MATLAB Code Optimization

Hossein Pourreza

Jan 31, 2017

Outline

- Writing MATLAB code
- Bottleneck analysis
- Optimization techniques
- Parallelization

MATLAB

- MATLAB is a programming environment as well as a high level interpreted language

- Flexible and extendable but slower than a compiled language

- Hard to maintain a MATLAB code, if not properly used or commented

Profiling

- Before doing any optimization, you need to identify the bottlenecks
  - Where most of time is spent in your code

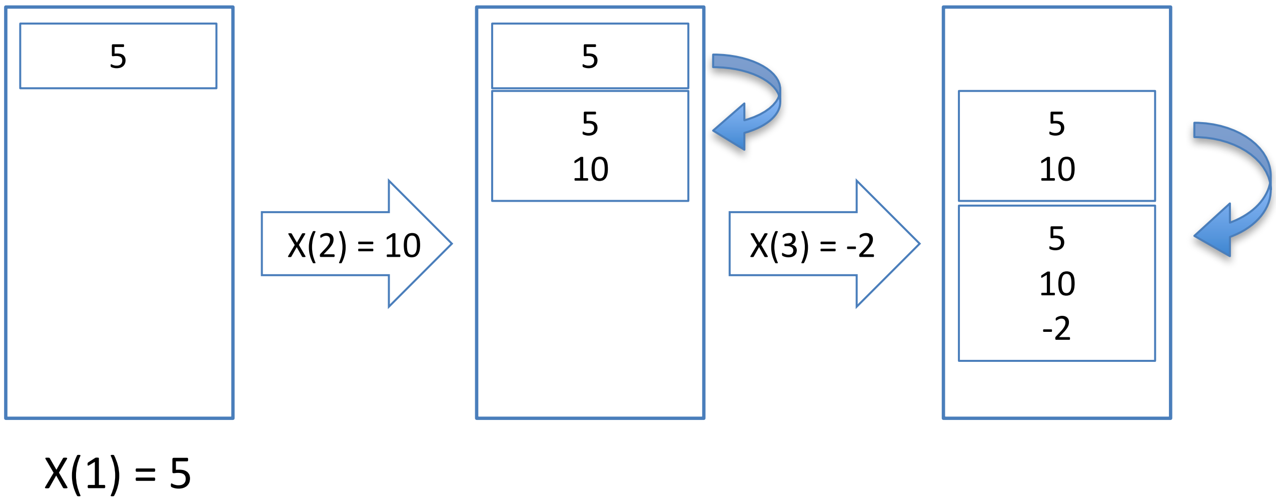
- It is important to have functions instead of scripts

- MATLAB comes with a built-in profiler
  - profile on, profile off, profile viewer commands
  - Click on Run and Time from the top ribbon

Pre-allocating memory

- Without pre-allocation

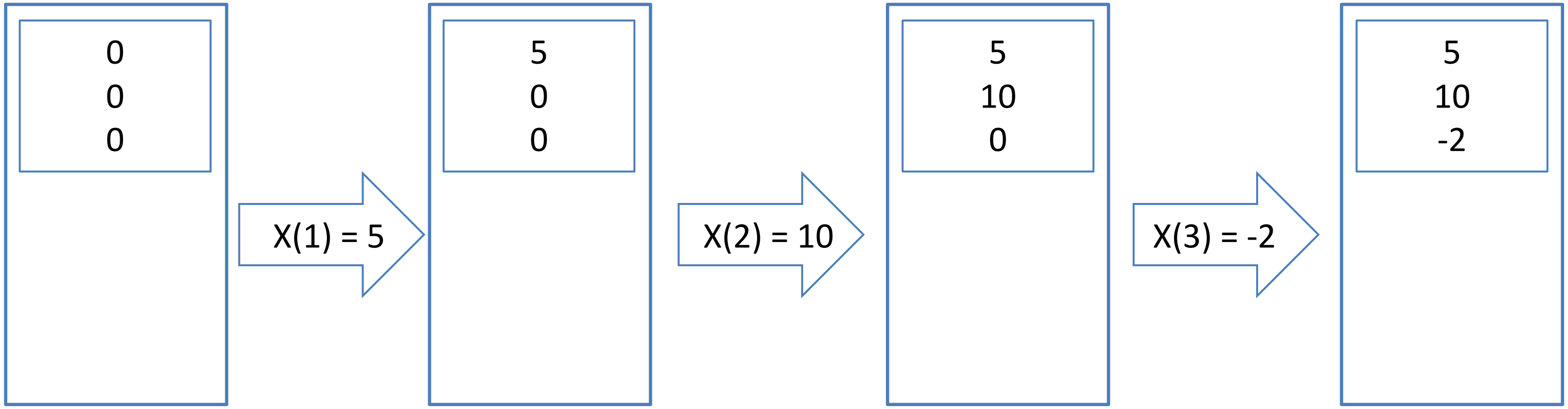
```
X(1) = 5
X(2) = 10
X(3) = -2
```



Pre-allocating memory

- With pre-allocation

```
X = zeros(3,1)
X(1) = 5
X(2) = 10
X(3) = -2
```



X = zeros(3)

