MATLAB EXPO 2016

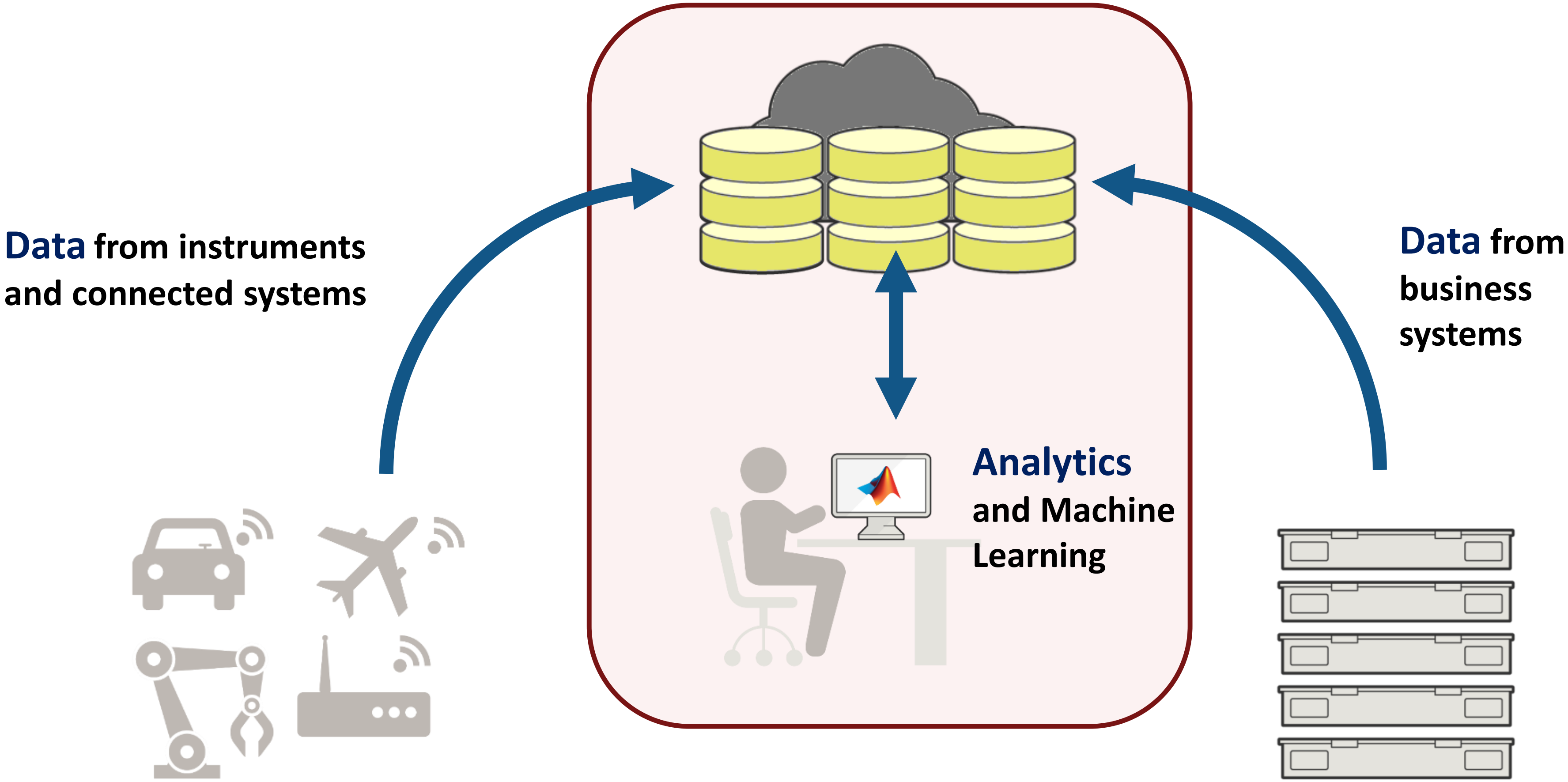
Big Data and Tall Arrays

Ben Tordoff

Agenda

➡ Introduction
- Remote Arrays in MATLAB
- Tall Arrays for Big Data
- Scaling up
- Summary

Architecture of an analytics system



Data from instruments and connected systems

Analytics and Machine Learning

Data from business systems

How big is big?

What does “Big Data” even mean?

“Any collection of data sets so large and complex that it becomes difficult to process using … traditional data processing applications.”

(Wikipedia)

“Any collection of data sets so large that it becomes difficult to process using traditional MATLAB functions, which assume all of the data is in memory.”

(MATLAB)

How big is big?

