CS2035B Assignment 2: Arrays and Efficiency

Table of Contents

Identificationx

Your Name: Lishan Huang Your Student Number 250777962

avg1 Source Code

```
function x = avg1(A)
%get the size of matrix A
[n,m]=size(A);
%create a array with empty infomation
x=[];
%detect whether is  a one row or not
%if is, transfer it into a column
if n==1 && m >1
    A=A';
    b=n;
    n=m;
    m=b;
end
%use for loop the get the sum of each column
for i=1:m
    sum=0;
    %the sum of column
    for a=1:n
        sum=sum+A(a,i);
    end
    %return the average of each column
    x(i)=sum/n;
end
```

avg2 Source Code

```
function x = avg2(A)
%get the size of matrix
[n,m]=size(A);
%detect whether is a one row matrix or not
if n==1 && m >1
    A=A';
    b=n;
    n=m;
    m=b;
end
%initialize the size and the type of output
x=zeros(1,m);
%for loop the get the average of each column
for i=1:m
    sum=0;
    for a=1:n
        sum=sum+A(a,i);
    end
    x(i)=sum/n;
end
```

avg3 Source Code

```
function x = avg3(A)
%get the size of matrix A
[n,m]=size(A);
%detect whether the matrix is a one row matrix or not
if n==1 && m >1
    A=A';
    b=n;
    n=m;
    m=b;
end
%sum up each column and devide the size of each column
x=sum(A)/n;
```

stderr1 Source Code

```
function x = stdder1(A)
%get the size of matrix A
[n,m]=size(A);
```

```
%get the average number of each column of matrix A
average=avg1(A);
%create a array with empty information
x=[];
%get the size of the result of funtion A
[a,b]=size(average);
%detect whether is  a one row or not
%if is, transfer it into a column
if n==1 && m >1
    A=A';
    B=n;
    n=m;
    m=B;
end
%Computing Standard Errors Efficiently by for loop
for i=1:m
    dif=0;
    %sum up the different between the value of each element and the
 average
    %number
    for a=1:n
        dif=dif+(average(1,i)-A(a,i))^2;
    end
    %calculate the square root of the sum devides n-1
    x(i)=sqrt(dif/(n-1));
end
```

stderr2 Source Code

```
function x = stdder2(A)
%get the size of the result of funtion A
[n,m]=size(A);
%get the average number of each column of matrix A
average=avg3(A);
[a,b]=size(average);
%detect whether is  a one row or not
%if is, transfer it into a column
if n==1 && m >1
    A=A';
    B=n;
    n=m;
    m=B;
end
```

```
%initialize the size and the element type of x
x=zeros(1,m);
%Computing Standard Errors Efficiently by for loop
for i=1:m
    dif=0;
      %sum up the different between the value of each element and the
 average
    %number
    for a=1:n
        dif=dif+(average(1,i)-A(a,i))^2;
    end
    %calculate the square root of the sum devides n-1
    x(i)=sqrt(dif/(n-1));
end
```

stderr3 Source Code

```
function x = stderr3(A)
%get the average number of each column of matrix A
average=avg3(A);
%get the average number of each column of matrix A
[n,m]=size(A);
%detect whether is  a one row or not
%if is, transfer it into a column
if n==1 && m >1
    A=A';
    B=n;
    n=m;
    m=B;
end
%use sqrt function to ge the Standard Errors Efficiently
x=sqrt(sum((A-average).^2)./(n-1));
```