Vectorization

- Avoiding loops and using operations that work on data items in parallel
- Vectorized operations can take advantage of underlying optimized routines (e.g., BLAS)
- Most of MATLAB’s built-in functions are vectorized

Parallel Computing Toolbox

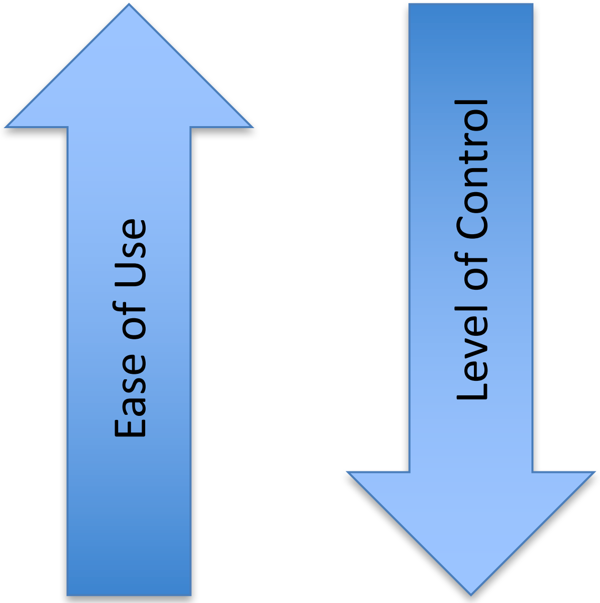
- To take advantage of cores on a node, a cluster, or a cloud

- To create MATLAB workers and assign tasks

- Can be done in different levels

Writing parallel applications

- Built-in support
  - Usually, by setting ‘UseParallel’ to true
- High level programming constructs
  - parfor, batch
- Low level programming construct
  - spmd, parfeval



Built-in support

- Different toolboxes have the capability to run in parallel
  - Optimization toolbox
  - Statistics toolbox
  - Bioinformatics toolbox

- List of functions with built-in parallel support
  - https://www.mathworks.com/products/parallel-computing/parallel-support.html

