

159 Lecture 14: Importing and Exporting Data

Note: **To type a command** in a *Mathematica* notebook, use the mouse to move the cursor until it is horizontal, click the left mouse button, and type the command. **To enter a command**, use the mouse to move the cursor until it is vertical and over the command line, then click the left mouse button, and finally press and hold the `Shift` key followed by `Enter` key or use the `Enter` key on the smaller number keypad (lower right-hand corner of keyboard). **To expand cells**, double click on the cell bracket (blue vertical bar to the right) with the downward pointing arrow.

Import and Export

The commands **Import** and **Export** can be used to either bring data from a file into a *Mathematica* notebook or put data from a *Mathematica* notebook into a file.

■ Import

Import ["*file.ext*"] imports data from a file, assuming that it is in the format indicated by the file extension *ext*, and converts it to a *Mathematica* expression.

Import ["*file*" , "*format*"] imports data in the specified format from a file.

Import can handle numerical and textual data, graphics, sounds, material from notebooks, and general expressions in various formats.

For supported formats, see "More Information" in the help file or use the command **\$ImportFormats**.

```
In[1]:= $ImportFormats
```

```
Out[1]= {3DS, ACO, AIFF, ApacheLog, AU, AVI, Base64, Binary, Bit, BMP, Byte, BYU, BZIP2, CDED, CDF, Character16, Character8, Complex128, Complex256, Complex64, CSV, CUR, DBF, DICOM, DIF, Directory, DXF, EDF, ExpressionML, FASTA, FITS, FLAC, GenBank, GeoTIFF, GIF, Graph6, GTOPO30, GZIP, HarwellBoeing, HDF, HDF5, HTML, ICO, Integer128, Integer16, Integer24, Integer32, Integer64, Integer8, JPEG, JPEG2000, JVX, LaTeX, List, LWO, MAT, MathML, MBOX, MDB, MGF, MMCIF, MOL, MOL2, MPS, MTP, MTX, MX, NB, NetCDF, NOFF, OBJ, ODS, OFF, Package, PBM, PCX, PDB, PDF, PGM, PLY, PNG, PNM, PPM, PXR, QuickTime, RawBitmap, Real128, Real32, Real64, RIB, RSS, RTF, SCT, SDF, SDTS, SDTSD, SHP, SMILES, SND, SP3, Sparse6, STL, String, SXC, Table, TAR, TerminatedString, Text, TGA, TIFF, TIGER, TSV, UnsignedInteger128, UnsignedInteger16, UnsignedInteger24, UnsignedInteger32, UnsignedInteger64, UnsignedInteger8, USGSDEM, UUE, VCF, WAV, Wave64, WDX, XBM, XHTML, XHTMLMathML, XLS, XML, XPORT, XYZ, ZIP}
```

■ Export

Export["*file.ext*", *expr*] exports data to a file, converting it to a format corresponding to the file extension *ext*.

Export["*file*", *expr*, "*format*"] exports data to a file, converting it to the specified format.

Export can handle numerical and textual data, graphics, sounds, material from notebooks, and general expressions in various formats.

For supported formats, see "More Information" in the help file or use the command **\$ExportFormats**.

```
In[2]:= $ExportFormats
```

```
Out[2]= {3DS, ACO, AIFF, AU, AVI, Base64, Binary, Bit, BMP, Byte, BYU, BZIP2, CDF, Character16,
Character8, Complex128, Complex256, Complex64, CSV, DICOM, DIF, DXF, EMF, EPS,
ExpressionML, FASTA, FITS, FLAC, FLV, GIF, Graph6, GZIP, HarwellBoeing, HDF, HDF5, HTML,
Integer128, Integer16, Integer24, Integer32, Integer64, Integer8, JPEG, JPEG2000,
JVX, List, LWO, MAT, MathML, Maya, MGF, MIDI, MOL, MOL2, MTX, MX, NB, NetCDF, NOFF, OBJ,
OFF, Package, PBM, PCX, PDB, PDF, PGM, PLY, PNG, PNM, POV, PPM, PXR, RawBitmap, Real128,
Real32, Real64, RIB, RTF, SCT, SDF, SND, Sparse6, STL, String, SVG, SWF, Table, TAR,
TerminatedString, TeX, Text, TGA, TIFF, TSV, UnsignedInteger128, UnsignedInteger16,
UnsignedInteger24, UnsignedInteger32, UnsignedInteger64, UnsignedInteger8, UUE,
VRML, WAV, Wave64, WDX, WMF, X3D, XBM, XHTML, XHTMLMathML, XLS, XML, XYZ, ZIP, ZPR}
```