In 1085 William 1st commissioned a survey of England

- ~2 million words and figures collected over two years
- too big to handle in one piece
- collected and summarized in regional pieces
- used to generate revenue (tax), but most of the data then sat unused

The Large Hadron Collider reached peak performance on 29 June 2016

- 2076 bunches of 120 billion protons currently circulating in each direction
- ~1.6×10^{14} collisions per week, >30 petabytes of data per year
- too big to even store in one place
- used to explore interesting science, but taking researchers a long time to get through

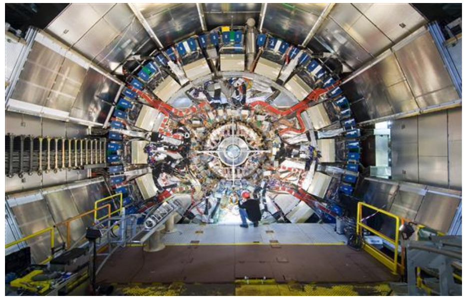
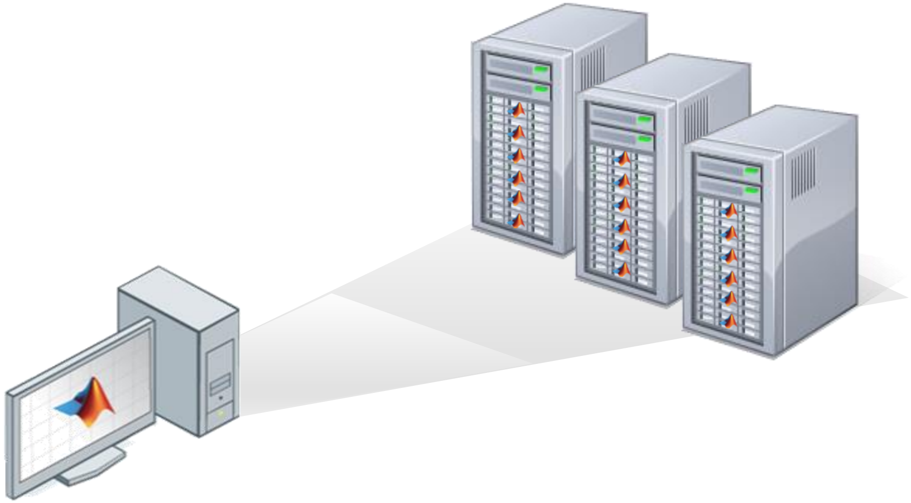


Image courtesy of CERN.
Copyright 2011 CERN.

How big is big?

Most of our data lies somewhere in between the extremes

- >10GB might be too much for one laptop / desktop (“inconveniently large”)



Big problems

So what's the big problem?

- Standard tools won't work
- Getting the data is hard; processing it is even harder
- Need to learn new tools and new coding styles
- Have to rewrite algorithms, often at a lower level of abstraction

We want to let you:

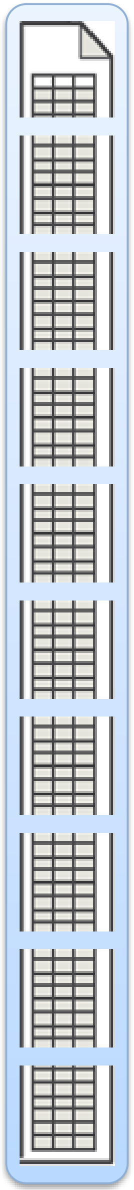
- Prototype algorithms quickly using small data
- Scale up to huge data-sets running on large clusters
- **Use the same MATLAB code for both**

New solution for R2016b: tall arrays

Quick overview (detail later!):

- Treat data in multiple files as one large table/array
- Write normal array / table code
- Behind the scenes operate on pieces

`tall array`

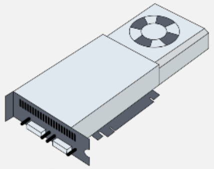
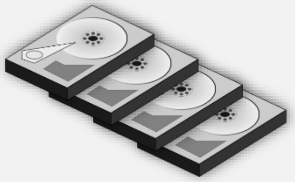


Agenda

- Introduction
- ➡ Remote Arrays in MATLAB
- Tall Arrays for Big Data
- Scaling up
- Summary

Remote arrays in MATLAB

MATLAB provides array types for data that is not in “normal” memory

distributed array (since R2006b)		Data lives in the combined memory of a cluster of computers
gpuArray (since R2010b)		Data lives in the memory of the GPU card
tall array (since R2016b)		Data lives on disk, maybe spread across many disks (distributed file-system)

Remote arrays in MATLAB

Rule: take the calculation to where the data is

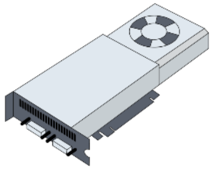
Normal array – calculation happens in main memory:

```
x = rand(...)

x_norm = (x - mean(x)) ./ std(x)
```

Remote arrays in MATLAB

Rule: take the calculation to where the data is

gpuArray – all calculation happens on the GPU:

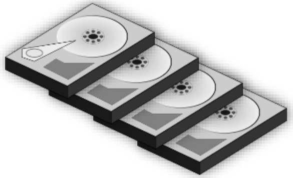


```
x = gpuArray(...)

x_norm = (x - mean(x)) ./ std(x)
```

distributed – calculation is spread across the cluster:

```
x = distributed(...)

x_norm = (x - mean(x)) ./ std(x)
```

tall – calculation is performed by stepping through files:



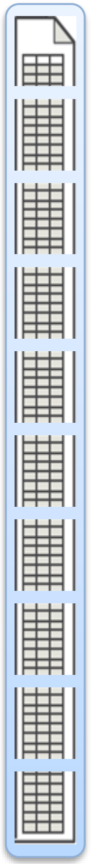
```
x = tall(...)

x_norm = (x - mean(x)) ./ std(x)
```

