Spatial Data and Analysis

Discussion 7

Felipe González

UC Berkeley

October 23*th*

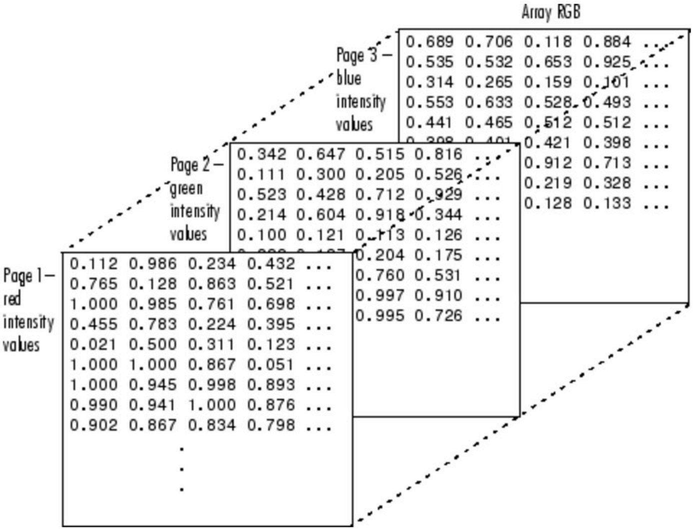
Outline

1. Multidimensional arrays
2. Regressions
3. Moving window regressions

Multidimensional arrays

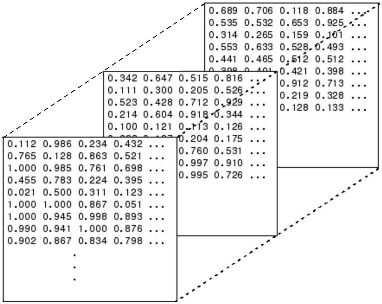
- A $N \times M$ matrix is also called a “2D Array” in Matlab
- 2D Arrays are the basic unit to construct multidimensonal arrays (more than 2 dimensions)
- This is appropriate for our interest because we care about geographic space, esentially a 2D space

3D Array

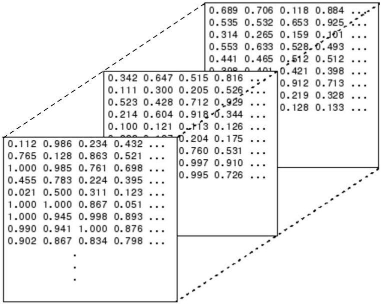


Array RGB
Page 3 – blue intensity values
0.689 0.706 0.118 0.884 ...
0.535 0.532 0.653 0.925 ...
0.314 0.265 0.159 0.101 ...
0.553 0.633 0.528 0.493 ...
0.441 0.465 0.512 0.512 ...
0.421 0.398 ...
0.912 0.713 ...
0.219 0.328 ...
0.128 0.133 ...
Page 2 – green intensity values
0.342 0.647 0.515 0.816 ...
0.111 0.300 0.205 0.526 ...
0.523 0.428 0.712 0.929 ...
0.214 0.604 0.918 0.344 ...
0.100 0.121 0.113 0.126 ...
0.204 0.175 ...
0.760 0.531 ...
0.997 0.910 ...
0.995 0.726 ...
Page 1 – red intensity values
0.112 0.986 0.234 0.432 ...
0.765 0.128 0.863 0.521 ...
1.000 0.985 0.761 0.698 ...
0.455 0.783 0.224 0.395 ...
0.021 0.500 0.311 0.123 ...
1.000 1.000 0.867 0.051 ...
1.000 0.945 0.998 0.893 ...
0.990 0.941 1.000 0.876 ...
0.902 0.867 0.834 0.798 ...

4D Array



0.689 0.706 0.118 0.884 ...
0.535 0.532 0.653 0.925 ...
0.314 0.265 0.159 0.101 ...
0.553 0.633 0.528 0.493 ...
0.441 0.465 0.512 0.512 ...
0.421 0.398 ...
0.912 0.713 ...
0.219 0.328 ...
0.128 0.133 ...
0.342 0.647 0.515 0.816 ...
0.111 0.300 0.205 0.526 ...
0.523 0.428 0.712 0.929 ...
0.214 0.604 0.918 0.344 ...
0.100 0.121 0.113 0.126 ...
0.204 0.175 ...
0.760 0.531 ...
0.997 0.910 ...
0.995 0.726 ...
0.112 0.986 0.234 0.432 ...
0.765 0.128 0.863 0.521 ...
1.000 0.985 0.761 0.698 ...
0.455 0.783 0.224 0.395 ...
0.021 0.500 0.311 0.123 ...
1.000 1.000 0.867 0.051 ...
1.000 0.945 0.998 0.893 ...
0.990 0.941 1.000 0.876 ...
0.902 0.867 0.834 0.798 ...



