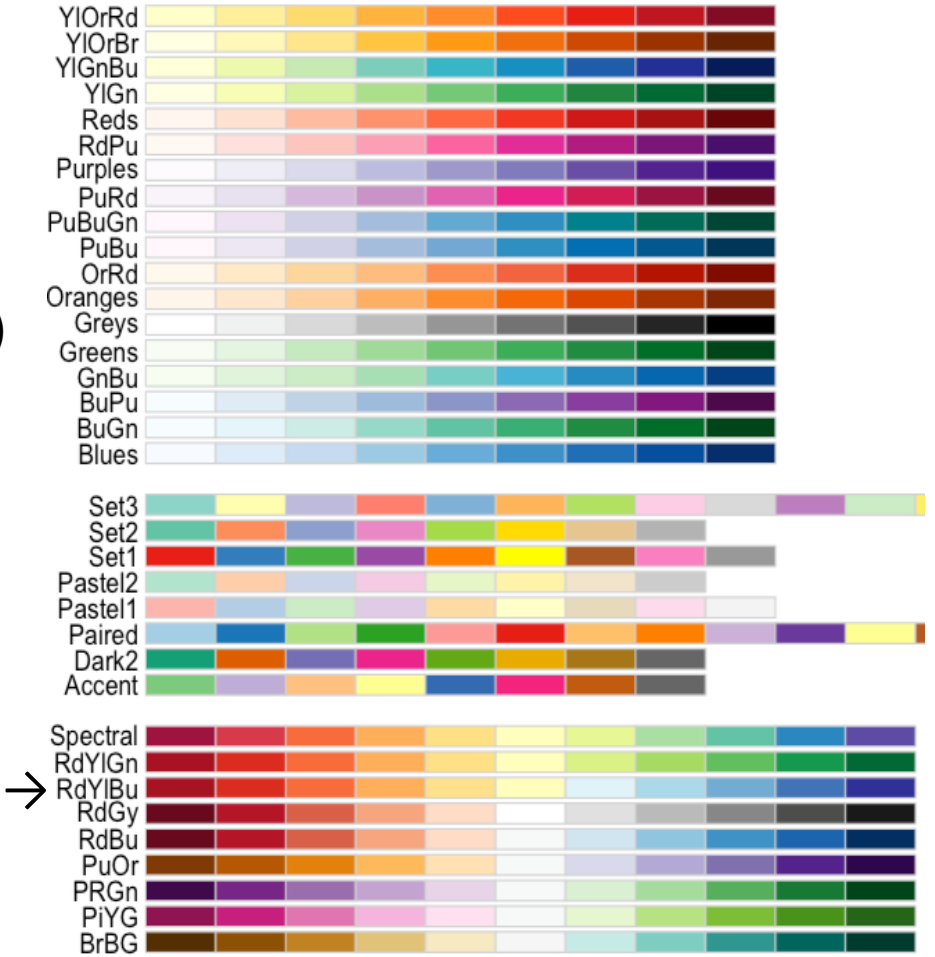
カラーパレットの作成 (RColorBrewerパッケージ)

カラーパレットの生成

```
> breaks  <-c(0,100000,200000,300000,400000,
              500000,600000,800000,1200000,
              2000000, max(dland2$pred))          # 色の区切り位置
> pal0     <- brewer.pal(length(breaks)-1, “RdYlGn”)# パレット生成
> pal      <- rev( pal0 )                         # 色を逆順に(赤=大きい)
```

