MATLAB EXPO 2016
KOREA

4월 28일 (목)

등록 하기 matlabexpo.co.kr

MATLAB Programming Techniques for Efficiency and Performance

성 호 현 차장
Senior Application Engineer
The MathWorks Korea

Agenda

- MATLAB Infrastructure
  - Editor
  - Graphics
- Workflows
  - Managing / Testing Code
  - Sharing Apps and Custom Toolboxes
- Performance
  - Acceleration Strategy
  - Execution Engine
  - Parallel computing and GPU computation
- Wrap up & QnA

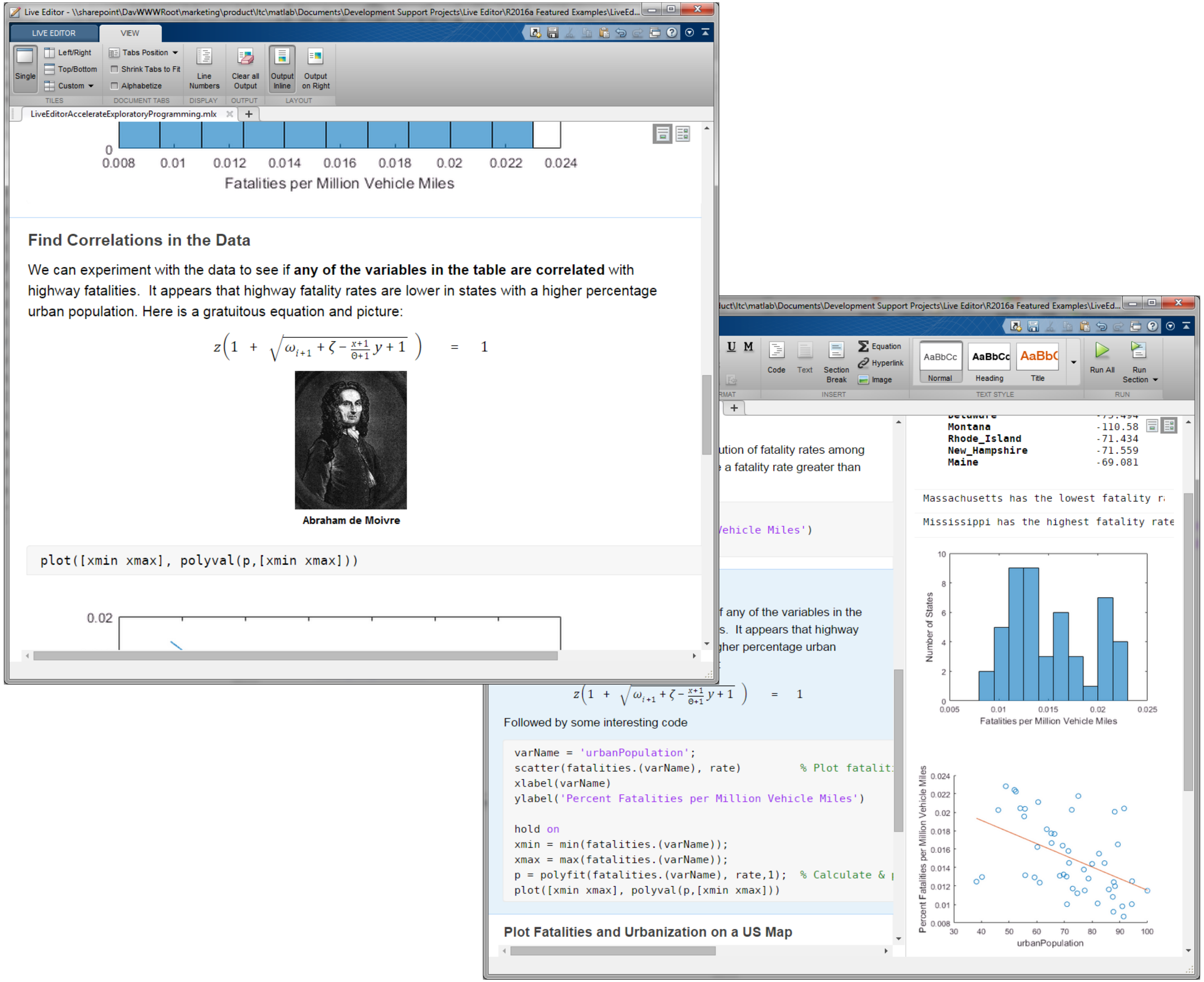
Live Editor

Modes

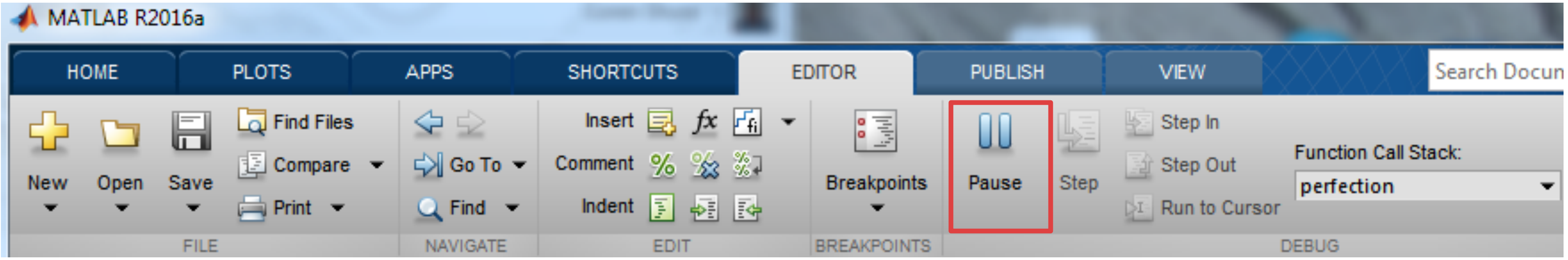
- Accelerate exploratory programming