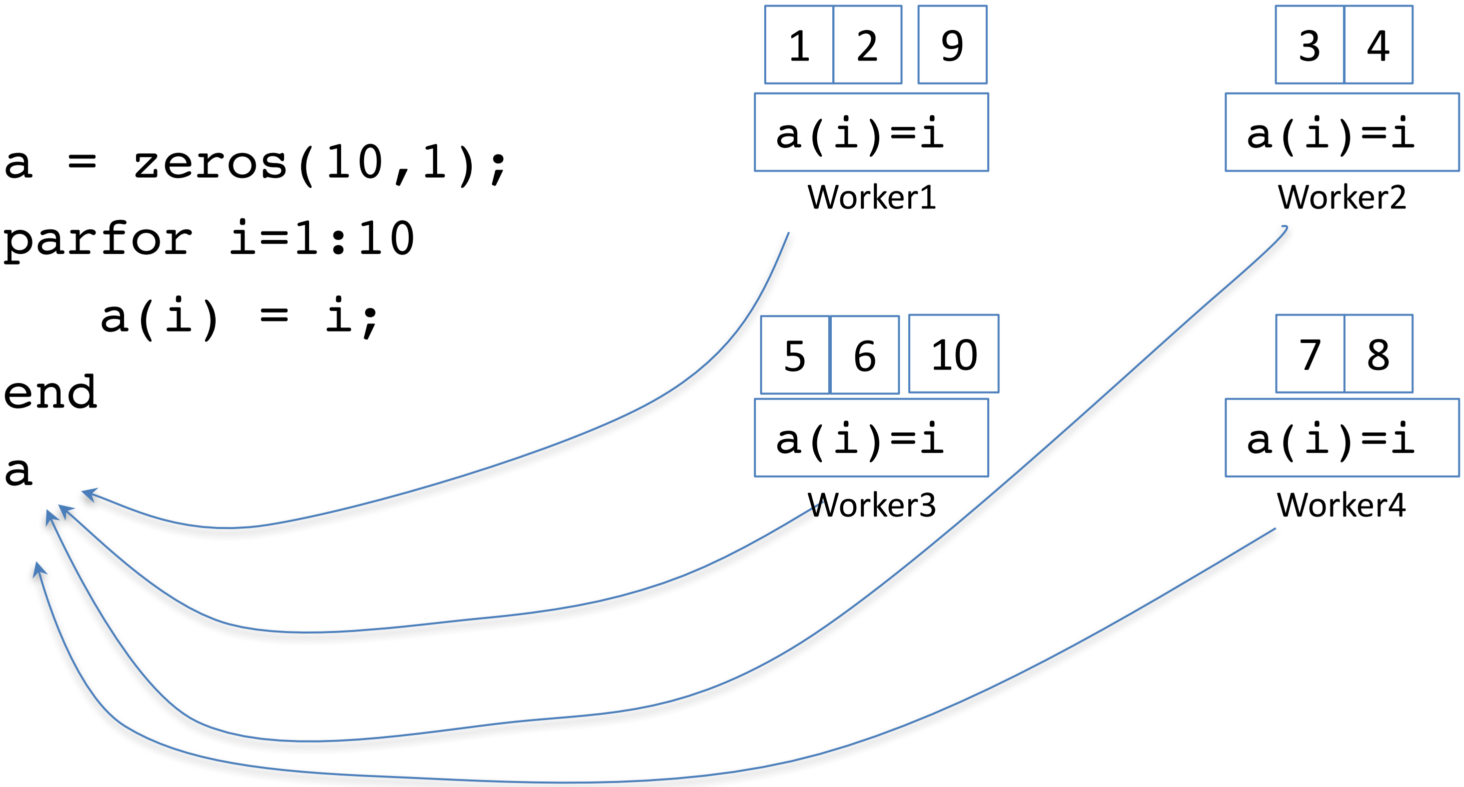
Using parfor

- parfor distributes iterations of an independent loop among MATLAB workers

- Constraints on the body of loop
  - No break or return
  - No nested parfor
  - No introduction of new variables (e.g., load, eval, etc.)

Mechanics of parfor

```
a = zeros(10,1);
parfor i=1:10
   a(i) = i;
end
a
```

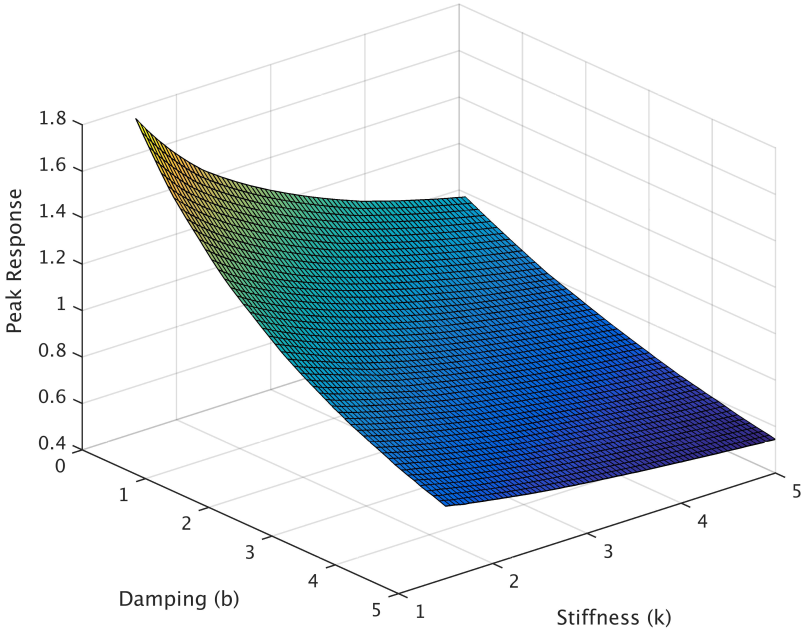


Parameter sweep

- Exploring all the options and selecting the max

```
mx"+bx'+kx=0, b=1,2,… and k=1,2,...
```

- Plotting the result



Next step

- Write C code and integrate it into your MATLAB code

- Use advanced constructs such as spmd to create parallel tasks

- Use GPU to speedup some computations more
  - Several ways of using GPUs: built-in functions, using gpuarray, and invoking CUDA kernels

Useful links

- http://ubcmatlabguide.github.io
- https://www.mathworks.com/videos/optimizing-and-accelerating-your-matlab-code-107711.html
- http://blogs.mathworks.com/loren/
- http://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.1001745
- https://www.mir.wustl.edu/Portals/0/Documents/Uploads/CHPC/PCT_Masterclass.pdf