■ Example 1

Let's start by creating a set of data and exporting it to a file!

```
In[3]:= toaddata =
```

```
    {{1939, 32800}, {1944, 55800}, {1949, 73600}, {1954, 138000},
     {1959, 202000}, {1964, 257000}, {1969, 301000}, {1974, 584000}};
```

```
In[4]:= Export["toads1.txt", toaddata]
```

```
Out[4]= toads1.txt
```

To import the data, we use the **Import** command!

```
In[5]:= Import["toads1.txt"]
```

```
Out[5]= {1939, 32800}
         {1944, 55800}
         {1949, 73600}
         {1954, 138000}
         {1959, 202000}
         {1964, 257000}
         {1969, 301000}
         {1974, 584000}
```

Notice that the data doesn't appear to be in a list format, as it was when we exported it. To see how *Mathematica* "sees" the data, we can use the **Head** command.

```
In[6]:= Head [%]
```

```
Out[6]= String
```

Mathematica considers the data from toads1.txt to be a string of text, not a list, as we originally started with for our toad data!

```
In[7]:= Head [toaddata]
```

```
Out[7]= List
```

To import the data as a list, specify this in the **"format"** part of an **Import** command.

```
In[8]:= Import ["toads1.txt", "List"]
```

```
Out[8]= {{1939, 32800}, {1944, 55800}, {1949, 73600}, {1954, 138000},
         {1959, 202000}, {1964, 257000}, {1969, 301000}, {1974, 584000}}
```

```
In[9]:= Head [%]
```

```
Out[9]= List
```

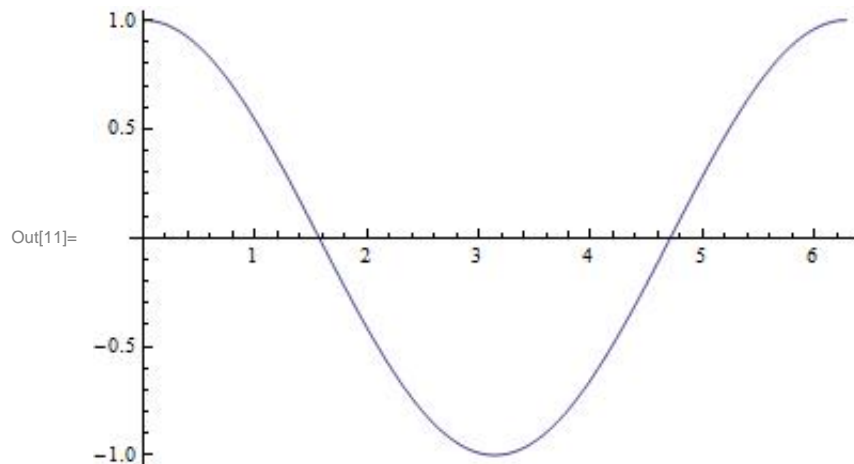
■ Example 2

We can also import or export graphics!

```
In[10]:= Export ["plot1.jpg", Plot[Cos[x], {x, 0, 2 π}]]
```

```
Out[10]= plot1.jpg
```

```
In[11]:= Import ["plot1.jpg"]
```



```
In[12]:= Head [%]
```

```
Out[12]= Image
```

■ Where is this stuff going?

The files that are imported or exported have to be stored somewhere in the computer's memory. It is a good idea to know where these files are, especially since we may want to import or export data to a file in a specific location.

