## Analyze Big Data in MATLAB Using MapReduce

Localizar el fichero que se encuentra en la carpeta de instalación de MATLAB;

**‘airlinesmall.csv’**

Leerlo utilizando la instrucción de lectura de ficheros Excel de Matlab:

**[num,txt] = xlsread(___)**

¿Qué se obtiene?

Buscar el máximo retraso de todos los vuelos midiendo el tiempo empleado

## Analyze Big Data in MATLAB Using MapReduce

Try This Example

This example shows how to use the datastore and mapreduce functions to process a large amount of file-based data. The MapReduce algorithm is a mainstay of many modern "big data" applications. This example operates on a single computer, but the code can scale up to use Hadoop®.

http://stat-computing.org/dataexpo/2009/the-data.html.

A small subset of the data set is also included with MATLAB® to allow you to run this and other examples without downloading the entire data set.

## Analyze Big Data in MATLAB Using MapReduce

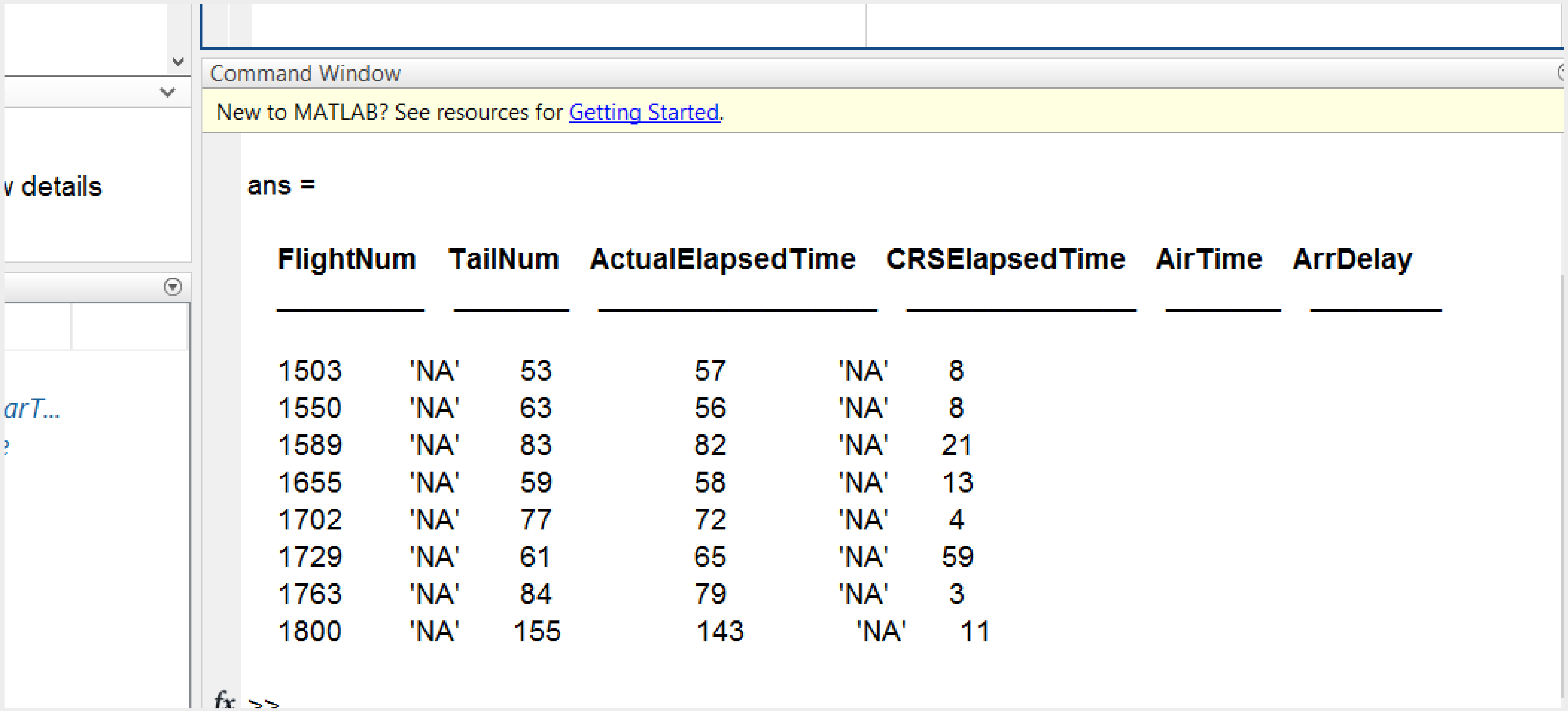
### Introduction to datastore

Creating a datastore allows you to access a collection of data in a chunk-based manner. A datastore can process arbitrarily large amounts of data, and the data can even be spread across multiple files. You can create a datastore for a collection of tabular text files (demonstrated next), a SQL database (Database Toolbox™ required) or a Hadoop® Distributed File System (HDFS™).

Create a datastore for a collection of tabular text files and preview the contents.

```
ds = datastore('airlinesmall.csv');
dsPreview = preview(ds);
dsPreview(:,10:15)
```

## Analyze Big Data in MATLAB Using MapReduce



```
ans =

    FlightNum    TailNum    ActualElapsedTime    CRSElapsedTime    AirTime    ArrDelay
    _________    _______    _________________    ______________    _______    ________

    1503         'NA'       53                   57                'NA'       8
    1550         'NA'       63                   56                'NA'       8
    1589         'NA'       83                   82                'NA'      21
    1655         'NA'       59                   58                'NA'      13
    1702         'NA'       77                   72                'NA'       4
    1729         'NA'       61                   65                'NA'      59
    1763         'NA'       84                   79                'NA'       3
    1800         'NA'      155                  143                'NA'      11

fx >>
```

## Analyze Big Data in MATLAB Using MapReduce

The datastore automatically parses the input data and makes a best guess as to the type of data in each column. In this case, use the 'TreatAsMissing' Name-Value pair argument to have datastore replace the missing values correctly. For numeric variables (such as 'AirTime'), datastore replaces every instance of 'NA' with a NaN value, which is the IEEE arithmetic representation for Not-a-Number.