Agenda

- Introduction
- Remote Arrays in MATLAB
- ➡ Tall Arrays for Big Data
- Scaling up
- Summary

Tall arrays (new R2016b)

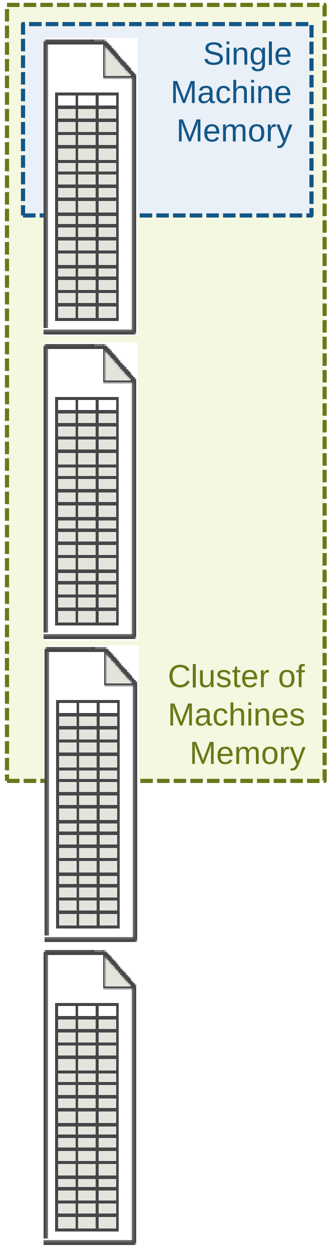
- MATLAB data-type for data that doesn’t fit into memory
- Ideal for lots of observations, few variables (hence “tall”)
- Looks like a normal MATLAB array
  - Supports numeric types, tables, datetimes, categoricals, strings, etc…
  - Basic maths, stats, indexing, etc.
  - **Statistics and Machine Learning Toolbox** support (clustering, classification, etc.)



Tall arrays (new R2016b)

- Data is in one or more files
- Typically tabular data
- Files stacked vertically
- Data doesn’t fit into memory

  (even cluster memory)



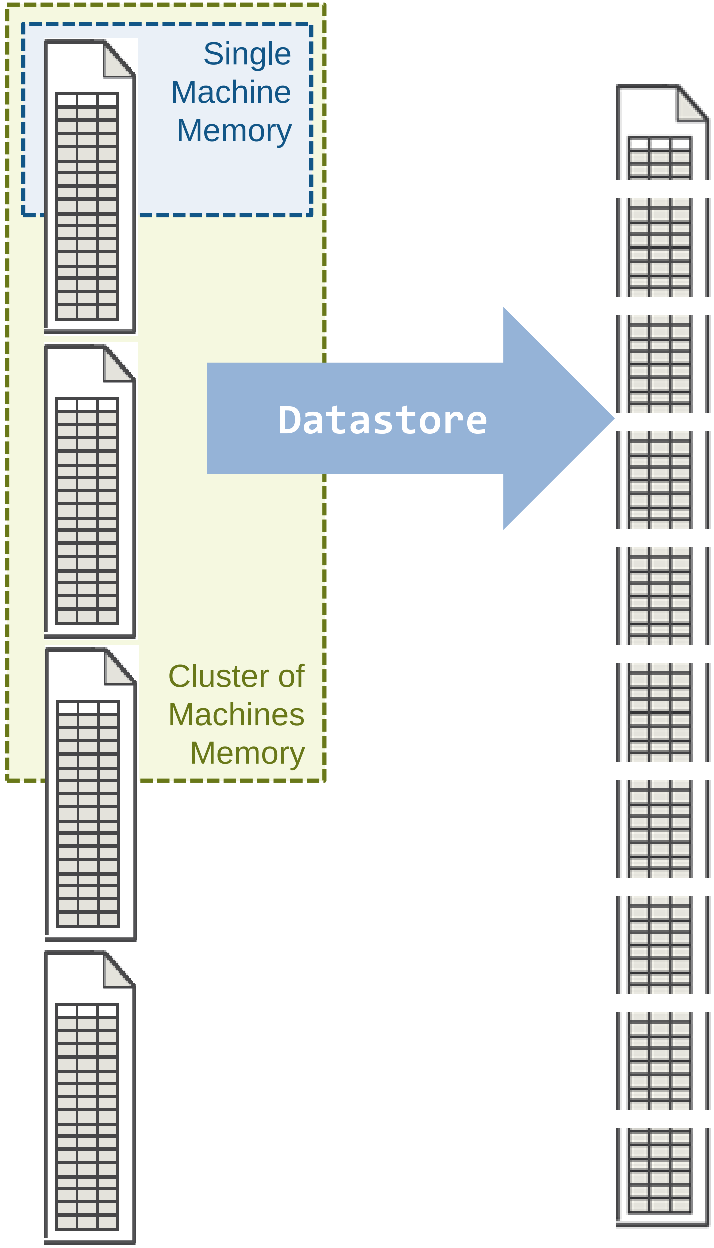
Tall arrays (new R2016b)

- Use datastore to define file-list

```
ds = datastore('*.csv')
```

- Allows access to small pieces of data that fit in memory.

```
while hasdata(ds)
   piece = read(ds);
   % Process piece
end
```



