Accelerating MATLAB algorithms

- Code improvement
- Parallel computing
- C code generation
Want faster MATLAB code?
Want faster MATLAB code?

Optimise

Accelerate through code improvement
Where are the bottlenecks?

unnecessary output

memory allocation

vectorisation
How long can you work on making a routine task more efficient before you're spending more time than you save? (Across five years)

<table>
<thead>
<tr>
<th>How much time you shave off</th>
<th>50/day</th>
<th>5/day</th>
<th>Daily</th>
<th>Weekly</th>
<th>Monthly</th>
<th>Yearly</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 second</td>
<td>1 day</td>
<td>2 hours</td>
<td>30 minutes</td>
<td>4 minutes</td>
<td>1 minute</td>
<td>5 seconds</td>
</tr>
<tr>
<td>5 seconds</td>
<td>5 days</td>
<td>12 hours</td>
<td>2 hours</td>
<td>21 minutes</td>
<td>5 minutes</td>
<td>25 seconds</td>
</tr>
<tr>
<td>30 seconds</td>
<td>4 weeks</td>
<td>3 days</td>
<td>12 hours</td>
<td>2 hours</td>
<td>30 minutes</td>
<td>2 minutes</td>
</tr>
<tr>
<td>1 minute</td>
<td>8 weeks</td>
<td>6 days</td>
<td>1 day</td>
<td>4 hours</td>
<td>1 hour</td>
<td>5 minutes</td>
</tr>
<tr>
<td>5 minutes</td>
<td>9 months</td>
<td>4 weeks</td>
<td>6 days</td>
<td>21 hours</td>
<td>5 hours</td>
<td>25 minutes</td>
</tr>
<tr>
<td>30 minutes</td>
<td>6 months</td>
<td>5 weeks</td>
<td>5 days</td>
<td>1 day</td>
<td>2 hours</td>
<td></td>
</tr>
<tr>
<td>1 hour</td>
<td>10 months</td>
<td>2 months</td>
<td>10 days</td>
<td>2 days</td>
<td>5 hours</td>
<td></td>
</tr>
<tr>
<td>6 hours</td>
<td>2 months</td>
<td>2 weeks</td>
<td>1 day</td>
<td>2 weeks</td>
<td>1 day</td>
<td></td>
</tr>
<tr>
<td>1 day</td>
<td>8 weeks</td>
<td>5 days</td>
<td>1 day</td>
<td>2 weeks</td>
<td>1 day</td>
<td>5 days</td>
</tr>
</tbody>
</table>

speed

memory

readability
Want faster MATLAB code?

Optimise

Parallelise

Accelerate with parallel computing
Tailoring Parallel Computing

Ease of Use

CPU

Built in support

Task intensive

parfor jobs & tasks

Options available

Data intensive

distributed arrays
spmd

communicating job

Greater Control
cluster
Parallel Computing Toolbox

Compute cluster/cloud

MATLAB Distributed Computing Server

Scheduler

Worker

Worker

Worker

Worker
GPU arrays
Tailoring Parallel Computing

- Options available
  - GPU
    - gpuArray
    - arrayfun
    - CUDA code

Ease of Use

Greater Control
Computation time exceeds data transfer time?

- yes
  - Running independent operations on many elements?
    - yes
      - Parallelise on GPU
    - no
      - Stick with CPU
- no
  - Stick with CPU
Want faster MATLAB code?

- Optimise
  - yes
  - Parallelise
    - yes
    - Generate MEX

Accelerate with MATLAB Coder
Introducing MATLAB Coder

MATLAB code

function [mean, stdev] = stats(vals)
    % #codegen
    % calculates a statistical mean and a standard deviation
    stdev = sqrt(sum((vals - avg(vals, len)).^2)/len);
    plot(vals, "-.");

C code

public static void Main()
{
    List<Circle> circleList = new List<Circle>();
    circleList.Add(new Circle(3, 0, 5));
    circleList.Add(new Circle(1, 4, 10));

    List<Rectangle> rectangleList = new List<Rectangle>();
    rectangleList.Add(new Rectangle(0, 0, 10, 20));
    rectangleList.Add(new Rectangle(1, 2, 20, 30));
    rectangleList.Add(new RightTriangle());
    triangleList.Add(new RightTriangle(20, 40, 1, 2));
    triangleList.Add(new RightTriangle(200, 400, 11, 21));
    triangleList.Add(new RightTriangle(78, 42, 17, 21));
}

Compiled C code (MEX)
Optimise

Parallelise

Generate MEX

...thanks for listening