Use the Windows Explorer to find the files we just exported, **toads1.txt** and **plot1.jpg**! In this case, the files are most likely located in a folder such as C:\Documents and Settings\Default User.

To see all the places *Mathematica* can find files, by default, use **\$Path**.

```
In[13]:= $Path
Out[13]= {C:\Program Files\Wolfram Research\Mathematica\7.0\SystemFiles\Links,
C:\Users\User 1\AppData\Roaming\Mathematica\Kernel,
C:\Users\User 1\AppData\Roaming\Mathematica\Autoload,
C:\Users\User 1\AppData\Roaming\Mathematica\Applications,
C:\ProgramData\Mathematica\Kernel, C:\ProgramData\Mathematica\Autoload,
C:\ProgramData\Mathematica\Applications, ., C:\Users\User 1,
C:\Program Files\Wolfram Research\Mathematica\7.0\AddOns\Packages,
C:\Program Files\Wolfram Research\Mathematica\7.0\AddOns\LegacyPackages,
C:\Program Files\Wolfram Research\Mathematica\7.0\SystemFiles\Autoload,
C:\Program Files\Wolfram Research\Mathematica\7.0\AddOns\Autoload,
C:\Program Files\Wolfram Research\Mathematica\7.0\AddOns\Applications,
C:\Program Files\Wolfram Research\Mathematica\7.0\AddOns\ExtraPackages,
C:\Program Files\Wolfram Research\Mathematica\7.0\SystemFiles\Kernel\Packages,
C:\Program Files\Wolfram Research\Mathematica\7.0\Documentation\English\System}
```

■ Example 3

Here's how to change location for where the files are imported from or exported to. In each of the first three commands below, the files are being saved in the Z: directory. In each case, the file is saved in a different format - CSV (comma separated value format), DAT (*Mathematica's* data format), and XLS (Excel spreadsheet format - prior to Excel 2007). **NOTE: If the following commands in this example do not work on your computer, try changing the "z:" in the commands to a letter corresponding to a flash drive or ilocker directory.**

Notice that for a file name typed in as a text string (i.e. in quotes), single backslashes (\) are represented with double backslashes (\\).

After entering the export commands, locate each of these files with the Windows Explorer. Open the first two with Notepad and the third with Excel - what do you notice?

```
In[14]:= Export["z:\\toads2.csv", toaddata]
Out[14]= z:\toads2.csv
```

```
In[15]= Export["z:\\toads3.dat", toaddata]
```

```
Out[15]= z:\toads3.dat
```

```
In[16]= Export["z:\\toads4.xls", toaddata]
```

```
Out[16]= z:\toads4.xls
```

Import the data from each of these files and see what you get.

```
In[17]= Import["z:\\toads2.csv"]
```

```
Out[17]= {{1939, 32 800}, {1944, 55 800}, {1949, 73 600}, {1954, 138 000},
          {1959, 202 000}, {1964, 257 000}, {1969, 301 000}, {1974, 584 000}}
```

```
In[18]= Head[%]
```

```
Out[18]= List
```

```
In[19]= Import["z:\\toads3.dat"]
```

```
Out[19]= {{1939, 32 800}, {1944, 55 800}, {1949, 73 600}, {1954, 138 000},
          {1959, 202 000}, {1964, 257 000}, {1969, 301 000}, {1974, 584 000}}
```

```
In[20]= Head[%]
```

```
Out[20]= List
```

```
In[21]= Import["z:\\toads4.xls"]
```

```
Out[21]= {{{1939., 32 800.}, {1944., 55 800.}, {1949., 73 600.}, {1954., 138 000.},
          {1959., 202 000.}, {1964., 257 000.}, {1969., 301 000.}, {1974., 584 000.}}}
```

```
In[22]= Head[%]
```

```
Out[22]= List
```

Put and Get

The commands **Put** (>>) and **Get** (<<) can also be used to put data into a file or get data from a file. These commands work like "Write" or "Read" commands in programming languages such as C or C++.

■ Put

$expr \gg filename$ writes $expr$ to a file.