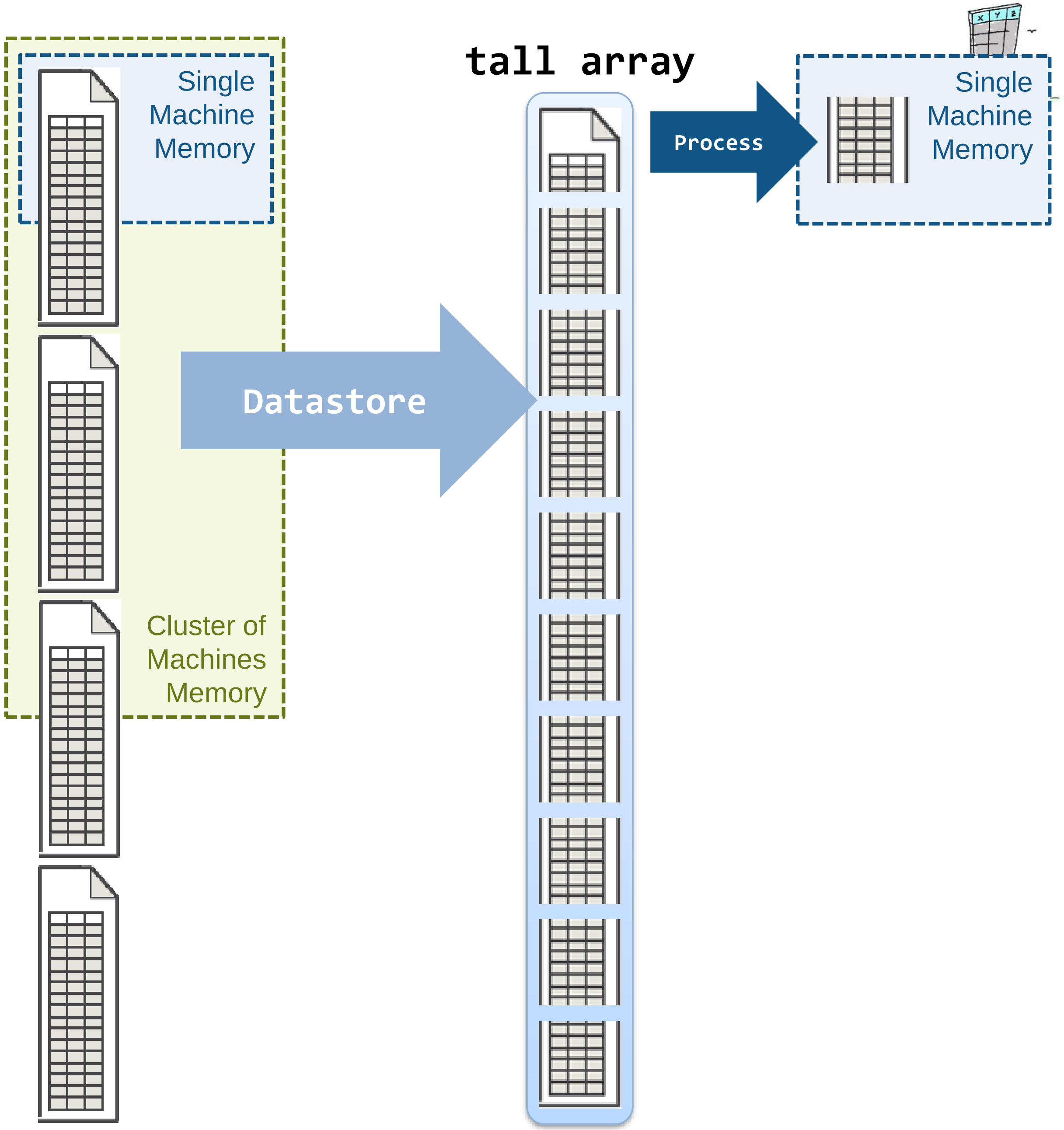
Tall arrays (new R2016b)

- Create tall table from datastore

```
ds = datastore('*.csv')
tt = tall(ds)
```

