MATLAB EXPO 2015
KOREA
2015년 5월 21일 목요일
인터컨티넨털 코엑스, 서울
생산성 증대를 위한 MATLAB 개발 환경 및 프로그래밍 기법

Application Engineer
엄 준 상 대리
Agenda

- Large Data set and efficient and parallel programming
- Improving code quality and performance
- Parallel Computing on Desktop
- GPU Computing
Example:
Large Data set and efficient programming

- Evaluate function at grid points
- Reevaluate function over larger blocks
- Compare the results
- Evaluate code performance
Summary of Example

- Used built-in timing functions
  
  \[
  \text{>> tic} \\
  \text{>> toc}
  \]

- Used Code Analyzer to find suboptimal code

- Preallocated arrays

- Vectorized code
Effect of Not Preallocating Memory

```matlab
>> x = 4
>> x(2) = 7
>> x(3) = 12
```

Resizing Arrays is Expensive
Benefit of Preallocation

>> x = zeros(3,1)
>> x(1) = 4
>> x(2) = 7
>> x(3) = 12

Reduced Memory Operations
Data Storage of MATLAB Arrays

>> x = magic(3)
x =
     8     1     6
     3     5     7
     4     9     2

Column-Major Memory Storage

See the June 2007 article in “The MathWorks News and Notes”:
Code Quality

- Writing “better” code
  - Less error-prone
  - Human readable code
  - Performance tuning

- Robustness
  - Validate, guard inputs/o
  - Handle errors, exceptions
Improving Code Quality in MATLAB

- Analyzing code
- Checking McCabe complexity
- Debugging
- Input and error handling
MATLAB Code Analyzer

- Optimize your code and avoid syntax errors
- Automatically check code in Editor
- Run on multiple files in folder and generate a report
Check McCabe Complexity

- McCabe complexity (\texttt{checkcode -cyc})
  - Quantitative measure of the complexity of a program

- Can lower the complexity by dividing a function into smaller, simpler functions

- Good rule of thumb is to aim for complexity around 10 or lower

Lower complexity $\Rightarrow$ Easier to understand, modify
Higher complexity $\Rightarrow$ More likely to contain errors

<table>
<thead>
<tr>
<th>Complexity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-10</td>
<td>Simple program, low risk</td>
</tr>
<tr>
<td>10-20</td>
<td>Moderate risk</td>
</tr>
<tr>
<td>20-50</td>
<td>High risk</td>
</tr>
<tr>
<td>50+</td>
<td>Untestable program, very high risk</td>
</tr>
</tbody>
</table>
Debugging with MATLAB

- Diagnose problems
  - Graphical user interface in Editor
  - Command line interface

- Set standard, conditional, or error breakpoints

- Step through a file and examine variable values
Measuring Code Performance

- **tic and toc**
  - For timing for smaller portions of code and scripts
  - Measures performance using a stopwatch timer

- **timeit**
  - For timing a function
  - Measures the function multiple times and computes the median

- **Profiler**
  - For identifying specific performance bottlenecks in code
  - Measures relative execution time
How does MATLAB store data?

*Container overhead*

```
\texttt{d = [1 2]
dcell = {{1 2}}
dstruct.d = [1 2]}
```

* Using values for 64-bit MATLAB
Sparse Matrices

- Require less memory and are faster

- When to use sparse?
  - < 1/2 dense on 64-bit (double precision)
  - < 2/3 dense on 32-bit (double precision)

- Functions that support sparse matrices
  >> help sparfun

- Blog Post: Creating Sparse Finite Element Matrices
Reading in Part of a Dataset from Files

- MAT file
  - Load and save part of a variable using the `matfile`

- ASCII file
  - Selectively choose rows and columns using `textscan`
  - Pointer keeps track of location in file

- Binary file
  - Read and write directly to/from file using `memmapfile`
  - Maps address space to file
**Batch processing…**
Load the entire file and process it all at once

**Stream processing**
Load a frame and process it before moving on to the next frame
System Objects

- A class of MATLAB objects that support streaming workflows

- Simplifies data access for streaming applications
  - Manages flow of data from files or network
  - Handles data indexing and buffering

- Contain algorithms to work with streaming data
  - Manages algorithm state
  - Available for Signal Processing, Communications, Video Processing, and Phased Array Applications

Available from
- DSP System Toolbox
- Communications System Toolbox
- Computer Vision System Toolbox
- Phased Array System Toolbox
- Image Acquisition Toolbox
Parallel Computing with MATLAB

User’s Desktop

Parallel Computing Toolbox

MATLAB Workers

MATLAB Distributed Computing Server

Compute Cluster
Example: Parameter Sweep of ODEs

- Solve a 2\textsuperscript{nd} order ODE
  \[ m\ddot{x} + b\dot{x} + kx = 0 \]
  \[ 1,2,\ldots \quad 1,2,\ldots \]

- Simulate with different values for \( b \) and \( k \)

- Record peak value for each run

- Plot results
Summary of Example

- Mixed task-parallel and serial code in the same function
- Ran loops on a pool of MATLAB resources
- Used Code Analyzer to help in converting existing `for`-loop into `parfor`-loop
Parallel Computing enables you to …

- Speed up Computations
- Work with Large Data

Larger Compute Pool

Larger Memory Pool
Gaining Performance with More Hardware

Using More Cores (CPUs)

Using GPUs

Device Memory
Overloaded MATLAB Functions

A = magic(1000);
G = gpuArray(A); %Push to GPU memory
b = rand(1000,1,'gpuArray'); %Create on GPU
F = fft(G);
x = G\b;
z = gather(x); %Bring back into MATLAB

Full list of built-in functions that support GPUArray
Parallel Computing → GPU Computing
→ Establish Arrays on a GPU
→ Run Built-In Functions on a GPU
MATLAB EXPO 2015
Using `arrayfun` on GPU

```matlab
gain = 1.5;
offset = -0.1;
x = rand(1000,1,'gpuArray');  %Create on GPU
x = arrayfun(@myGPUfun, x)  %Execute on GPU

function c = myGPUfun(x, gain, offset)
c = (x .* gain) + offset;
end
```

Full list of functions for use with `arrayfun` on GPU

Parallel Computing → GPU Computing  
→ Run Element-wise MATLAB Code on GPU
Key Takeaway

- Consider performance benefit of vector and matrix operations in MATLAB

- Leverage parallel computing tools to take advantage of additional computing resources

- Consider Code Quality and Performance
Additional Resources

- **Documentation**
  - Source Control Integration
  - Techniques for Improving Performance
  - Unit Testing Framework
  - Toolbox Distribution and Documentation Tools

- **Webinars**
  - Programming with MATLAB
  - Speeding up MATLAB Applications
  - Managing and Sharing MATLAB Code

- **MATLAB Central**
  - Open exchange for the MATLAB and Simulink user community
Questions?