$expr \gg "filename"$ is equivalent to $expr \gg filename$.

`Put[$expr_1, expr_2, \dots, "filename"$]` writes a sequence of expressions $expr_i$ to a file.

Put starts writing output at the beginning of the file. It deletes whatever was previously in the file.

Put inserts a newline (line feed) at the end of its output.

The command `FilePrint["name"]` shows the contents of the file $name$.

■ Example 4

```
In[23]:= Expand[(x + y)^4]
```

```
Out[23]= x^4 + 4 x^3 y + 6 x^2 y^2 + 4 x y^3 + y^4
```

```
In[24]:= % >> z:\tempfile
```

```
In[25]:= FilePrint["z:\\tempfile"]
```

```
x^4 + 4*x^3*y + 6*x^2*y^2 + 4*x*y^3 + y^4
```

```
In[26]:= 57 >> z:\tempfile
```

```
In[27]:= FilePrint["z:\\tempfile"]
```

```
57
```

```
In[28]:= A = Import["z:\\toads3.dat"];
```

```
In[29]:= Put[Expand[(x + y)^4], 57,  $\pi + \text{Sin}[\pi/2]$ ,  $\frac{78}{23}$ , A, "z:\\tempfile"]
```

```
In[30]:= FilePrint["z:\\tempfile"]
```

```
x^4 + 4*x^3*y + 6*x^2*y^2 + 4*x*y^3 + y^4
```

```
57
```

```
1 + Pi
```

```
78/23
```

```
{{1939, 32800}, {1944, 55800}, {1949, 73600}, {1954, 138000}, {1959, 202000},  
{1964, 257000}, {1969, 301000}, {1974, 584000}}
```

■ PutAppend

`expr >>> filename` appends `expr` to a file.

`expr >>> "filename"` is equivalent to `expr >>> filename`.

`PutAppend[expr1, expr2, ... , "filename"]` appends a sequence of expressions `expri` to a file.

Note that unlike `Put`, the contents of the file are NOT overwritten!

■ Example 5

```
In[31]:= D[ $\frac{\sqrt{x+1}}{\text{ArcSec}[x^2-x+1]}$ , {x, 2}] >>> z:\tempfile
```

```
In[32]:= FilePrint["z:\\tempfile"]
```

```
x^4 + 4*x^3*y + 6*x^2*y^2 + 4*x*y^3 + y^4
57
1 + Pi
78/23
{{1939, 32800}, {1944, 55800}, {1949, 73600}, {1954, 138000}, {1959, 202000},
{1964, 257000}, {1969, 301000}, {1974, 584000}}
Sqrt[1 + x]*((2*(-1 + 2*x)^2)/((1 - x + x^2)^4*(1 - (1 - x + x^2)^(-2)))*
ArcSec[1 - x + x^2]^3) + (-1 + 2*x)^2/((1 - x + x^2)^5*
(1 - (1 - x + x^2)^(-2)))^(3/2)*ArcSec[1 - x + x^2]^2) +
(2*(-1 + 2*x)^2)/((1 - x + x^2)^3*Sqrt[1 - (1 - x + x^2)^(-2)])*
ArcSec[1 - x + x^2]^2) - 2/((1 - x + x^2)^2*Sqrt[1 - (1 - x + x^2)^(-2)])*
ArcSec[1 - x + x^2]^2) - (-1 + 2*x)/(Sqrt[1 + x]*(1 - x + x^2)^2*
Sqrt[1 - (1 - x + x^2)^(-2)])*ArcSec[1 - x + x^2]^2) -
1/(4*(1 + x)^(3/2)*ArcSec[1 - x + x^2])
```

■ Get

`<<name` reads in a file, evaluating each expression in it, and returning the last one.

`<<"name"` is equivalent to `<<name`.

`Get` by default successively searches for files in the directories specified by the elements of `$Path`.

`Get[name, Path->{"dir1", "dir2", ... }]` successively searches for files in each of the `diri`.

The `Get` command can also be used to load in packages that contain extra Mathematica commands that may not be used as frequently.