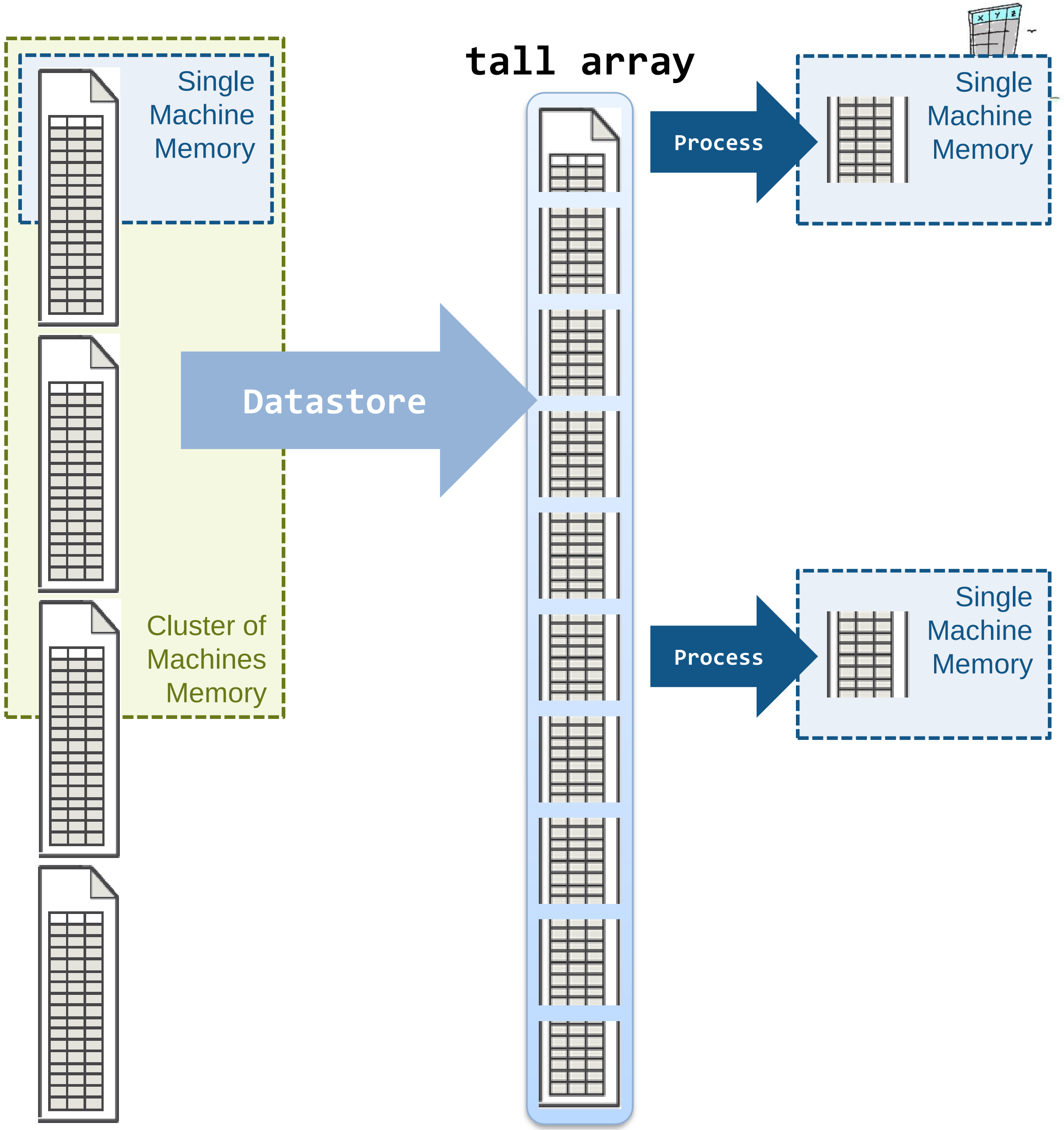
- Operate on whole tall table just like ordinary table

```
summary(tt)

max(tt.EndTime - tt.StartTime)
```



Tall arrays (new R2016b)

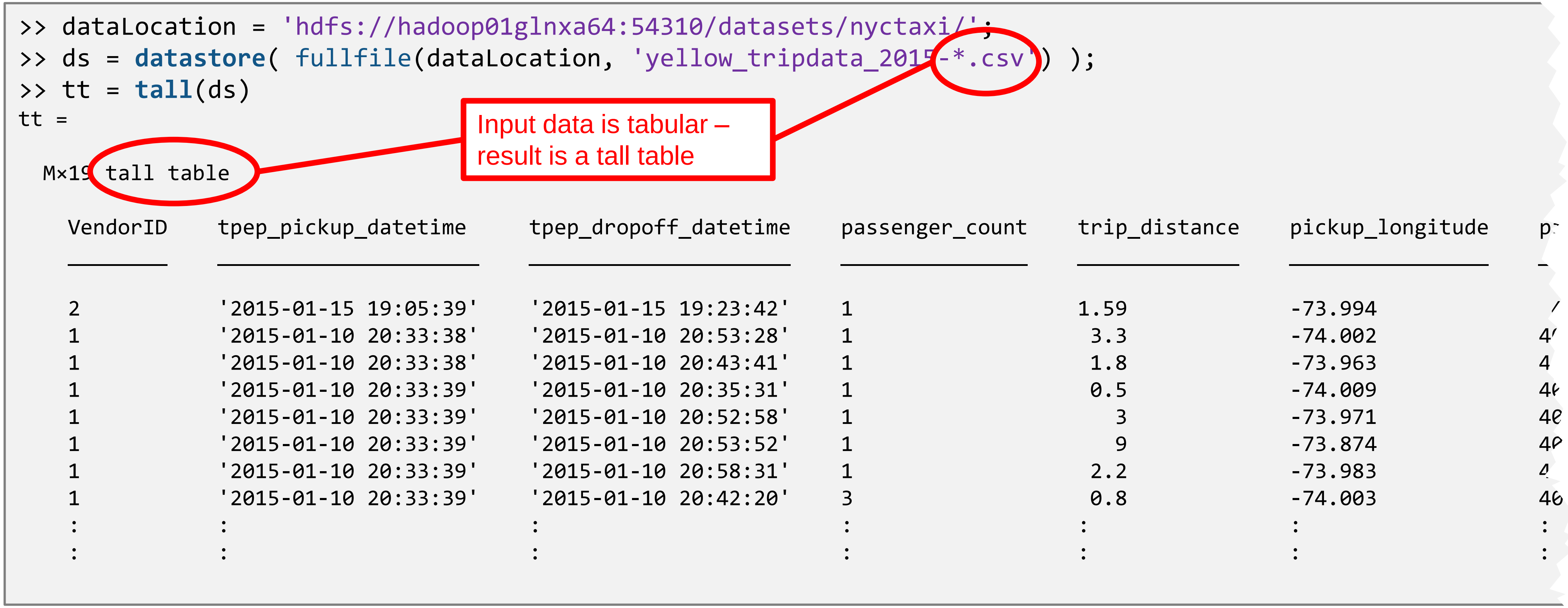
- With Parallel Computing Toolbox, process several pieces at once



Tall arrays (new R2016b)

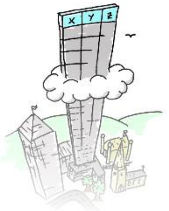
Example

New York taxi fares (150,000,000 rows (~25GB) per year)



```
>> dataLocation = 'hdfs://hadoop01glnxa64:54310/datasets/nyctaxi/';
>> ds = datastore( fullfile(dataLocation, 'yellow_tripdata_2015-*.csv') );
>> tt = tall(ds)
tt =

  M×19 tall table

    VendorID    tpep_pickup_datetime     tpep_dropoff_datetime    passenger_count    trip_distance    pickup_longitude    p
    ________    _____________________    _____________________    _______________    _____________    ________________    _

    2           '2015-01-15 19:05:39'    '2015-01-15 19:23:42'    1                  1.59             -73.994              /
    1           '2015-01-10 20:33:38'    '2015-01-10 20:53:28'    1                   3.3             -74.002             4(
    1           '2015-01-10 20:33:38'    '2015-01-10 20:43:41'    1                   1.8             -73.963             4
    1           '2015-01-10 20:33:39'    '2015-01-10 20:35:31'    1                   0.5             -74.009             4(
    1           '2015-01-10 20:33:39'    '2015-01-10 20:52:58'    1                     3             -73.971             40
    1           '2015-01-10 20:33:39'    '2015-01-10 20:53:52'    1                     9             -73.874             40
    1           '2015-01-10 20:33:39'    '2015-01-10 20:58:31'    1                   2.2             -73.983             4
    1           '2015-01-10 20:33:39'    '2015-01-10 20:42:20'    3                   0.8             -74.003             46
    :           :                        :                        :                  :                :                   :
    :           :                        :                        :                  :                :                   :
```

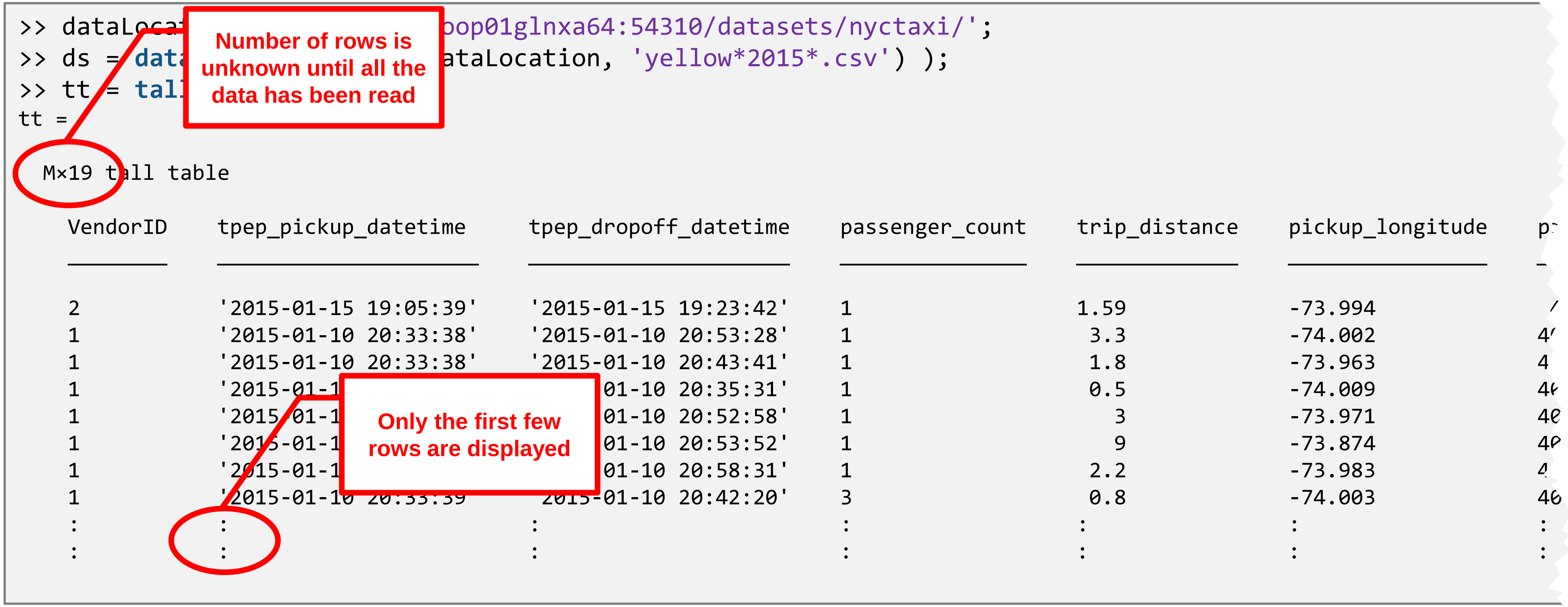
Input data is tabular – result is a tall table



Tall arrays (new R2016b)

Example

New York taxi fares (150,000,000 rows (~25GB) per year)



```
>> dataLoca                        oop01glnxa64:54310/datasets/nyctaxi/';
>> ds = data                       ataLocation, 'yellow*2015*.csv') );
>> tt = tal
tt =

  M×19 tall table

    VendorID    tpep_pickup_datetime     tpep_dropoff_datetime    passenger_count    trip_distance    pickup_longitude    p
    ________    _____________________    _____________________    _______________    _____________    ________________    _

    2           '2015-01-15 19:05:39'    '2015-01-15 19:23:42'    1                  1.59             -73.994
    1           '2015-01-10 20:33:38'    '2015-01-10 20:53:28'    1                   3.3             -74.002             4
    1           '2015-01-10 20:33:38'    '2015-01-10 20:43:41'    1                   1.8             -73.963             4
    1           '2015-01-1                 01-10 20:35:31'        1                   0.5             -74.009             4
    1           '2015-01-1                 01-10 20:52:58'        1                     3             -73.971             4
    1           '2015-01-1                 01-10 20:53:52'        1                     9             -73.874             4
    1           '2015-01-1                 01-10 20:58:31'        1                   2.2             -73.983             4
    1           '2015-01-10 20:33:39      2015-01-10 20:42:20'    3                   0.8             -74.003             46
    :           :                        :                        :                  :                :                   :
    :           :                        :                        :                  :                :                   :
```

Number of rows is unknown until all the data has been read

Only the first few rows are displayed

Tall arrays (new R2016b)

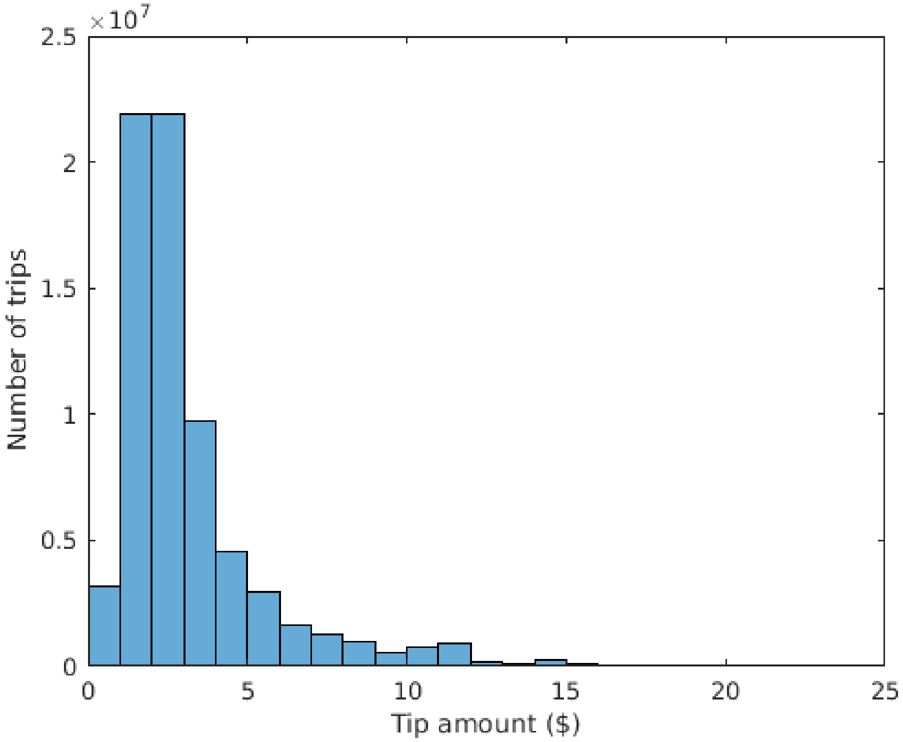
Example

Once created, can process much like an ordinary table

```
% Remove some bad data
tt.trip_minutes = minutes(tt.tpep_dropoff_datetime - tt.tpep_pickup_datetime);
tt.speed_mph = tt.trip_distance ./ (tt.trip_minutes ./ 60);
ignore = tt.trip_minutes <= 1 | ...     % really short
    tt.trip_minutes >= 60 * 12 | ...    % unfeasibly long
    tt.trip_distance <= 1 | ...         % really short
    tt.trip_distance >= 12 * 55 | ...   % unfeasibly far
    tt.speed_mph > 55 | ...             % unfeasibly fast
    tt.total_amount < 0 | ...           % negative fares?!
    tt.total_amount > 10000;            % unfeasibly large fares
tt(ignore, :) = [];

% Credit card payments have the most accurate tip data
keep = tt.payment_type == {'Credit card'};
tt = tt(keep,:);

% Show tip distribu
histogram( tt.tip_a
```

