4741

Home work 1 (Question 2)

Submitted by

Aman jain (aj644), Vineel Yellapantula (vy45)

Question 2

Answer 2 (a)

Yes the shape of two products is different. (u'*v) is a matrix, an array of size (1, 1). while (x*'y) is a scalar of float type.

It looks to be applying same rule, ie sum of product of elements in both cases. Size of x and y is defined as (3,) (ie vector) where as size of u and v is taken as (3,1) (ie matrix)

It looks to be following a rule that product of 2 matrices is a matrix, and dot porduct of two "vectors" (in this context) is a scalar.

In [1]:
```
u = rand(3 ,1)
u
```

Out[1]:
```
3×1 Array{Float64,2}:
 0.7894170949553805
 0.3528473293002832
 0.08325422475727762
```

In [2]:
```
v = rand(3 ,1)
v
```

Out[2]:
```
3×1 Array{Float64,2}:
 0.9571495487797506
 0.1062443217035014
 0.9666046326208866
```

In [3]:
```
u'*v
```

Out[3]:
```
1×1 Array{Float64,2}:
 0.8735521607376096
```

In [4]:
```
size(u'*v)
```

Out[4]:
```
(1, 1)
```

In [5]:
```
x = rand(3)
x
```

Out[5]:
```
3-element Array{Float64,1}:
 0.5041584738939358
 0.19434770283716363
 0.08066628784014229
```

In [6]:
```
y = rand(3)
y
```

Out[6]:
```
3-element Array{Float64,1}:
 0.6388418218902399
 0.5275715154762064
 0.9285253827596363
```

In [7]:
```
x'*y
```

Out[7]:
```
0.4995105258914938
```

In [8]:
```
size(x'*y)
```

Out[8]:
```
()
```

In [9]:
```
size(v)
```

Out[9]:
```
(3, 1)
```

Answer 2 (b)

dot(u,v) returns a scalar which is sum of the product of values corresponding indices in u and v. Its size is empty. sum(u.*v) basically returnes the sum of elements of 31 matrix (u.*v). this is a scalar number. sum(x'y) returns same scalar as x'*y. Which is expected as sum of a scalar will be that scalar itself.

size for both is null - which indicates a scalar.

In [16]:
```
using LinearAlgebra
dot(u,v)
```

Out[16]:
```
0.8735521607376096
```

In [17]:
```
sum(u.*v)
```

Out[17]:
```
0.8735521607376096
```

In [18]:
```
size(sum(u.*v))
```

Out[18]:
```
()
```

In [19]:
```
dot(x,y)
```

Out[19]:
```
0.4995105258914938
```

In [20]:
```
size(dot(x,y))
```

Out[20]:
```
()
```

In [21]:
```
sum(x'*y)
```

Out[21]:
```
0.4995105258914938
```

In [22]:
```
using LinearAlgebra
dot(u,v)
```

Out[22]:
```
0.8735521607376096
```

Answer 2 (c)

Both first column and first rows are of type vector with shape (5,).

In [23]:
```
A = rand(5,5)
```

Out[23]:
```
5×5 Array{Float64,2}:
 0.835459   0.117072  0.559577  0.221404   0.0457756
 0.565814   0.246357  0.907711  0.487894   0.284159
 0.0556159  0.883425  0.2153    0.696645   0.670316
 0.629746   0.251109  0.977536  0.0210967  0.000712038
 0.105401   0.116161  0.302218  0.414807   0.594126
```

In [24]:
```
first_row = A[1,:]
```

Out[24]:
```
5-element Array{Float64,1}:
 0.8354593715506222
 0.11707229096420502
 0.559577063271492
 0.22140431013025963
 0.04577558663905079
```

In [25]:
```
size(first_row)
```

Out[25]:
```
(5,)
```

In [26]:
```
first_col = A[:,1]
```

Out[26]:
```
5-element Array{Float64,1}:
 0.8354593715506222
 0.5658139448331878
 0.05561585225056742
 0.6297455586914775
 0.10540080978271416
```

In [27]:
```
size(first_col)
```

Out[27]:
```
(5,)
```

Answer 2 (d)

Below are the two ways to take the inner product:

sum(first_row.*first_col) and

first_row'*first_col

In [28]:
```
sum(first_row.*first_col)
```

Out[28]:
```
0.9396080164463062
```

In [29]:
```
first_row'*first_col
```

Out[29]:
```
0.9396080164463062
```

In [ ]:

In [ ]: