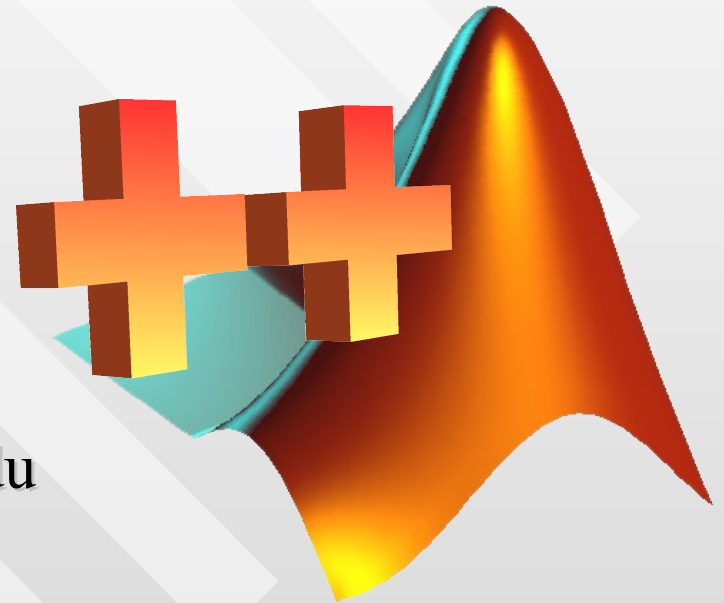


Who's the Boss?

IAP 2009

Scott Gorlin
gorlins@mit.edu



<http://stellar.mit.edu/S/project/advanced-matlab/>

Workflow

- Matlab is great for data analysis, but...
 - First have to get data IN to Matlab!
- Have to get data OUT
 - Either as figure or analyzed data
- Sometimes need other applications to do our work

Data Formats

- Matlab native
 - Proprietary *.mat files
 - Mostly incompatible with other programs
- Can save figures/plots as jpg, but sometimes this is not enough

Data Formats

- Trick #1
 - Export figures as *.emf or *.ai/*.eps
 - Enhanced Metafile for MS applications like PPT
 - Adobe Illustrator or Encapsulated Postscript format for anything else
 - These are VECTOR formats
 - Much higher resolution, lower filesize
 - Better colors (Matlab is a poor JPG renderer)
 - Editable in Word/Illustrator
 - Use PNG/TIFF if you absolutely need a raster image

Data Formats

- **ASCII Data**
 - Application stores data in a text file
 - Regardless of extension, you can view in Wordpad
- **CSV**
 - Num1, num2, num3 etc
- **TAB**
 - Num1 num2 num3 etc
- **And other formats**

Data Formats

- ASCII Data
 - Most of these can be directly imported through the workspace
 - Also see `importdata`, `textread`, and `help` for 'File Formats'

Data Formats

- Failsafe way to get data into Matlab:
 - Copy into Excel spreadsheet
 - Read spreadsheet in Matlab with `xlsread`

Data Formats

- ASCII Data – Some MAJOR problems
 - Data is much larger on disk than it needs to be
 - Can be ~10x larger, at least, than an optimized format
 - 1 byte/char = 255 values, but we use < 100 of them
 - 0-9 for numeric, A-Z,a-z for chars, never !@#\$%*)# etc
 - Sometimes 2 bytes/char ~ 65000 unused values!
 - Only good for chars or simple sets of matrices – not always appropriate
- Sometimes you just can't get it from your program!

Data Formats

- Binary Data
 - Optimized file use
 - Much smaller – data described exactly with bits, instead of in text strings
 - Application can write its own format
 - Files almost always natively binary
 - Can't be viewed with Wordpad, so we have to translate it into Matlab

Data Formats

- Binary Data
 - Sometimes format is proprietary
 - Company may provide a reading utility/SDK, ie a dll, which can be sourced in Matlab or other
 - Not every proprietary format is possible to read into Matlab, unless you hack it

Data Formats

- Binary Data
 - Many applications will use an open source or published format to store data
 - All we need to do is find the source documentation, and we can write a Matlab function to read in the data file!

Data Formats

- Finding format documentation
 - Sometimes a help file describes the binary format of data
 - Usually, with programs written in C (most are), a *structure definition* is provided in a *header file*

Data Formats

- Finding format documentation
 - Structure definition
 - Structures in C are like structures in Matlab – contain a set of related data – but they are usually scalar, not structure matrices
 - In C, a structure must be defined (because of static memory allocation) and this is usually done in a ‘header file’ with a name like ‘dat.h’, dat being the extension of the data file created
 - Usually a datafile is nothing but this structure written to disk, exactly how the structure is defined

Data Formats

- Finding format documentation
 - Often, for common data formats, you will just be provided with the header file!
 - Or your program may list the header file in an appendix, even if it is not open source
 - NIfTI is a common format for MRI volume data, <http://nifti.nimh.nih.gov/>

Nii Example

- NIfTI - *.nii files for fMRI
 - Format is documented in nifti1.h, provided on Stellar site and distributed by NIMH
 - Let's walk through this example to see how to read a custom data format...
 - *.h files are text, but Matlab can color keywords just like in m-files
 - File -> Preferences -> Editor/Debugger -> Languages
 - Select C/C++ and ensure that .h is included in File Extensions

Nii Example

- Much of the header is C code we can ignore
 - Green is documentation, commented with `//` or `/* */` for block comments

```
1  /** \file nifti1.h
2      \brief Official definition of the nifti1 header.  Written by Bob Cox, SSCC, NIMH.
3
4      HISTORY:
5
6          29 Nov 2007 [rickr]
7              - added DT_RGBA32 and NIFTI_TYPE_RGBA32
8              - added NIFTI_INTENT codes:
9                  TIME_SERIES, NODE_INDEX, RGB_VECTOR, RGBA_VECTOR, SHAPE
10     */
11
12     #ifndef _NIFTI_HEADER_
13     #define _NIFTI_HEADER_
14
15     /*****
16         ** This file defines the "NIFTI-1" header format.          **
17         ** It is derived from 2 meetings at the NIH (31 Mar 2003 and **
```

Nii Example

- Reading the documentation, we discover that a *.nii file contains the following, in order:
 - A ‘header’ describing the data
 - Any number of optional ‘extensions’
 - Raw binary data

Nii Example

NIFTI-1 FILE STORAGE:

"nil" means that the image data is stored in the ".img" file corresponding to the header file (starting at file offset 0).

"n+1" means that the image data is stored in the same file as the header information. We recommend that the combined header+data filename suffix be ".nii". When the dataset is stored in one file, the first byte of image data is stored at byte location (int)vox_offset in this combined file. The minimum allowed value of vox_offset is 352; for compatibility with some software, vox_offset should be an integral multiple of 16.

Nii Example

- So, we must:
 - Read the header
 - Interpret parts of the header, such as size of the image
 - Find and read image data, and assemble according to header
- Not all formats will be so well documented
 - Generally, a file will just be one structure or one N-dimensional matrix and will be easier to interpret

Nii Example

- Nii Header:
 - `struct` keyword followed by a list of data types and their names

```
138  /*! \struct nifti_1_header
139      \brief Data structure defining the fields in the nifti1 header.
140          This binary header should be found at the beginning of a valid
141          NIFTI-1 header file.
142  */
143
144  struct nifti_1_header { /* NIFTI-1 usage          */ /* ANALYZE 7.5 field(s) */
145                          /******                */ /******                */
146
147                          /*--- was header_key substruct ---*/
148  int    sizeof_hdr;      /*!< MUST be 348          */ /* int sizeof_hdr;      */
149  char   data_type[10];   /*!< ++UNUSED++         */ /* char data_type[10];  */
150  char   db_name[18];     /*!< ++UNUSED++         */ /* char db_name[18];    */
151  int    extents;        /*!< ++UNUSED++         */ /* int extents;         */
152  short  session_error;  /*!< ++UNUSED++         */ /* short session_error; */
153  char   regular;        /*!< ++UNUSED++         */ /* char regular;        */
154  char   dim_info;       /*!< MRI slice ordering. */ /* char hkey_un0;       */
155
```

Data Formats

- Reading a binary file

- `fid = fopen(filename, permission_tmode, machineformat)`
- Filename is a string, name of your data file
- Permission also a string, and should be 'r' for reading (defaults in binary mode)
- Machineformat is critical: refers to Big Endian or Little Endian
 - 'b' for Big, 'l' for Little

Data Formats

- Endianness
 - Refers to whether digits get big L->R or R->L
 - For binary string 0101:
 - Little Endian: $[0\ 1\ 0\ 1] * 2^{[0\ 1\ 2\ 3]'} = 10$
 - Big Endian: $[0\ 1\ 0\ 1] * 2^{[3\ 2\ 1\ 0]'} = 5$
 - PC's are typically LE, Macs may be BE
 - Make sure to specify according to your data, otherwise it may be wrong!
 - Easy to try both ways in debugging

Data Formats

- We have opened a ‘filestream,’ now we can read each data item into a structure field
 - `Fread`, `fscanf`, etc
- C syntax is different than M syntax
 - `int`, `char`, `single`, etc – look up their equivalents in help for `fread`

Nii Example

- For instance,
 - Load_nifti_hdr from
Freesurfer analysis package,
posted online


```
/*! \struct nifti_1_he
    \brief Data struct
        This binary
        NIFTI-1 hea
*/

struct nifti_1_header
```

Notice the use of arrays

```
76 - hdr.data_type      = fscanf(fp, '%c', 10);
77 - hdr.db_name       = fscanf(fp, '%c', 18);
78 - hdr.extents      = fread(fp, 1, 'int');
79 - hdr.session_error = fread(fp, 1, 'short');
80 - hdr.regular      = fread(fp, 1, 'char');
81 - hdr.dim_info     = fread(fp, 1, 'char');
```

```
int    sizeof_hdr;
char   data_type[10];
char   db_name[18];
int    extents;
short  session_error;
char   regular;
char   dim_info;
```



Data Formats

- Tip #1
 - Using a command like `fread` may return a double, even if you have read an integer or char
 - You can ‘typecast’ by calling `char(readData)`, etc

Data Formats

- Tip #2
 - A data item may be another structure. This will probably also be defined in the same .h file
 - In nifti1.h, the nii header is followed by two structures:

```
287 struct nifti1_extender { char extension[4] ; } ;
288 typedef struct nifti1_extender nifti1_extender ;
289
290 /*! \struct nifti1_extension
291     \brief Data structure defining the fields of a he
292     */
293 struct nifti1_extension {
294     int     esize ; /*!< size of extension, in bytes (m
295     int     ecode ; /*!< extension code, one of the NIF
296     char *  edata ; /*!< raw data, with no byte swappin
297 } ;
298 typedef struct nifti1_extension nifti1_extension ;
299
```

Data Formats

- Reiteration:
 - A structure in C is just the collection of native data variables written to disk, in the order they are defined
 - You do not read the structure from the file, but create an identical structure in matlab by assembling each data item
 - There may be no 'binary space' between relevant data items – every bit of the file gets mapped into one of the data fields

Data Formats

- Notes on C syntax

- Arrays are declared by size, with [] `char data_type[10];`

- The ‘*’ means a pointer

- This may be useless, or the file may contain the dereferenced data
- Here, edata is actually stored as a character array of size esize, so we can read it (this is a common way to store strings in C)

```
293 struct nifti1_extension {
294     int     esize ; /*!< size of extension, in bytes (m
295     int     ecode ; /*!< extension code, one of the NIF
296     char * edata ; /*!< raw data, with no byte swappin
297 } ;
```

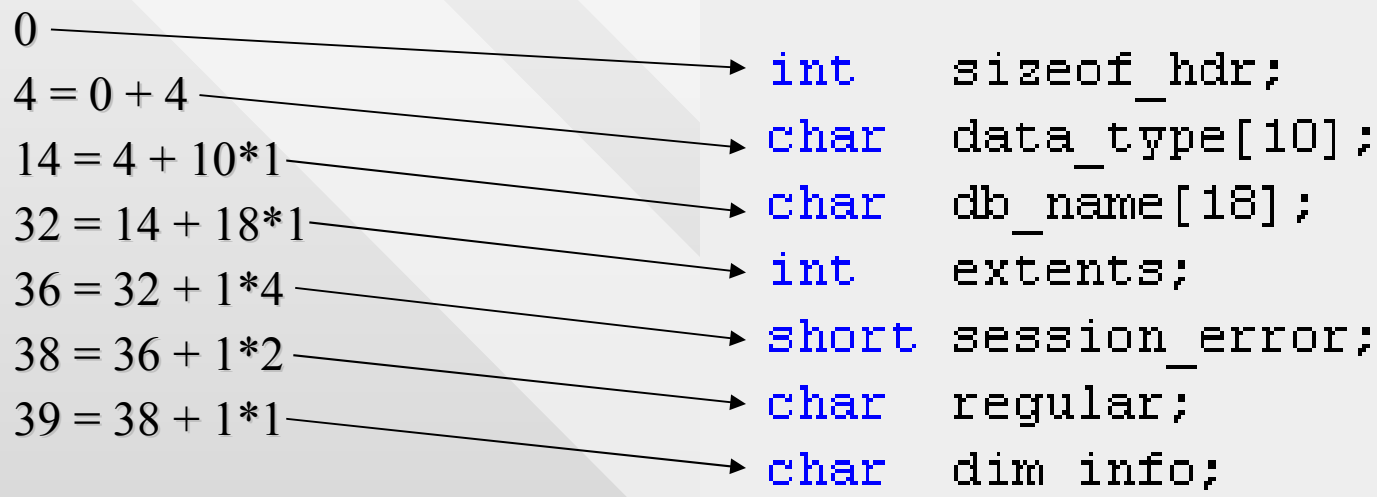
Data Formats

- Tip #3
 - If you don't need every data item, you can use `fseek` to jump to the location you need
 - When you read, your location in the file progresses from the beginning of the variable to the end
 - You travel `count*nbytes` per read, `nbytes = 4` for `int`, `1` for `int8`, `8` for `double`, etc (see `fread`)

Data Formats

- Tip #3

- Example positions:



Data Formats

- Tip #4
 - Writing a file is just the opposite: fwrite
 - Just write out, in order, all the relevant fields and your program will be able to open the file
- Tip #5
 - Write a class!!

Interaction

- Data file passing is good, but sometimes we want *interactivity* between applications
- ‘External Interfaces’ in help
 - DLL’s: code libraries written in C to control hardware, etc, can often be called in Matlab
 - See `loadlibrary`
 - Custom C/Java code to interact with X
 - COM and ActiveX: Windows native communication protocol

Interaction

- **Active X**
 - Allows programs to control other programs in Windows
 - Client: calling program
 - Server: program started to run a specific task (slave)

Interaction

- Active X
 - MANY examples
 - Each program works differently, have to read docs to use correctly
 - Most extensively used are MS programs like Excel, Word, PPT