Data only read once, despite 21 operations

```
Evaluating tall expression using the Parallel Pool 'local':
- Pass 1 of 1: Completed in 4.9667 min
Evaluation completed in 5 min
```



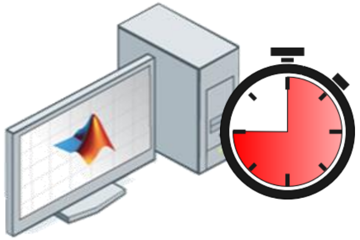
Agenda

- Introduction
- Remote Arrays in MATLAB
- Tall Arrays for Big Data
- ➡ Scaling up
- Summary

Scaling up

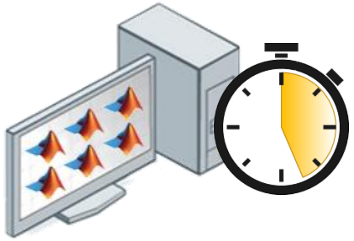
If you just have **MATLAB**:

- Run through each ‘chunk’ of data one by one



If you also have **Parallel Computing Toolbox**:

- Use all local cores to process several ‘chunks’ at once



If you also have a cluster with **MATLAB Distributed Computing Server (MDCS)**:

- Use the whole cluster to process many ‘chunks’ at once

Scaling up

Working with clusters from MATLAB desktop:

- General purpose MATLAB cluster
  - Can co-exist with other MATLAB workloads (parfor, parfeval, spmd, jobs and tasks, distributed arrays, …)
  - Uses local memory and file caches on workers for efficiency

- Spark-enabled Hadoop clusters
  - Data in HDFS
  - Calculation is scheduled to be near data
  - Uses Spark’s built-in memory and disk caching

Agenda

- Introduction
- Remote Arrays in MATLAB
- Tall Arrays for Big Data
- Scaling up
- ➡ Summary

Summary

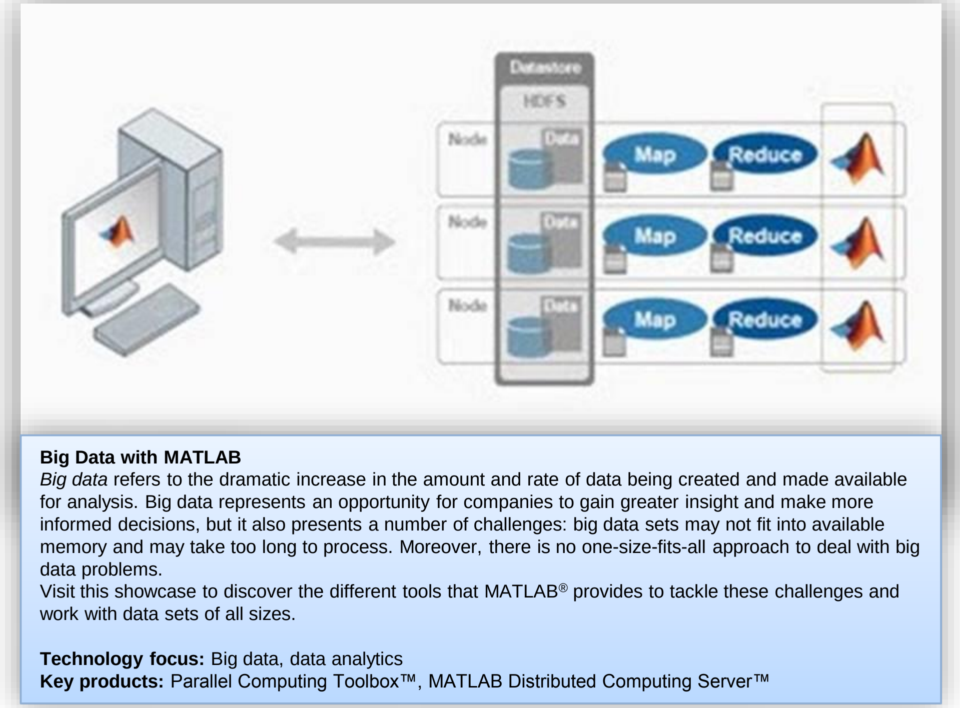
tall arrays

New!

- MATLAB data-type for data that doesn't fit into memory
- Minimizes I/O by deferring calculation
- Looks like a normal MATLAB array or table
  - Supports numeric types, tables, datetimes, categoricals, strings, etc…
  - Basic maths, stats, indexing, etc.
  - **Statistics and Machine Learning Toolbox** support (clustering, classification, etc.)
- Scale up using clusters, including Spark

Live demo

- Come and find us at the “BigData with MATLAB” demo station
  - more details
  - live demos
  - Meet the developers behind the features



Big Data with MATLAB
Big data refers to the dramatic increase in the amount and rate of data being created and made available for analysis. Big data represents an opportunity for companies to gain greater insight and make more informed decisions, but it also presents a number of challenges: big data sets may not fit into available memory and may take too long to process. Moreover, there is no one-size-fits-all approach to deal with big data problems.
Visit this showcase to discover the different tools that MATLAB® provides to tackle these challenges and work with data sets of all sizes.

Technology focus: Big data, data analytics
Key products: Parallel Computing Toolbox™, MATLAB Distributed Computing Server™

Questions