tests Source Code

```
%%
x=rand(100,6);
y=rand(1,100);
%% the value of avg1, avg2, avg3 and mean of X and Y
fprintf("avg1 function of x value")
avg1(x)
```

```
fprintf("avg2 function of x value")
avg2(x)
fprintf("avg3 function of x value")
avg3(x)
fprintf("matlab function of x value")
mean(x)

fprintf("avg1 function of y value")
avg1(y)
fprintf("avg2 function of y value")
avg2(y)
fprintf("avg3 function of y value")
avg3(y)
fprintf("matlab function of y value")
mean(y)

%% the value of stderr1, stderr2, stderr3 and function std of X and Y
fprintf("stderr1 function of x value")
stderr1(x)
fprintf("stderr2 function of x value")
stderr2(x)
fprintf("stderr3 function of x value")
stderr3(x)
fprintf("matlab function of x value")
std(x)

fprintf("stderr1 function of y value")
stderr1(y)
fprintf("stderr2 function of y value")
stderr2(y)
fprintf("stderr3 function of y value")
stderr3(y)
fprintf("matlab function of y value")
std(y)


x=rand(100,6);
y=rand(1,100);
fprintf("avg1")
avg1(x)
avg1(y)
fprintf("avg2")
avg2(x)
avg2(y)
fprintf("avg3")
avg3(x)
avg3(y)
fprintf("standard output")
mean(x)
mean(y)
fprintf("stderr1")
stderr1(x)
stderr1(y)
fprintf("stderr2")
```

```
stderr2(x)
stderr2(y)
fprintf("stderr3")
stderr3(x)
stderr3(y)
fprintf("standard output")
std(x)
std(y)

avg1
ans =

    0.5071    0.4563    0.5106    0.5477    0.5233    0.5129


ans =

    0.5285

avg2
ans =

    0.5071    0.4563    0.5106    0.5477    0.5233    0.5129


ans =

    0.5285

avg3
ans =

    0.5071    0.4563    0.5106    0.5477    0.5233    0.5129


ans =

    0.5285

standard output
ans =

    0.5071    0.4563    0.5106    0.5477    0.5233    0.5129


ans =

    0.5285

stderr1
ans =

    0.2583    0.2816    0.2911    0.2823    0.2946    0.3022
```

```
ans =

    0.2888

stderr2
ans =

    0.2583    0.2816    0.2911    0.2823    0.2946    0.3022


ans =

    0.2888

stderr3
ans =

    0.2583    0.2816    0.2911    0.2823    0.2946    0.3022


ans =

    0.2888

standard output
ans =

    0.2583    0.2816    0.2911    0.2823    0.2946    0.3022


ans =

    0.2888
```