- Create an interactive narrative
- Teach with interactive documents

Symbolic Math Toolbox support

- Alternate for MuPAD notebooks
- Typeset equations

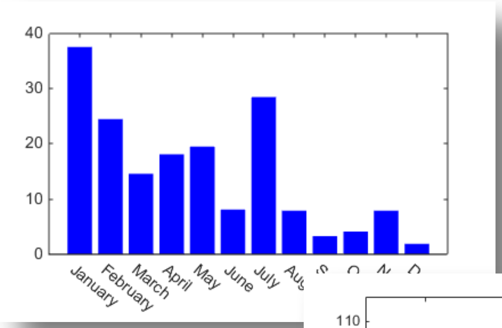


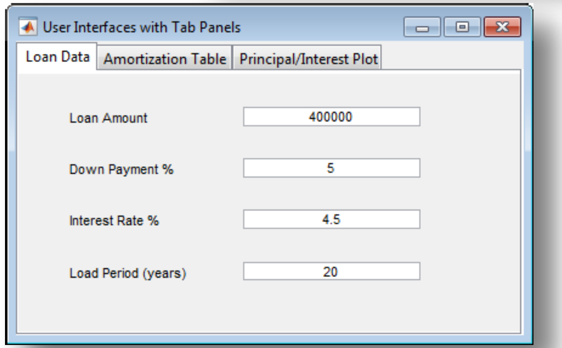
Pause Button in Classic Editor/Debugger



New Graphics System

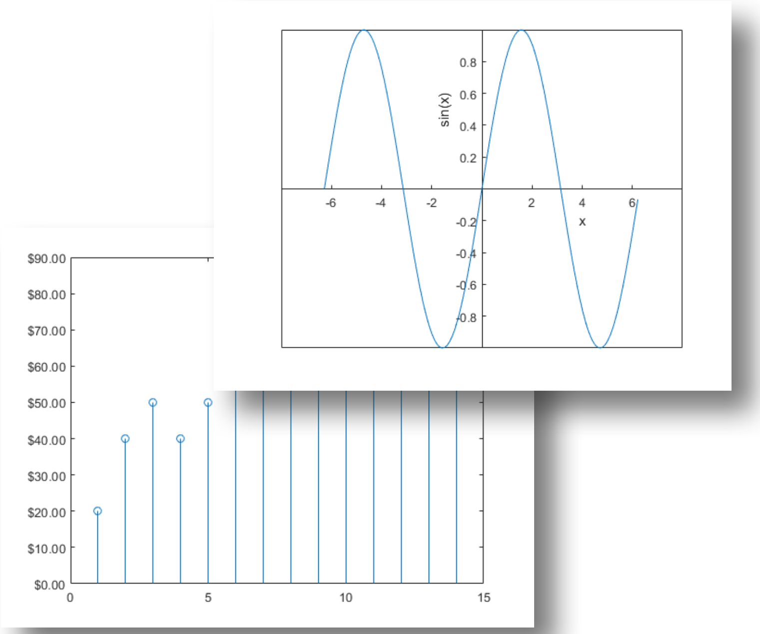
- Rotatable tick labels
- Automatic updating of `datetime` tick labels
- New visualization functions
  - `histogram`
  - `animatedline`
- Multiple colormaps per figure
- Multilingual text and symbols
- User interfaces with tab panels



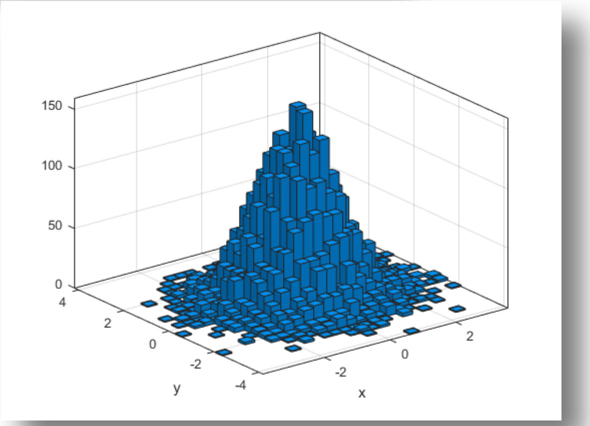


Visualization Enhancements

- Graphics enhancements for customizing plot axes
  - Setting locations to cross at the origin
  - Controlling the appearance of an individual axis in a plot

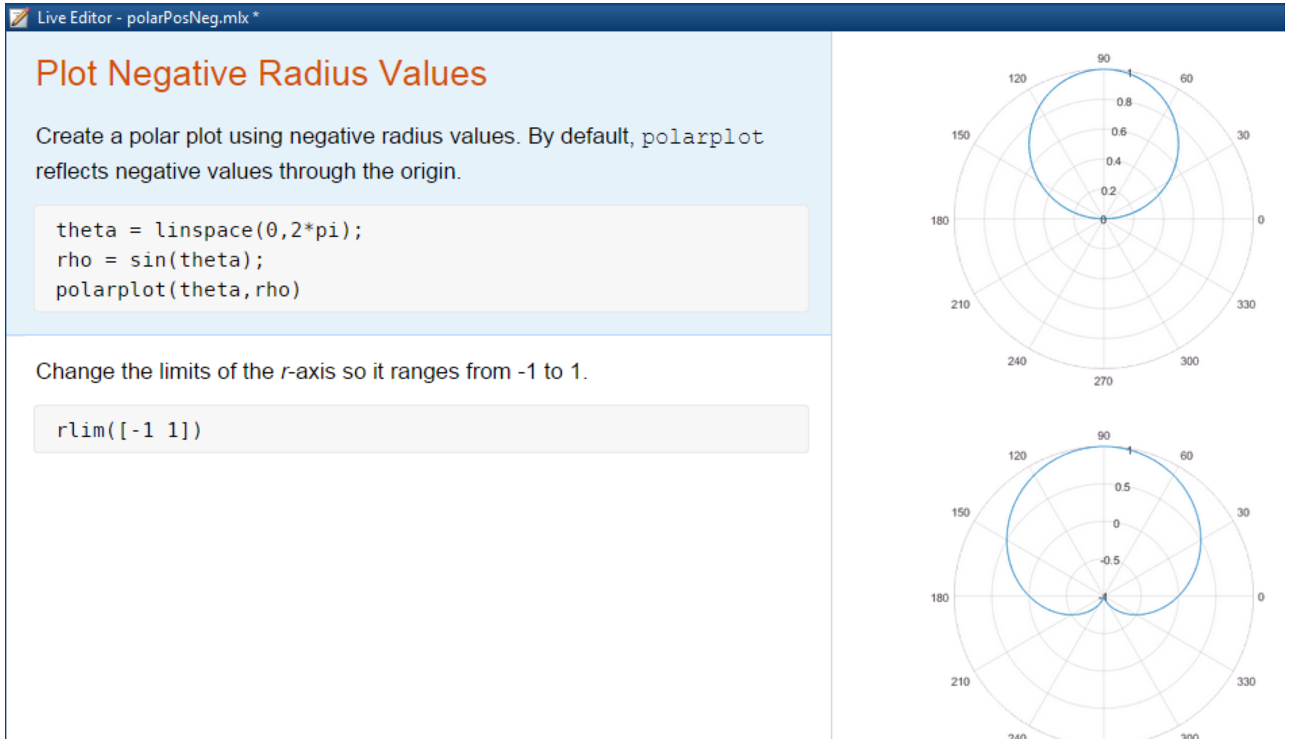


- New functions for bivariate histograms
  - Plot using **`histogram2`**
  - Bin using **`histcounts2`**