```
ds = datastore('airlinesmall.csv', 'TreatAsMissing', 'NA');
ds.SelectedFormats{strcmp(ds.SelectedVariableNames, 'TailNum')} = '%s';
ds.SelectedFormats{strcmp(ds.SelectedVariableNames, 'CancellationCode')} =
'%s';
dsPreview = preview(ds);
dsPreview(:,{'AirTime','TaxiIn','TailNum','CancellationCode'})
```

## Analyze Big Data in MATLAB Using MapReduce

```
ans=8×4 table
    AirTime    TaxiIn    TailNum    CancellationCode
    _______    ______    _______    ________________

      NaN       NaN       'NA'            'NA'
      NaN       NaN       'NA'            'NA'
      NaN       NaN       'NA'            'NA'
      NaN       NaN       'NA'            'NA'
      NaN       NaN       'NA'            'NA'
      NaN       NaN       'NA'            'NA'
      NaN       NaN       'NA'            'NA'
      NaN       NaN       'NA'            'NA'
```

## Analyze Big Data in MATLAB Using MapReduce

### Scan for rows of interest

datastore objects contain an internal pointer to keep track of which chunk of data the read function returns next. Use the hasdata and read functions to step through the entire data set, and filter the data set to only the rows of interest. In this case, the rows of interest are flights on United Airlines ("UA") departing from Boston ("BOS").

```
subset = [];

while hasdata(ds)
    t = read(ds);
    t = t(strcmp(t.UniqueCarrier, 'UA') & strcmp(t.Origin, 'BOS'),
:);
    subset = vertcat(subset, t);
end

subset(1:10,[9,10,15:17])
```

## Analyze Big Data in MATLAB Using MapReduce

```
ans=10×5 table
    UniqueCarrier    FlightNum    ArrDelay    DepDelay    Origin
    _____________    _________    ________    ________    ______

        'UA'            121          -9           0       'BOS'
        'UA'           1021          -9          -1       'BOS'
        'UA'            519          15           8       'BOS'
        'UA'            354           9           8       'BOS'
        'UA'            701         -17           0       'BOS'
        'UA'            673          -9          -1       'BOS'
        'UA'             91          -3           2       'BOS'
        'UA'            335          18           4       'BOS'
        'UA'           1429           1          -2       'BOS'
        'UA'             53          52          13       'BOS'
```

## Introduction to mapreduce

MapReduce is an algorithmic technique to "divide and conquer" big data problems. In MATLAB, mapreduce requires three input arguments:

A datastore to read data from

A "mapper" function that is given a subset of the data to operate on. The output of the map function is a partial calculation. mapreduce calls the mapper function one time for each chunk in the datastore, with each call operating independently.