The syntax is `<<PackageName`` where `PackageName` is a Mathematica package file that defines the commands to be used. Note that an equivalent way to do this is with the `Needs` command, i.e. enter `Needs["PackageName`"]`.

Example 6

In[33]:= << z:\tempfile

$$\text{Out[33]} = \sqrt{1+x} \left(\frac{2(-1+2x)^2}{(1-x+x^2)^4 \left(1 - \frac{1}{(1-x+x^2)^2}\right) \text{ArcSec}[1-x+x^2]^3} + \frac{(-1+2x)^2}{(1-x+x^2)^5 \left(1 - \frac{1}{(1-x+x^2)^2}\right)^{3/2} \text{ArcSec}[1-x+x^2]^2} + \frac{2(-1+2x)^2}{(1-x+x^2)^3 \sqrt{1 - \frac{1}{(1-x+x^2)^2}} \text{ArcSec}[1-x+x^2]^2} - \frac{2}{(1-x+x^2)^2 \sqrt{1 - \frac{1}{(1-x+x^2)^2}} \text{ArcSec}[1-x+x^2]^2} \right) - \frac{-1+2x}{\sqrt{1+x} (1-x+x^2)^2 \sqrt{1 - \frac{1}{(1-x+x^2)^2}} \text{ArcSec}[1-x+x^2]^2} - \frac{1}{4(1+x)^{3/2} \text{ArcSec}[1-x+x^2]}$$

Example 7

Here is a way to make a histogram and box-and-whisker plot of the estimated fish populations from a capture-recapture sampling run. First import in the data from a CSV file created with Excel and saved online. Use the **Flatten** command to get rid of the inner braces in the nested list that results.

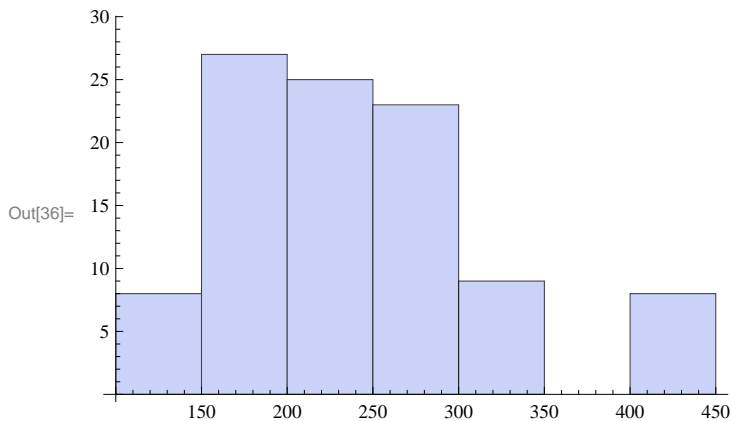
In[34]:= **fishdata =****Import["http://www.bsu.edu/web/mkarls/FishPopulationData.csv"]**

```
Out[34]= {{220}, {188}, {165}, {188}, {264}, {331}, {264}, {220}, {165}, {264}, {188}, {188}, {264},
{264}, {188}, {441}, {441}, {441}, {220}, {441}, {132}, {331}, {264}, {264}, {220}, {165},
{264}, {331}, {188}, {220}, {331}, {331}, {264}, {220}, {331}, {220}, {220}, {220}, {220},
{331}, {165}, {188}, {188}, {146}, {264}, {441}, {264}, {188}, {188}, {188}, {188}, {264},
{220}, {188}, {264}, {220}, {264}, {264}, {220}, {441}, {331}, {264}, {220}, {188},
{441}, {264}, {146}, {165}, {146}, {188}, {165}, {220}, {146}, {331}, {264}, {220},
{220}, {120}, {188}, {220}, {220}, {132}, {188}, {120}, {264}, {220}, {264}, {220},
{441}, {165}, {264}, {188}, {188}, {264}, {220}, {220}, {220}, {264}, {220}, {188}}
```