More Graphics Features

- `polarplot`
  - Incuding negative radial axis limits
- Family of parametric plotting functions
  - `fplot`
  - `fplot3`
  - `fcontour`
  - `fsurf`
  - `fmesh`

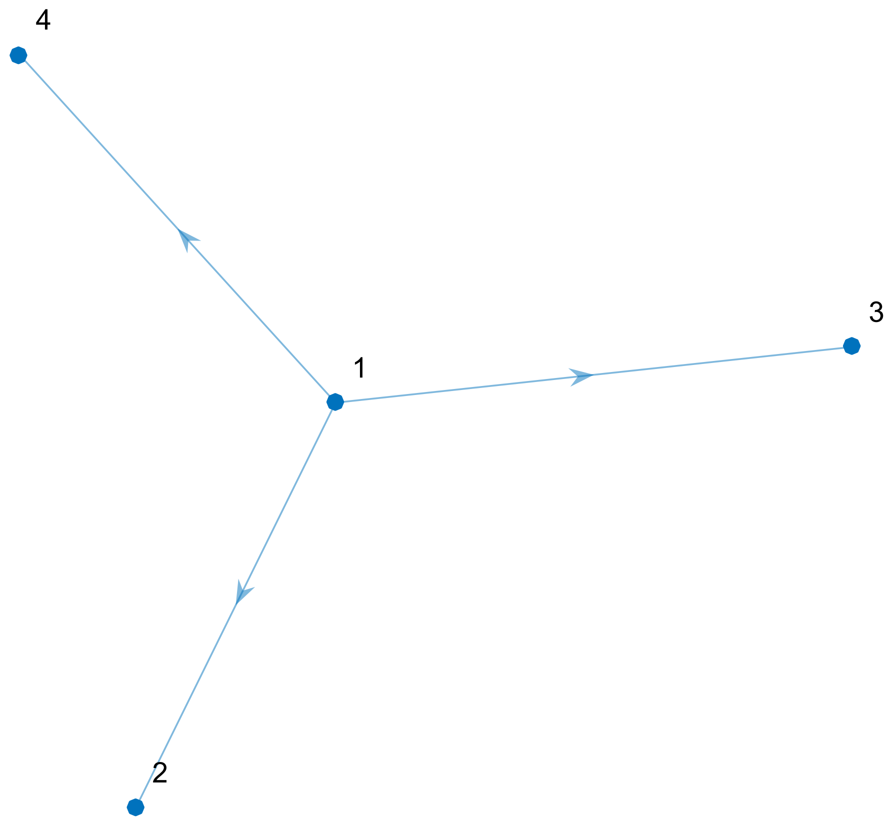


Graphs in MATLAB

A directed graph

with four nodes

and three edges.



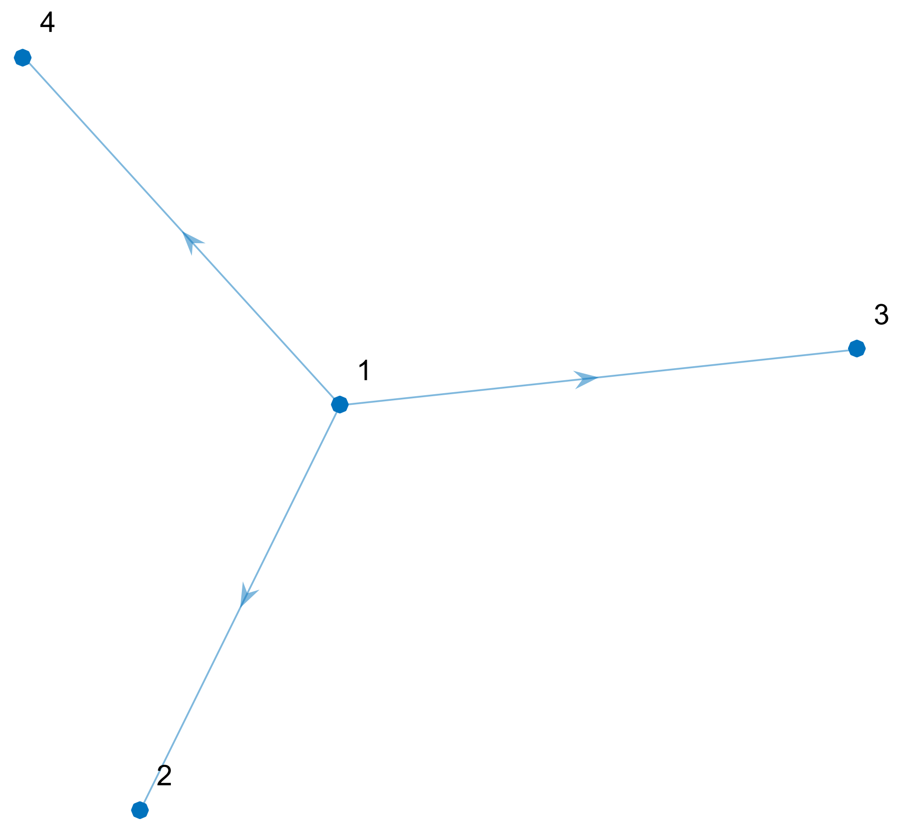
Graphs in MATLAB

A Graph object

- Create
- Manipulate
- Analyze

A GraphPlot object

- View

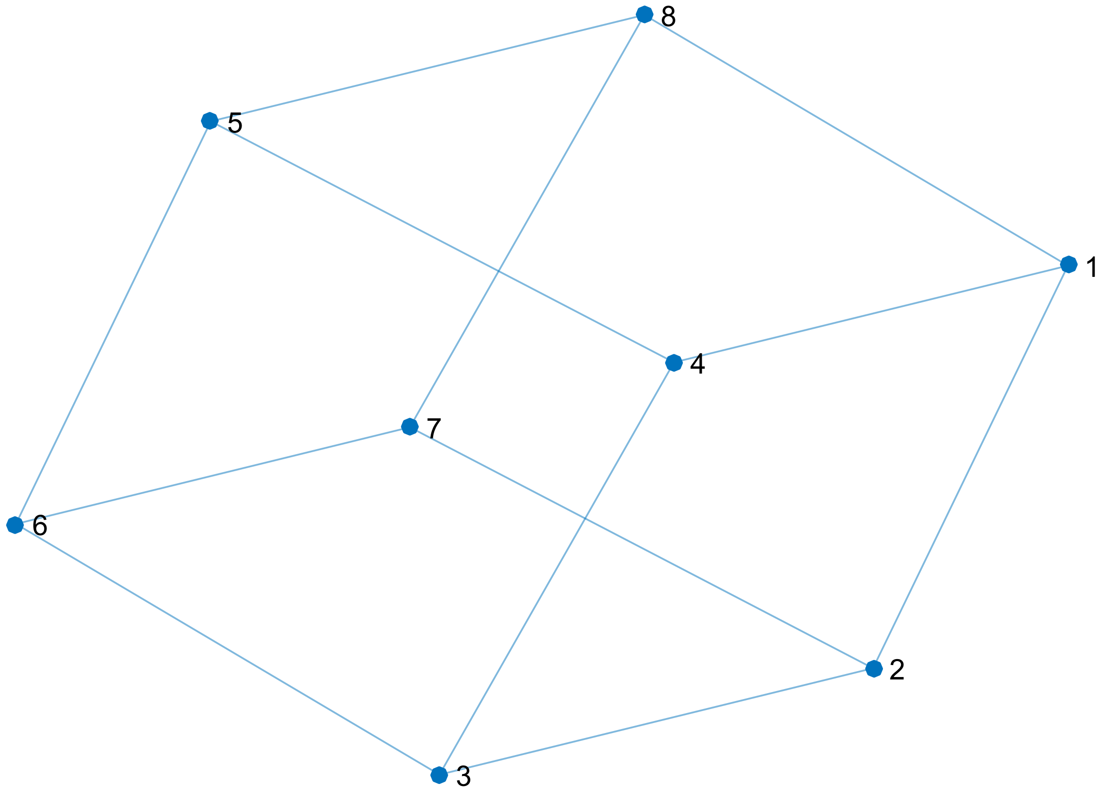


Let’s make a simple Graph

```
sourceNodes = [ 1 1 1 2 2 3 3 4 5 5 6 7 ];
targetNodes = [ 2 4 8 3 7 4 6 5 6 8 7 8 ];
G = graph( sourceNodes , targetNodes )

G =
  graph with properties:
    Edges: [12x1 table]
    Nodes: [8x0 table]
```

Plot a Graph

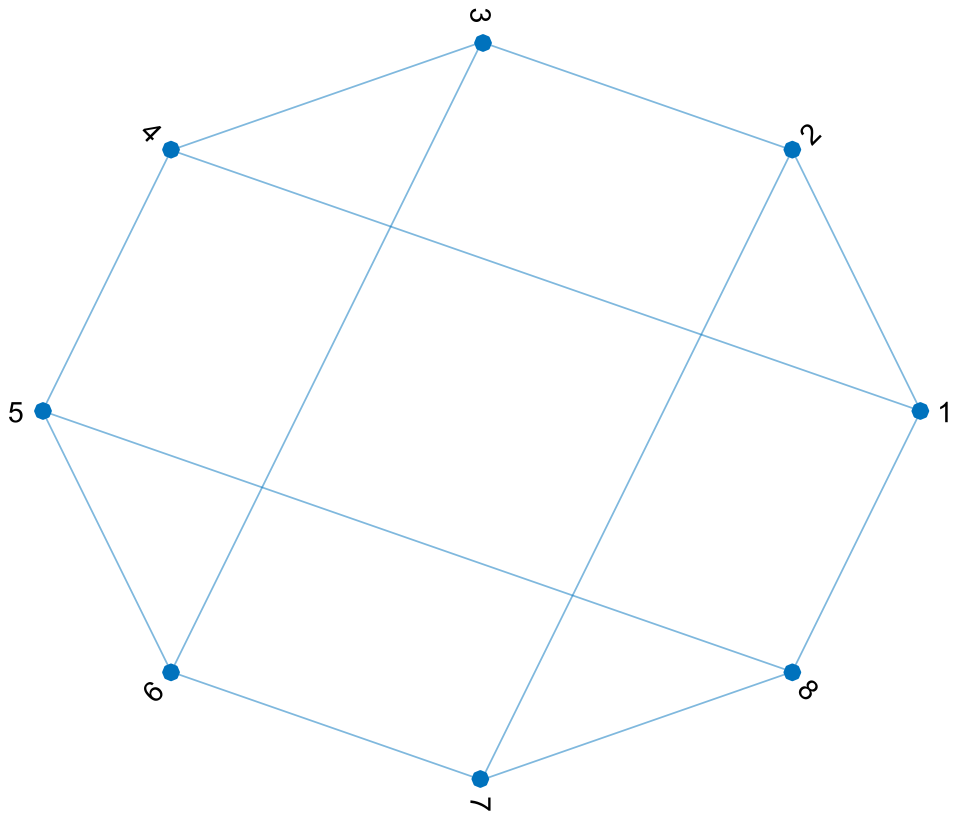
```
P = plot(G);
```



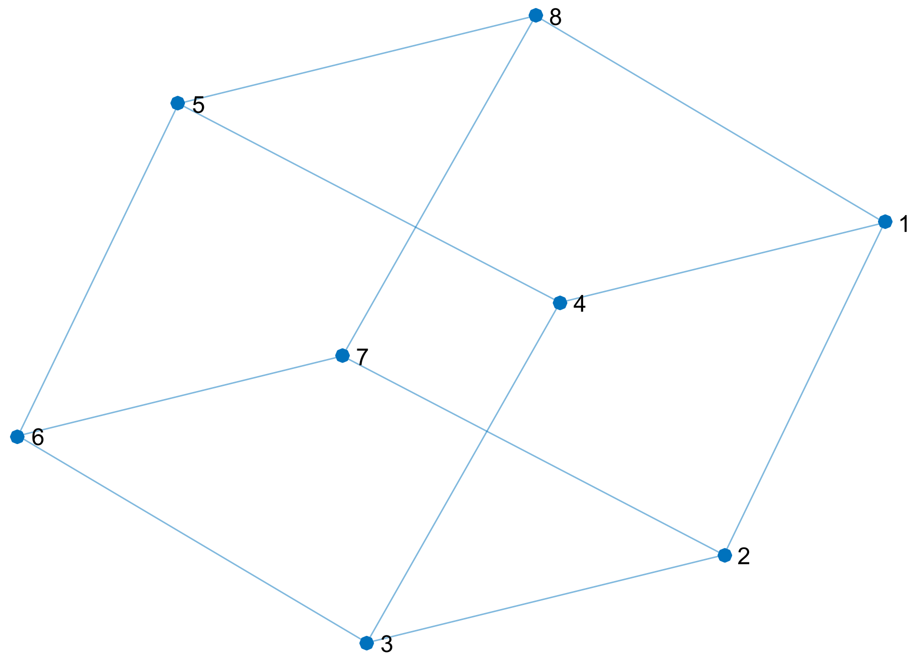

```
sourceNodes = [1 1 1 2 2 3 3 4 5 5 6 7];
targetNodes = [2 4 8 3 7 4 6 5 6 8 7 8];
```