A "reducer" function that is given the aggregate outputs from the mapper function. The reducer function finishes the computation begun by the mapper function, and outputs the final answer.

This is an over-simplification to some extent, since the output of a call to the mapper function can be shuffled and combined in interesting ways before being passed to the reducer function. This will be examined later in this example.

## Use mapreduce to perform a computation

A simple use of mapreduce is to find the longest flight time in the entire airline data set. To do this:

The "mapper" function computes the maximum of each chunk from the datastore.

The "reducer" function then computes the maximum value among all of the maxima computed by the calls to the mapper function.

First, reset the datastore and filter the variables to the one column of interest.

```
reset(ds);
ds.SelectedVariableNames = {'ActualElapsedTime'};
```

Write the mapper function, maxTimeMapper.m. It takes three input arguments:

The input data, which is a table obtained by applying the read function to the datastore.

A collection of configuration and contextual information, info. This can be ignored in most cases, as it is here.

An intermediate data storage object, which records the results of the calculations from the mapper function. Use the add function to add Key/Value pairs to this intermediate output. In this example, the name of the key ('MaxElapsedTime') is arbitrary.

Save the following mapper function (maxTimeMapper.m) in your current folder.

```
type maxTimeMapper

function maxTimeMapper(data, ~, intermKVStore)
% Copyright 2014 The MathWorks, Inc.

maxElaspedTime = max(data{:,:});
add(intermKVStore, 'MaxElaspedTime',maxElaspedTime);
end
```

## MAPPER FUNCTION

Next, write the reducer function. It also takes three input arguments:

A set of input "keys". Keys will be discussed further below, but they can be ignored in some simple problems, as they are here.

An intermediate data input object that mapreduce passes to the reducer function. This data is in the form of Key/Value pairs, and you use the hasnext and getnext functions to iterate through the values for each key.

A final output data storage object. Use the add and addmulti functions to directly add Key/Value pairs to the output.

Save the following reducer function (maxTimeReducer.m) in your current folder.

## Analyze Big Data in MATLAB Using MapReduce

```
type maxTimeReducer

function maxTimeReducer(~, intermValsIter,
outKVStore)
% Copyright 2014 The MathWorks, Inc.

maxElaspedTime = -inf;
while hasnext(intermValsIter)
    maxElaspedTime = max(maxElaspedTime,
getnext(intermValsIter));
end
add(outKVStore, 'MaxElaspedTime',
maxElaspedTime);
end
```

**REDUCER FUNCTION**

## Analyze Big Data in MATLAB Using MapReduce

Once the mapper and reducer functions are written and saved in your current folder, you can call `mapreduce` using the `datastore`, mapper function, and reducer function. If you have Parallel Computing Toolbox (PCT), MATLAB will automatically start a pool and parallelize execution. Use the `readall` function to display the results of the MapReduce algorithm.

```
result = mapreduce(ds, @maxTimeMapper, @maxTimeReducer);
```

```
*      MAPREDUCE PROGRESS      *
********************************
Map   0% Reduce   0%
Map  16% Reduce   0%
Map  32% Reduce   0%
Map  48% Reduce   0%
Map  65% Reduce   0%
Map  81% Reduce   0%
Map  97% Reduce   0%
Map 100% Reduce   0%
Map 100% Reduce 100%
readall(result)
ans=1×2 table
          Key             Value
    ________________    ______

    'MaxElaspedTime'    [1650]
```

## Analyze Big Data in MATLAB Using MapReduce

## Analyze Big Data in MATLAB Using MapReduce

Master en Ingeniería Informática. ULPGC