0.689 0.706 0.118 0.884 ...
0.535 0.532 0.653 0.925 ...
0.314 0.265 0.159 0.101 ...
0.553 0.633 0.528 0.493 ...
0.441 0.465 0.512 0.512 ...
0.421 0.398 ...
0.912 0.713 ...
0.219 0.328 ...
0.128 0.133 ...
0.342 0.647 0.515 0.816 ...
0.111 0.300 0.205 0.526 ...
0.523 0.428 0.712 0.929 ...
0.214 0.604 0.918 0.344 ...
0.100 0.121 0.113 0.126 ...
0.204 0.175 ...
0.760 0.531 ...
0.997 0.910 ...
0.995 0.726 ...
0.112 0.986 0.234 0.432 ...
0.765 0.128 0.863 0.521 ...
1.000 0.985 0.761 0.698 ...
0.455 0.783 0.224 0.395 ...
0.021 0.500 0.311 0.123 ...
1.000 1.000 0.867 0.051 ...
1.000 0.945 0.998 0.893 ...
0.990 0.941 1.000 0.876 ...
0.902 0.867 0.834 0.798 ...

From 4D to 3D

- Create a fake 4D Array measuring the level of happiness (0–10) in a 2D space across two socioeconomic groups in two years

```
        A             = 5 + 5*rand(10,10);
        A(:,:,2)      = 5 + 5*rand(10,10);
        A(:,:,1,2) = A(:,:,1) + 1*rand(10,10);
        A(:,:,2,2) = A(:,:,2) + 1*rand(10,10);
```

- 1D is lon, 2D lat, 3D socioeconomic group, 4D year

Statistics in Arrays

- Calculate average happiness by socioeconomic group:

```
B = mean(A, 4);
disp('Happiness in year 1');
disp(mean(mean(B(:,:,1))))
disp('Happiness in year 2');
disp(mean(mean(B(:,:,2))))
```

Statistics in Arrays

- Calculate average happiness each year:

```
C = mean(A, 3);
disp('Happiness in year 1');
disp(mean(mean(C(:,:,1,1))))
disp('Happiness in year 2');
disp(mean(mean(C(:,:,1,2))))
```

Outline

1. Multidimensional arrays
2. Regressions
3. Moving window regressions

Regressions

- Understanding regressions:

$$Y \quad = \quad f(\mathbf{V})$$

Regressions

- Understanding regressions:

$$
\begin{aligned}
Y &= f(\mathbf{V}) \\
Y &= f(\mathbf{W}, \varepsilon)
\end{aligned}
$$

Regressions

- Understanding regressions:

$$
\begin{aligned}
Y &= f(\mathbf{V}) \\
Y &= f(\mathbf{W}, \varepsilon) \\
Y &= g(\mathbf{W}) + \varepsilon
\end{aligned}
$$

Regressions

- Understanding regressions:

$$
\begin{aligned}
Y &= f(\mathbf{V}) \\
Y &= f(\mathbf{W}, \varepsilon) \\
Y &= g(\mathbf{W}) + \varepsilon \\
Y &= \mathbf{\Theta}'\mathbf{W} + \varepsilon
\end{aligned}
$$

Regressions

- Understanding regressions:

$$\begin{aligned} Y &= f(\mathbf{V}) \\ Y &= f(\mathbf{W}, \varepsilon) \\ Y &= g(\mathbf{W}) + \varepsilon \\ Y &= \boldsymbol{\Theta}'\mathbf{W} + \varepsilon \\ Y &= \alpha + \beta X + \varepsilon \end{aligned}$$

Estimating regressions

- The most widely used technique to estimate a linear regression is ordinary least squares (OLS)
- When using OLS, we choose $\mathbf{\Theta}$ to minimize the objective function $(Y - \mathbf{\Theta X})^2$, where $\mathbf{X}$ equals a vector of ones and X, and $\mathbf{\Theta} = (\alpha \;\; \beta)$ in the previous example
- It is easy to show that a consistent estimator of $\mathbf{\Theta}$ is:

$$\widehat{\mathbf{\Theta}} \equiv \begin{pmatrix} \widehat{\alpha} \\ \widehat{\beta} \end{pmatrix} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'Y$$