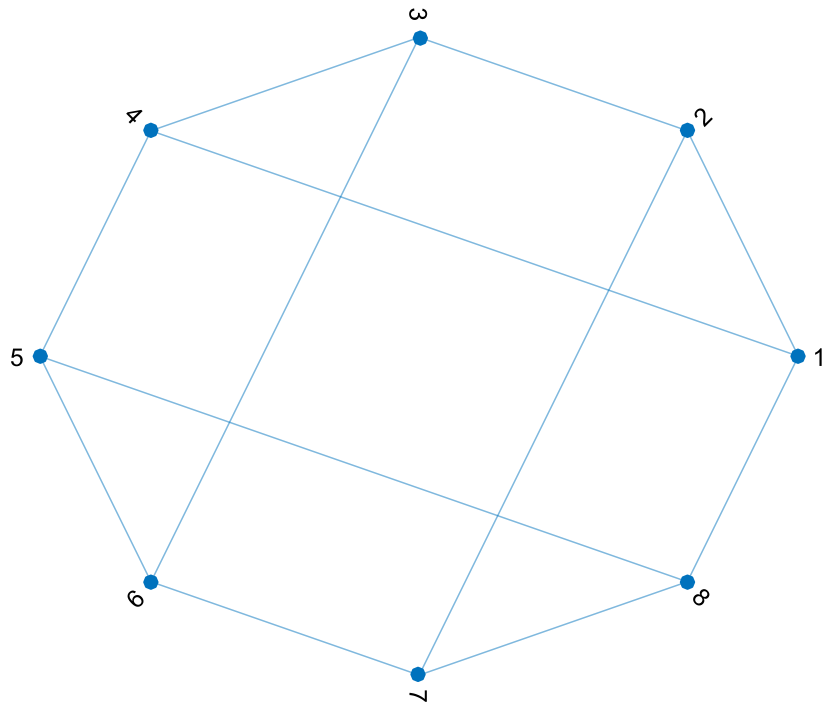
Plot a Graph

```
layout( P,'circle' )
```



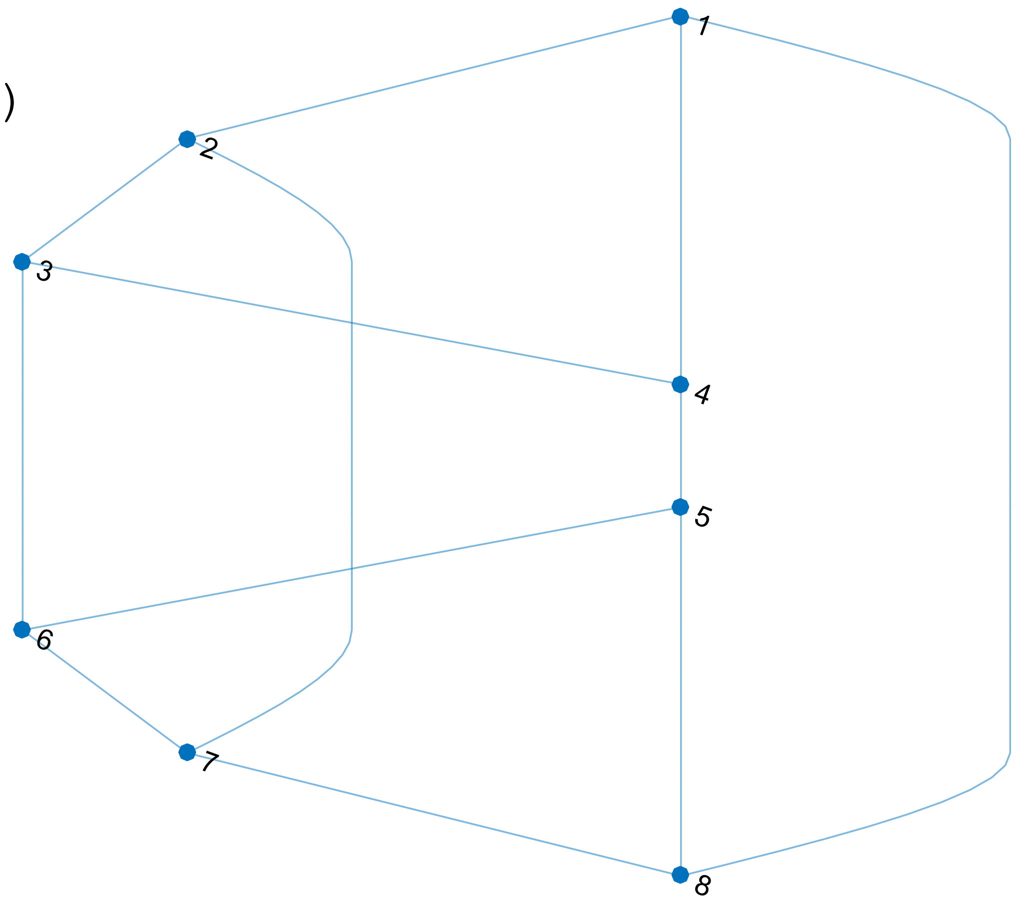
Are these drawings of the same graph?