ActiveX Example

- Matlab Automation Server

- 2nd lecture

- Ie,

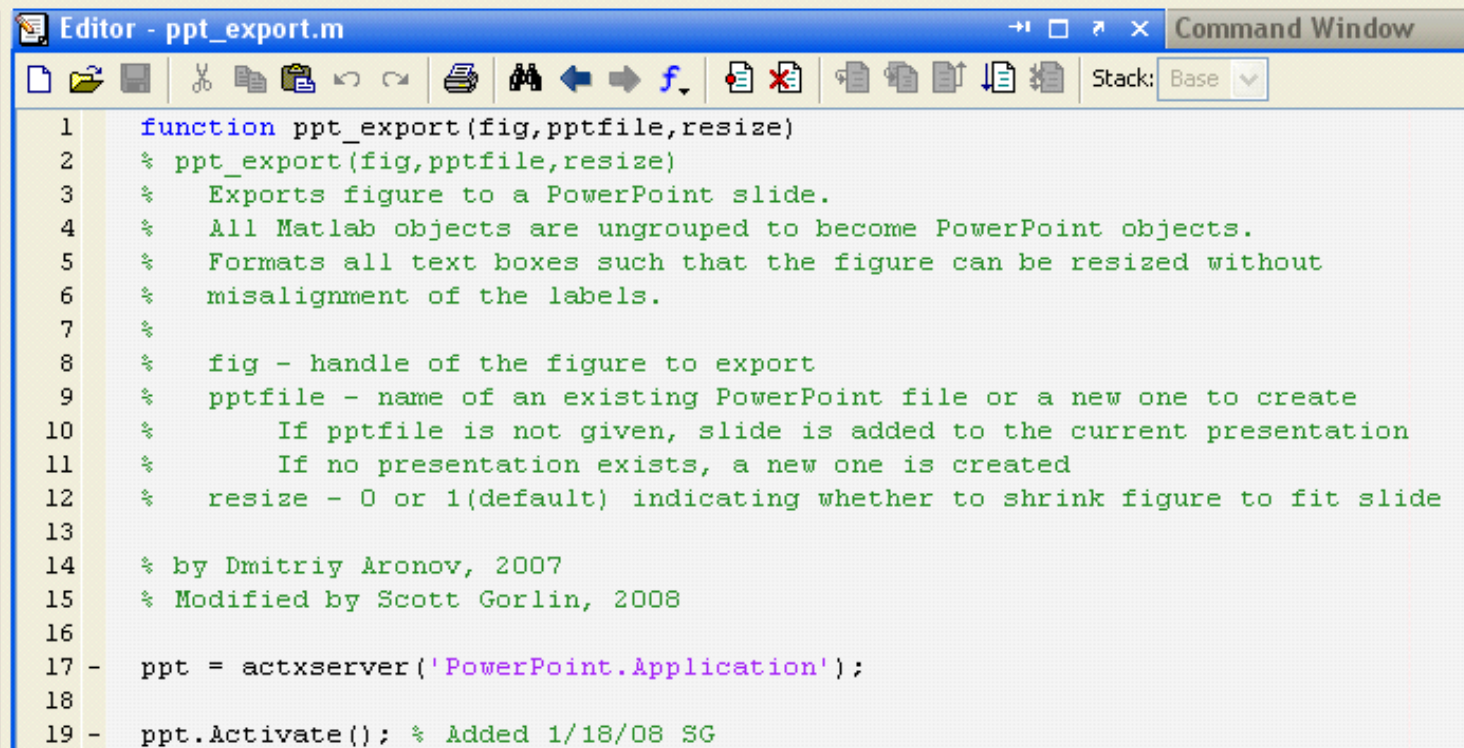
```
for i = 1:ncores
    h{i} = actxserver('matlab.application')
    h{i}.Execute('t = timer(''TimerFcn'', ''batchData(...)''); start(t)');
```