カラーパレットの確認

```
> display.brewer.all()
```

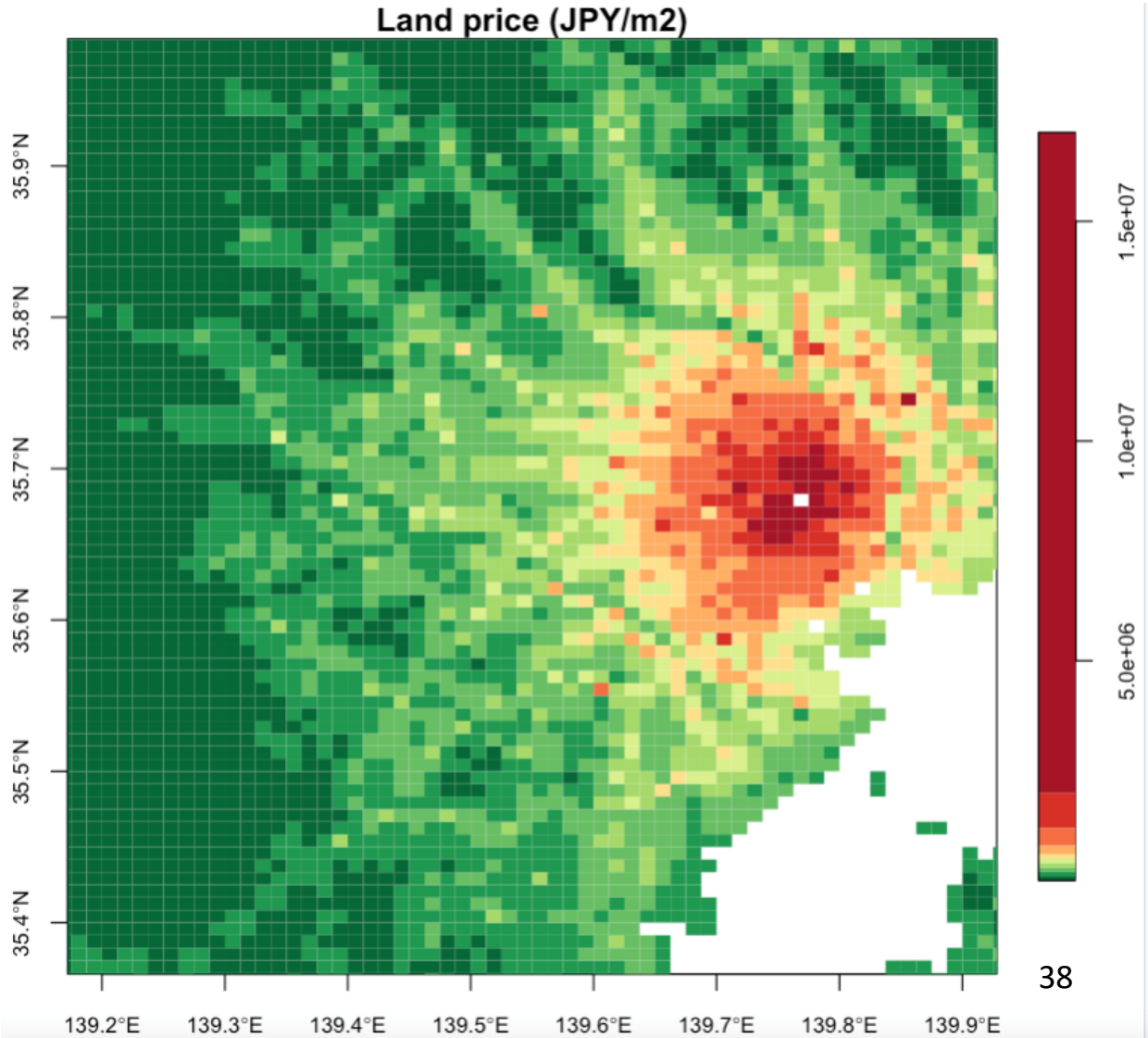


RdYlGn(Red-Yelow-Greenの略)→

plot関数を用いた地図化

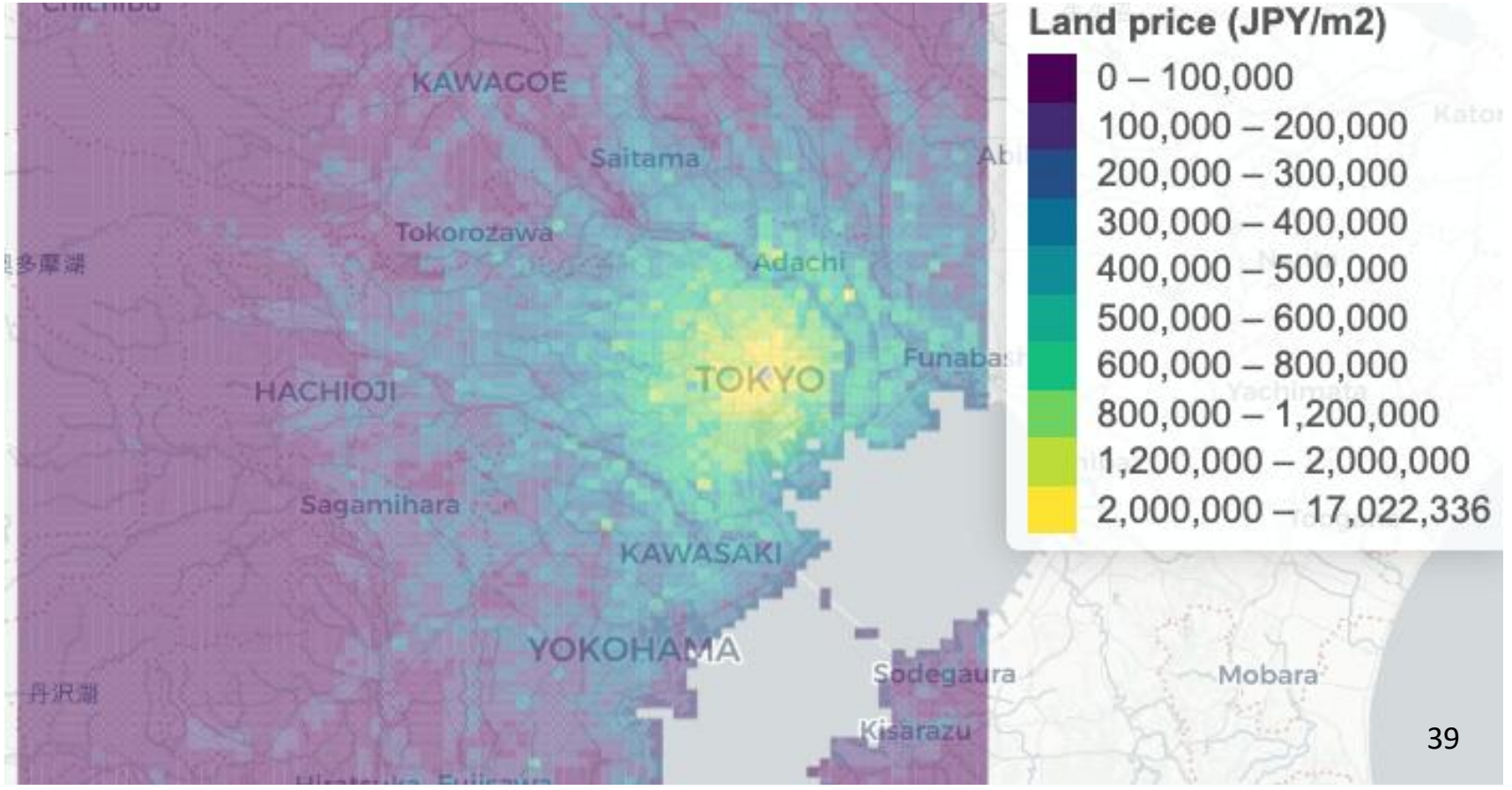
- レポートなどの地図作成に便利

```
plot(dland2[,"pred"],  # 住宅地価をプロット
    axes = TRUE,       # XY座標値を表示
    cex.axis = 0.8,    # XY座標値のフォントサイズ
    pal = pal,         # カラーパレット
    breaks = breaks,   # 色の区切り位置
    pch = 20,          # シンボル(例：20は円、15は四角)
    cex = 0.8,         # シンボルのサイズ(デフォルトは1)
    key.pos = 4,       # 凡例の位置（4は図の右）
    key.length = 0.8,  # 判例の長さ（1=図の幅）
    xlim = c(139.2, 139.9), ylim = c(35.5, 35.85),  # 表示範囲
    border=NA,         # メッシュの枠線を消す
    main = “Land price (JPY/m2)”) #タイトル
```



```
> mapview(dland2[,"pred"],   # 住宅地価をプロット
    cex=4,                   # プロットするシンボルのサイズ
    lwd=0,                   # シンボルの枠線の太さ(0は枠線なし)
    at = breaks,             # 色の区切り位置
    alpha.regions=0.8,       # 透過度（1は透過なし）
    ledgend=TRUE,            # TRUEの場合、凡例を表示
    layer.name="Land price (JPY/m2)") # タイトル
```

mapviewを用いた地図化



RとGIS：勉強に役立つサイト

高野佳佑先生(4月から一橋SDS研究科)公開のページの最初にまとめられています