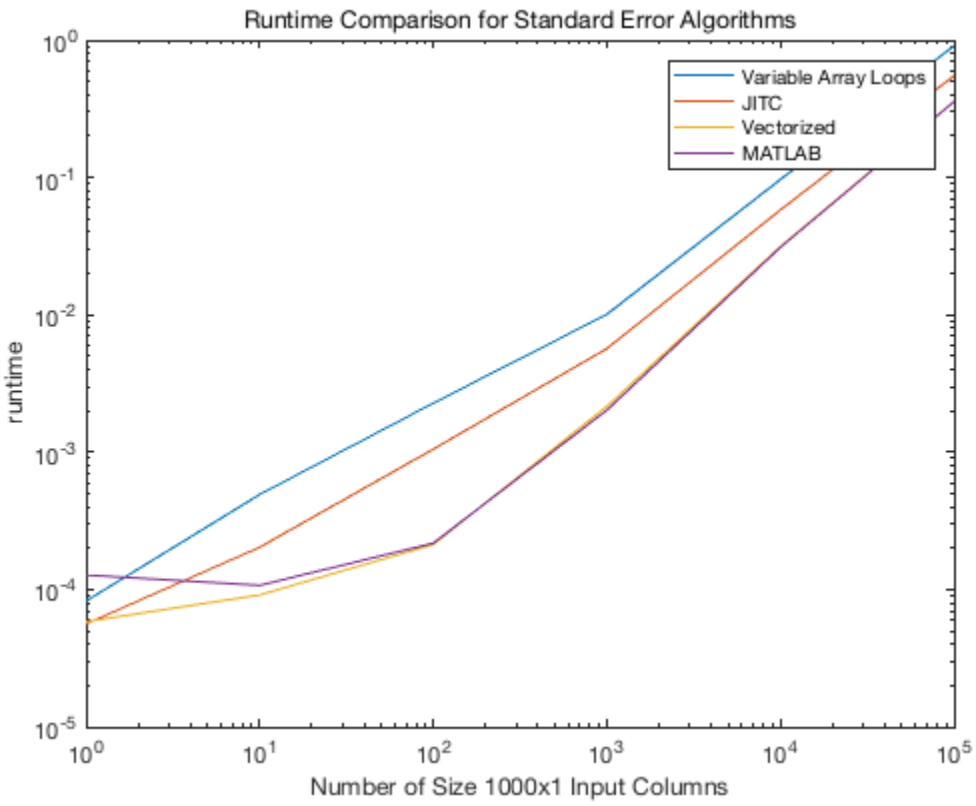
timing Source Code

```
%initialize the running time of the whole function in pow
pow = 5;
points = pow+1;
m = 1000;
%create a row that contains each result of the whole function run
n = logspace(0,pow,points);
T = zeros(4,points);
for k=1:points
    %initialize a matrix X with m row and n(k) column
    X=rand(m,n(k));
    %each funtion run 10 times
runs=10;
for i=1:runs
    %get the result of stderr1 with variable matrix X store it in
 elapsed
```

```
    tic
    stderr1(X);
    elapsed(i) = toc;
end
%get the mean of elapsed and store it in the first row
T(1,k) = mean(elapsed);
%print the mean running time
fprintf('stderr1 on %dx%d array: %f s\n',m,n(k),mean(elapsed));

for i=1:runs
tic
 %get the result of stderr2 with variable matrix X store it in elapsed
stderr2(X);
elapsed(i) = toc;
end
%get the mean of elapsed and store it in the second row
T(2,k) = mean(elapsed);
%print the mean running time
fprintf('stderr2 on %dx%d array: %f s\n',m,n(k),mean(elapsed));

for i=1:runs
tic
 %get the result of stderr3 with variable matrix X store it in elapsed
stderr3(X);
elapsed(i) = toc;
end
%get the mean of elapsed and store it in the third row
T(3,k) = mean(elapsed);
%print the mean running time
fprintf('stderr3 on %dx%d array: %f s\n',m,n(k),mean(elapsed));

for i=1:runs
tic
%get the result of matlab function with variable matrix X store it in
 elapsed
std(X);
elapsed(i) = toc;
end
%get the mean of elapsed and store it in the fourth row
T(4 ,k) = mean(elapsed);
%print the mean running time
fprintf('std     on %dx%d array: %f s\n',m,n(k),mean(elapsed));
end
%plot the result
loglog(n,T)
title('Runtime Comparison for Standard Error Algorithms')
legend ('Variable Array Loops', 'JITC', 'Vectorized','MATLAB')
xlabel('Number of Size 1000x1 Input Columns')
ylabel('runtime')
```

```
stderr1 on 1000x1 array: 0.000082 s
stderr2 on 1000x1 array: 0.000056 s
stderr3 on 1000x1 array: 0.000058 s
std     on 1000x1 array: 0.000127 s
```

```
stderr1 on 1000x10 array: 0.000491 s
stderr2 on 1000x10 array: 0.000203 s
stderr3 on 1000x10 array: 0.000091 s
std     on 1000x10 array: 0.000107 s
stderr1 on 1000x100 array: 0.002271 s
stderr2 on 1000x100 array: 0.001050 s
stderr3 on 1000x100 array: 0.000212 s
std     on 1000x100 array: 0.000217 s
stderr1 on 1000x1000 array: 0.010064 s
stderr2 on 1000x1000 array: 0.005668 s
stderr3 on 1000x1000 array: 0.002153 s
std     on 1000x1000 array: 0.002019 s
stderr1 on 1000x10000 array: 0.096543 s
stderr2 on 1000x10000 array: 0.057925 s
stderr3 on 1000x10000 array: 0.031399 s
std     on 1000x10000 array: 0.030963 s
stderr1 on 1000x100000 array: 0.907084 s
stderr2 on 1000x100000 array: 0.548898 s
stderr3 on 1000x100000 array: 0.356376 s
std     on 1000x100000 array: 0.356092 s
```



Published with MATLAB® R2017b