NB: double ', not “

ActiveX Example

- PowerPoint Slide Generator
 - Posted online



```
1 function ppt_export(fig,pptfile,resize)
2 % ppt_export(fig,pptfile,resize)
3 % Exports figure to a PowerPoint slide.
4 % All Matlab objects are ungrouped to become PowerPoint objects.
5 % Formats all text boxes such that the figure can be resized without
6 % misalignment of the labels.
7 %
8 % fig - handle of the figure to export
9 % pptfile - name of an existing PowerPoint file or a new one to create
10 % If pptfile is not given, slide is added to the current presentation
11 % If no presentation exists, a new one is created
12 % resize - 0 or 1(default) indicating whether to shrink figure to fit slide
13
14 % by Dmitriy Aronov, 2007
15 % Modified by Scott Gorlin, 2008
16
17 - ppt = actxserver('PowerPoint.Application');
18
19 - ppt.Activate(); % Added 1/18/08 SG
```

ActiveX Example

- **PowerPoint Slide Generator**

- `ppt = actxserver('PowerPoint.Application');`

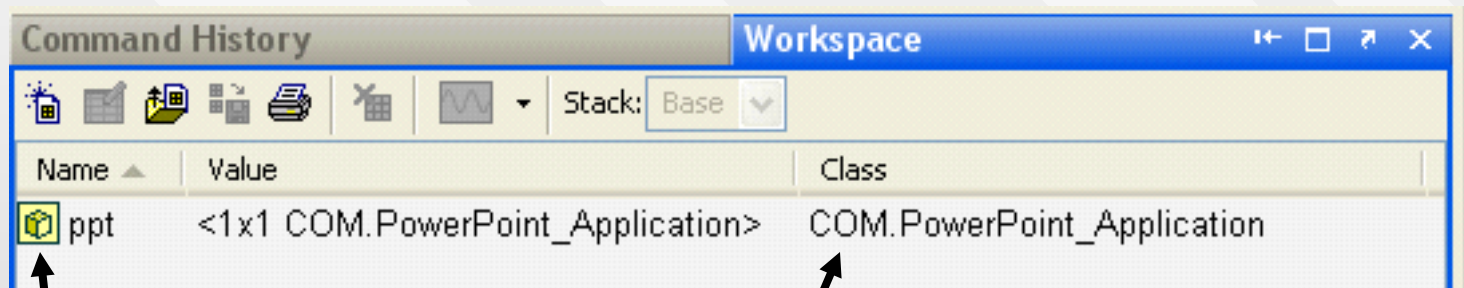
- `ppt.Activate();`

- **Creates `ppt`, a COM object, and uses it to launch PowerPoint**

- **Then, does a lot of fancy stuff – see online for details**

COM Objects

- Working with COM



Symbol for
'object,' and
here is passed by
reference

Object type

COM Objects

- Working with COM
 - Server application may have help on COM usage, but may not
 - Usually have to play/hack/try, but this isn't too hard

COM Objects

- Working with COM
 - Again, like with any object, we can:
 - `fields(obj)`
 - `methods(obj)`
 - `methodsview(obj)`
 - Because these objects are designed for external use, the methods are easy to understand and use

COM Objects

- `Execute(handle, string)`

Is the same as

- `h.Execute(string)`

```
//  
>> h = actxserver('matlab.application')
```

```
h =
```

```
COM.matlab_application
```

```
>> methodsview(h)
```

```
>>
```

Methods of class COM.matlab_application.release

Return Type	Name	Argu
string	Execute	(handle, string)
Variant(Pointer)	Feval	(handle, string, int32, Variant(Optional))
string	GetCharArray	(handle, string, string)
[SafeArray Pointer(do GetFullMatrix		(handle, string, string, SafeArray Pointer(double), Sa
Variant	GetVariable	(handle, string, string)
Variant(Pointer)	GetWorkspaceData	(handle, string, string)
	MaximizeCommandWindow	(handle)
	MinimizeCommandWindow	(handle)

COM Objects

- Working with COM
 - Dot notation (also with Java objects)
 - `Activate(obj)` is the same as `obj.Activate()`
 - Philosophically better
 - Represents object doing something, or a method specific to an object
 - Will only use dot notation when we write Java/C (Matlab is the oddball)

Friday

- Javalab
 - Extending Matlab, for power and speed
 - Native use and disuse