- その無茶振り, (Rで) GISが解決します： https://rpubs.com/k_takano
- 『事例で学ぶ経済・政策分析のためのGIS入門』のためのR入門：空間解析・可視化編：https://rpubs.com/jirei_de_gis/c_a_2_0_2

RによるGIS

sfパッケージとdplyrパッケージを組み合わせたGISに関して，さらに詳細・高度なジオプロセシングを解説しているWEBサイトをいくつか紹介します（作成者名の敬称略）．日本語のサイトとしては，以下がおすすめです．

- Rを使った地理空間データの可視化と分析 (S. Uryu)
- Rで行う公示地価パネルデータの整備と分析 (S. Kuroda)
- もしもArcGISのジオプロセシングをRで実装したら (K. Takano)

このチュートリアルは「もしも」をベースとして同じ筆者によって作成されたものですが，筆者によるこれらチュートリアルの当初のコンセプトは，「Rで行う公示地価パネルデータの整備と分析」の内容を，扱う空間データ操作の種類を増やしたり，dplyrで処理を記述する等によって補完するというものでした．英語のサイトとしては，以下がおすすめです．

- Geocomputation with R (R. Lovelace, J. Nowosad, J. Muenchow)
- Introduction to Spatial Data Programming with R (M. Dorman)
- R as GIS for Economists (T. Mieno)
- r-spatial (E. Pebesma, M. Appel, D. Nüst)
- Spatial Data Science (E. Pebesma, R. Bivand)
- Using Spatial Data with R (C.A. Engel)

今後の予定等

内容

- 第2回(4/21 月)　：空間データの処理・地図化
- 第3回(4/28 月)　：探索的空間データ解析
- 第4回(5/8, 木)　：空間計量経済モデルと応用

↑

各回で統計ソフトウェアRを用いた実例を紹介

Rコード置き場

https://github.com/dmuraka/HIAS_class

- 質問等は村上(dmuraka@ism.ac.jp)までご連絡ください