Regression and correlation

- In Lab 6, we want to estimate β using OLS. After estimation, we will obtain $\widehat{\alpha}$ and $\widehat{\beta}$
- The estimate $\widehat{\beta}$ is not *necessarily* equal to the correlation between Y and X (i.e., $\rho(Y, X)$)
- In the particular case in which $\sigma(Y) = \sigma(X)$ both are, however, exactly the same. The relation between both is:

$$\rho(Y, X) \;=\; \beta \cdot \frac{\sigma(X)}{\sigma(Y)}$$

Example of a regression in Matlab

```
% Fake data
    N = 1000 ;
    X = randn(N,1) ;
    Y = 5 + 2*X + randn(N,1);
% Regression
    coef  = regress(Y,[ones(N,1), X]);
    alpha = coef(1,1);
    beta  = coef(2,1);
% Answer
    disp('OLS estimate:')
    disp(beta)
    disp('Correlation')
    disp(beta*(std(X)/std(Y)))
```

Outline

1. Multidimensional arrays

2. Regressions

3. Moving window regressions

Moving window regression

- **What is a moving window regression?**

  A moving window regression is a local regression, where "local" means using a subsample of the data

- **How does it relates to a "normal" regression?**

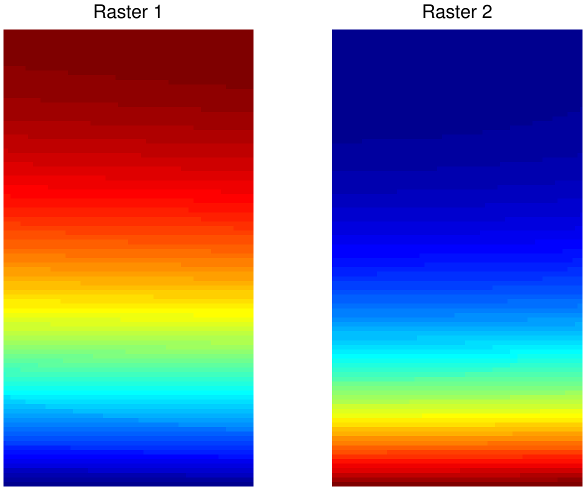
  In general, an OLS regression produces a single parameter, which is a weighted average of many "local" regressions. A moving window regression is like disaggregating the estimates from a "normal" regression

Creating two fake rasters

```
% Creates grid
    X_axis = 0:1:100 ;
    Y_axis = 0:1:100 ;
    [X_grid, Y_grid] = meshgrid(X_axis, Y_axis) ;

% Creates fake rasters
    R1 = X_grid  + 2*Y_grid - Y_grid.^2;
    R2 = Y_grid.^3 + X_grid.^2 + sqrt(Y_grid);
```

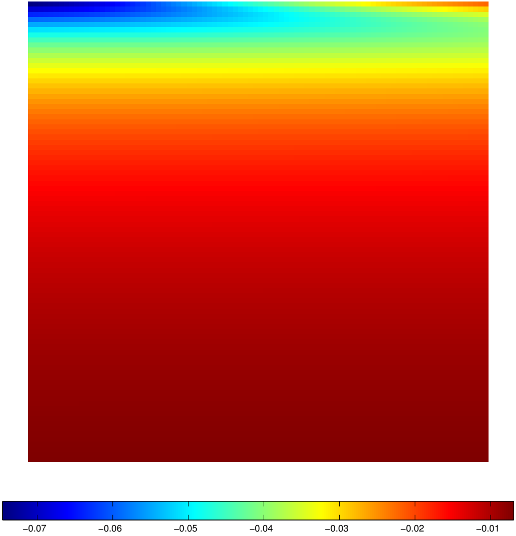
Visualizing our fake rasters



Algorithm structure for moving window regression

1. Select a point (x, y) in the grid where rasters are located
2. Using the rasters, select data of interest
3. Reshape data to be able to use it in regression
4. Estimate linear regression with a constant term
5. Extract slope coefficient
6. Construct and save correlation in location (x, y)
7. Repeat for every point in the raster

Visualizing correlations



Tips

- Look for the function `sc` in File Exchange at Matlab Central to solve the `NaN` problem when plotting the data
- Don't forget that the slope parameter is different from the correlation, and the adjustment is simple