Solution to the exercises of Session 3

Aaron Ponti

- **Section 1.1.2**

```
>> load fisheriris
>> n = nominal( species );
>> sort( n )
??? Undefined function or method 'sort' for input arguments
of type 'nominal'.
```

- **Section 1.2.1 / 1**

Put this into a script:

```
% Import data from iris.xls or iris.csv
importedData = importdata( 'iris.xls' ); % or 'iris.csv'

% Extract the species names and the variable names
varNames = importedData.textdata( 1, 2 : end );
species  = importedData.textdata( 2 : end , 1 );

% Create a nominal array called 'species' to
% label the measurements
n = { nominal( species ), 'species' };

% The variable names associated to the measurement
% matrix data.numeric are 'SL', 'SW', 'PL', 'PW'
% (for Sepal and Petal Length and Width)
m = { importedData.data, varNames{ 1 }, varNames{ 2 }, ...
    varNames{ 3 }, varNames{ 4 } };

% Create the observation names
NumObs = size( importedData.data, 1 );
NameObs = strcat( { 'Obs' }, num2str( ( 1 : NumObs )', '%d' ) );

% Create the dataset itself
iris = dataset( n, m, 'ObsNames', NameObs );
```

The dataset class can import Excel files automatically when used as follows:

```
>> iris = dataset( 'XLSFile', 'iris.xls', ...
    'ReadVarNames', true, 'ReadObsNames', false );
```

but this will work only for files with a simple and fixed structure: see *help dataset* for more information. More granularity can be achieved using the function *xlsread*.

For text files, the *dataset* function can be called as follows:

```
>> iris = dataset( 'File', 'iris.csv', 'FORMAT', format, ...
    'ReadVarNames', true, 'ReadObsNames', false );
```

where *format* defines the way data has to be imported from each row of the file into the dataset object; *dataset* uses the function *textscan* to read the file: see *help textscan* for more information on how to set the format. Alternatively, the function *textread* can be used.

- **Section 1.2.1 / 2**

```
>> iris = set( iris, 'Description', 'Fisher"s iris data (1936)' );
>> get( iris )

    Description: 'Fisher's iris data (1936)'
          Units: {}
       DimNames: {'Observations'  'Variables'}
       UserData: []
       ObsNames: {150x1 cell}
       VarNames: {'species'  'SL'  'SW'  'PL'  'PW'}
```

- **Section 2.3 / 1**

```
>> help normrnd

NORMRND Random arrays from the normal distribution.
    R = NORMRND(MU,SIGMA) returns an   array of random numbers chosen from a
    normal distribution with mean MU and standard deviation SIGMA.  The size
    of R is the common size of MU and SIGMA if both are arrays.  If either
    parameter is a scalar, the size of R is the size of the other
    parameter.

    R = NORMRND(MU,SIGMA,M,N,...) or R = NORMRND(MU,SIGMA,[M,N,...])
    returns an M-by-N-by-... array.

    See also normcdf, normfit, norminv, normlike, normpdf, normstat,
    random, randn.

    Reference page in Help browser
       doc normrnd
```

- **Section 2.3 / 2**

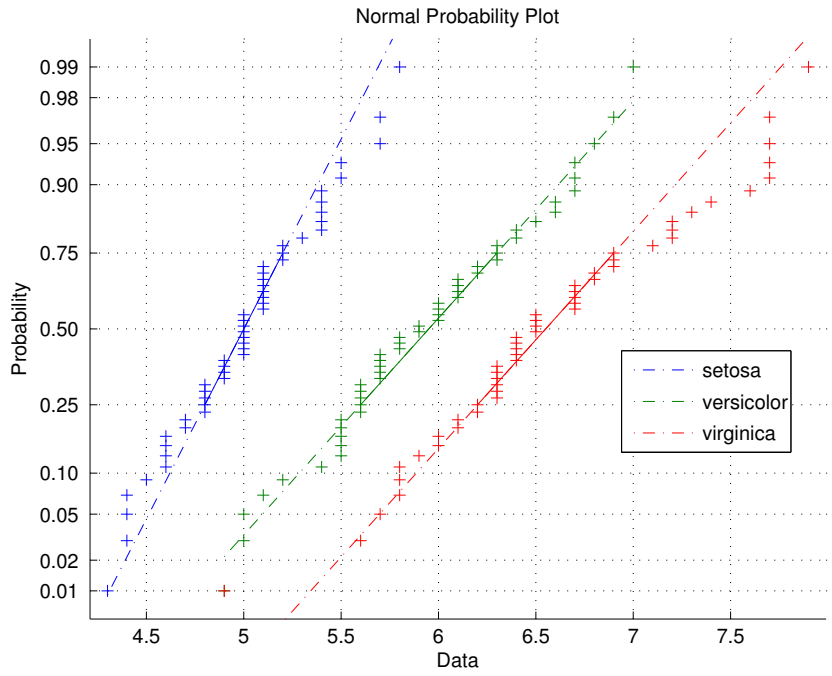

Figure 1: Normal probability plot for sepal lengths of the three Iris species *setosa*, *versicolor* and *virginica*.

```
>> normplot( [ ...
    iris.SL( iris.species == 'setosa' ), ...
    iris.SL( iris.species == 'versicolor' ), ...
    iris.SL( iris.species == 'virginica' ) ] );
>> legend( 'setosa', 'versicolor', 'virginica', 'Location', 'Best' );
```