

1. Array Creation and Operations

- (a) Create the following array $\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 3 & 1 & 1 \\ 1 & 1 & 1 & 4 & 1 \end{bmatrix}$

```
A = np.ones((4,5))
A[np.arange(1,4), np.arange(1, 4)] = np.arange(2,5)
```

- (b) Compute the row sums of the above matrix
`A.sum(axis=0)`
- (c) Compute the column sums of the above matrix
`A.sum(axis=1)`
- (d) Download and read into memory the matrix found below. Check that it is equal to the array you created above.

http://stanford.edu/~arbenson/cme193/data/lec4_array.txt

```
import os
os.system("wget http://stanford.edu/~arbenson/cme193/data/lec4_array.txt")
B = np.loadtxt('lec4_array.txt', skiprows=1, comments = '%', delimiter=',')
np.all(B == A)
```

2. Array Slicing and Indexing

Using the array above return the second and third rows and the columns containing an even number as a 2×2 array using...

- (a) integer indexes
`A[[1, 2],...][..., [1, 3]]`

- (b) slices
`A[1:3, 1:4:2]`

- (c) boolean arrays
`A[np.array([False, True, True, False]), ...][...,
 np.array([False, True, False, True, False])]`

- (d) boolean arrays computed from the array
`ind = np.apply_along_axis(lambda x: np.any(x % 2 == 0), 0, A)
 A[1:3, ind]`

3. Broadcasting

Using the above array assigned as *arr*, describe the following operations

- (a) `arr * 5`.
 Multiplies every element in *arr* by 5
- (b) `arr * np.arange(arr.shape[1])`
 Scales the columns of *arr* by 0, 1, 2, 3, 4 respectively
- (c) `arr * np.arange(arr.shape[0])`
 Error, operation does not broadcast
- (d) `arr.T * np.arange(arr.shape[0])`
 Scales the rows of *arr* by 0, 1, 2, 3 respectively and returns the transpose of *arr* scaled in this way.

- (e) compute the dot product of the array with $\begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$ in two ways
- ```
np.sum(arr * np.arange(5), axis=1)
np.dot(arr, np.arange(5))
```