CISC 3250
Systems Neuroscience

Matlab, part 2:
2⁺D matrices,
visualizations

Professor Daniel Leeds
dleeds@fordham.edu
JMH 332

Matrices: rows and columns

```
B=[2.2 1.4; -5 3.5; -7.8 4.3];
```

$$\begin{bmatrix} 2.2 & 1.4 \\ -5 & 3.5 \\ -7.8 & 4.3 \end{bmatrix}$$

- Spaces/commas separate columns
- Semi-colons (;) separate rows
- `name(row,col)` accesses single element

`B(2,1)` *returns* `-5`

3

Matrix indexing

Assume we have a 10x500 matrix of spike patterns for 10 neurons `spikeMat`

- `spikeMat(1,:)` contains spikes for neuron 1
- `spikeMat(4,:)` contains spikes for neuron 4

In general:

- `name(:,col)` accesses all elements in column

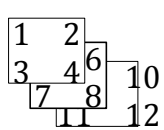
4

Matrices in *n* dimensions

```
x=[1 2 3; 4 5 6]
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

```
y(:,:,1)=[1 2; 3 4]
y(:,:,2)=[5 6; 7 8]
y(:,:,3)=[9 10; 11 12]
size(y) -> [2 2 3]
```



Typical brain data : location of neurons (x,y,z)
or location of neurons + time

Heat-maps

imagesc(Data) – view 2D matrix of scaled data as image

- Red/yellow is highest value, blue is lowest value

Visualize a 2D slice of brain data (size(brainData) -> 128x128x88)

```
slice=squeeze(brainData(:,:,20))
```

-> slice 20 of brain

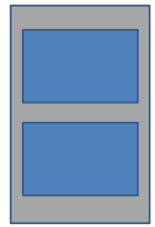
```
imagesc(slice)
```

Multiple plots

figure -> opens new plotting window

subplot(r,c,i) -> creates grid of plots with

- r rows
- c columns
- fill in position i



```
subplot(2,1,1); plot(vec1);
subplot(2,1,2); plot(vec2);
```

9

`squeeze` out 1-entry dimensions

```
currSlice=squeeze(brainData(:,60,:));

% currSlice has size 128x1x88
% won't be plotted by imagesc
%  - expects 2D matrix

currSlice=squeeze(currSlice);
% now currSlice has size 128x88
```

10

Multiple plots

figure -> opens new plotting window

subplot(r,c,i) -> creates grid of plots with

- r rows
- c columns
- fill in position i

```
currSlice=squeeze(brainData(:,:,10));
subplot(1,3,1); imagesc(currSlice);
currSlice=squeeze(brainData(:,:,20));
subplot(1,3,2); imagesc(currSlice);
currSlice=squeeze(brainData(:,:,30));
subplot(1,3,3); imagesc(currSlice);
```

11

Saving graphics results

- `print -dpng filename.png`
- `print -djpg filename.jpg`

12

Multiple plots

With looping

```
for i=1:32,
  currSlice=squeeze(brainData(:,i*4,:));
  subplot(4,8,i), imagesc(currSlice);
end;
```

13

Scaling vs. not-scaling

`imagesc(Data)` – view 2D matrix of scaled data as image

- Yellow (or red) is highest value, blue is lowest value

`image(Data)` – view 2D matrix of data as image

- Yellow (or red) is >=256 (or 64), blue is <=0

```
slice=squeeze(brainData(:,:,10));
figure; imagesc(slice);
```

vs

```
figure; image(slice)
```

Finding desired values

`find(vector<number)` `find(c<2)`

Return indices in `vector` that are less than `number`

Example:

```
vector=[5, -1, 0, 12];
smallLocations=find(vector<2);
```

Comparisons

- `d<2, d>2` strict inequality
- `d<=2, d>=2` semi-inequality
- `d==2` equality

16

Computing mask matrix

```
% grab slice
slice=squeeze(S1brain(:,44,:));
% find body inds
hiValues = find(slice>=50);

% create new mask, all 0 by default
maskMat = zeros(128,88);
% add in 1s
maskMat(hiValues)=1;
% will have inner circle of 0s
```

17

Skull stripping

```
% isolate column 40 in slice
oneColumn = slice(:,40);
% find zeros in column 40
zeroInds = find(oneColumn==0);
% find the front 0 of brain
frontHalf = find(zeroInds<=64);
frontInd = zeroInds(end);
% find the back 0 of brain
backHalf = find(zeroInds>64);
backInd = zeroInds(1);
% set non-brain data to 0
slice(1:frontInd,40) = 0;
slice(backInd:end,40) = 0;
```

errors from Dr Leeds

18

Skull stripping

```
% isolate column 40 in slice
oneColumn = maskMat(:,40);
% find zeros in column 40
zeroInds = find(oneColumn==0);
% find the front 0 of brain
frontHalf = find(zeroInds<=64);
frontInd = zeroInds(frontHalf(end));
% find the back 0 of brain
backHalf = find(zeroInds>64);
backInd = zeroInds(backHalf(1));
% set non-brain data to 0
slice(1:frontInd,40) = 0;
slice(backInd:end,40) = 0;
```

Corrections from Dr Leeds

How can we loop for all cols?

19

Skull stripping

```
for col=1:88,
  oneColumn = maskMat(:,col);
  % find zeros in front/back
  zeroInds = find(oneColumn==0);
  frontHalf = find(zeroInds<=64);
  frontInd = zeroInds(frontHalf(end));
  backHalf = find(zeroInds>64);
  backInd = zeroInds(backHalf(1));
  % set non-brain data to 0
  slice(1:frontInd,col)=0;
  slice(backInd:end,col)=0;
end;
```

How can we loop for all cols?

20