In[35]:= **fishdata = Flatten[fishdata]**

```
Out[35]= {220, 188, 165, 188, 264, 331, 264, 220, 165, 264, 188, 188, 264, 264, 188, 441,
441, 441, 220, 441, 132, 331, 264, 264, 220, 165, 264, 331, 188, 220, 331, 331,
264, 220, 331, 220, 220, 220, 220, 331, 165, 188, 188, 146, 264, 441, 264, 188, 188,
188, 188, 264, 220, 188, 264, 220, 264, 264, 220, 441, 331, 264, 220, 188, 441, 264,
146, 165, 146, 188, 165, 220, 146, 331, 264, 220, 220, 120, 188, 220, 220, 132, 188,
120, 264, 220, 264, 220, 441, 165, 264, 188, 188, 264, 220, 220, 220, 264, 220, 188}
```

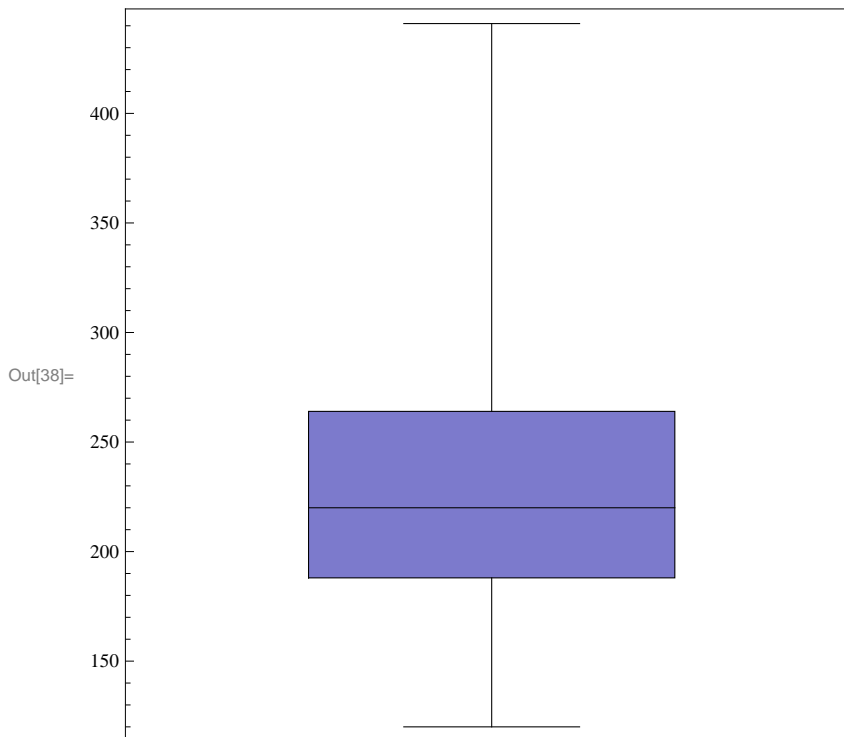
```
In[36]:= Histogram[fishdata]
```



Next use **Get** to load the **Statistical Plots** package which contains the **BoxWhiskerPlot** command (see the **Help** file for more information on this command). Then apply the **BoxWhiskerPlot** command to the fish population data to produce a *box-and-whisker plot*! This plot (also known as a *box plot*) gives the data's *median*, *1st* and *3rd quartiles*, and *smallest* and *largest* values.

```
In[37]:= << StatisticalPlots`
```

```
In[38]:= BoxWhiskerPlot[fishdata]
```



Check with the following built-in commands: **Median**, **Max**, **Min**, and **Quartiles**!

```
In[39]:= Median[fishdata]
```

Out[39]= 220

```
In[40]:= Max[fishdata]
```

```
Out[40]= 441
```

```
In[41]:= Min[fishdata]
```

```
Out[41]= 120
```

```
In[42]:= Quartiles[fishdata]
```

```
Out[42]= {188, 220, 264}
```

Note: Portions of this lecture come from the help files built into *Mathematica* 6.0. The rest was created by M. A. Karls in Fall 2008 and revised in Fall 2009.