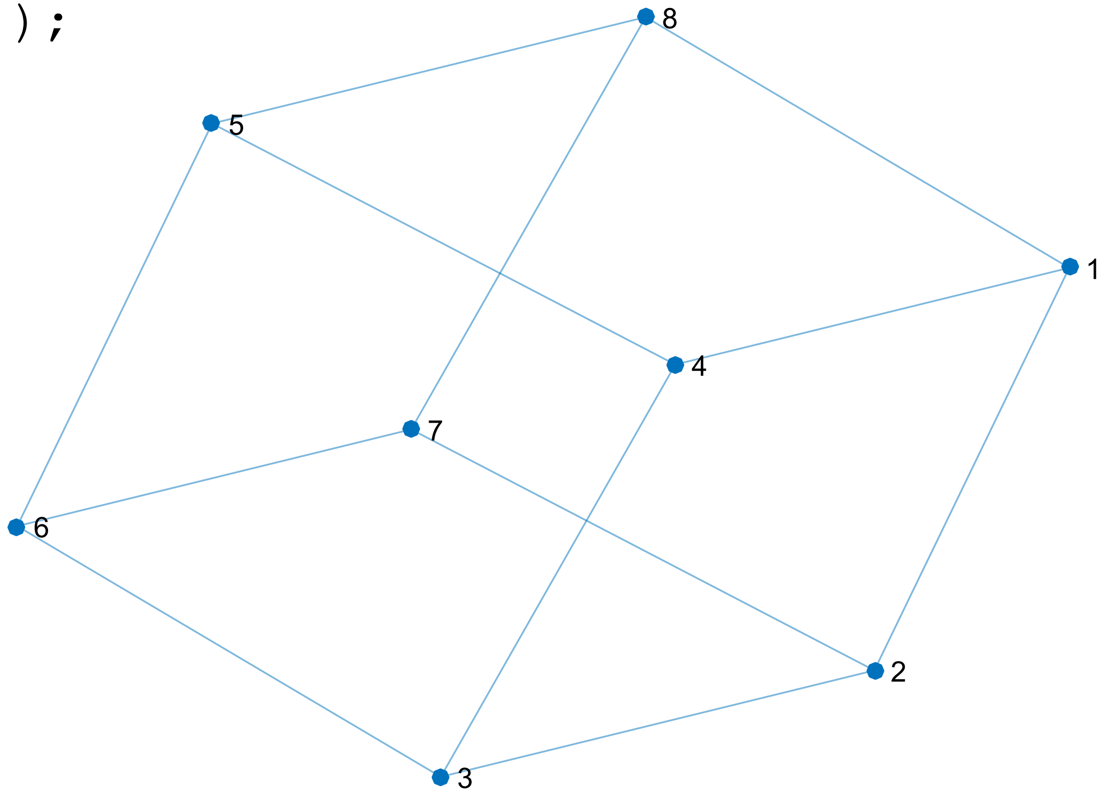
Plot a Graph

```
layout(P, 'layered' )
```



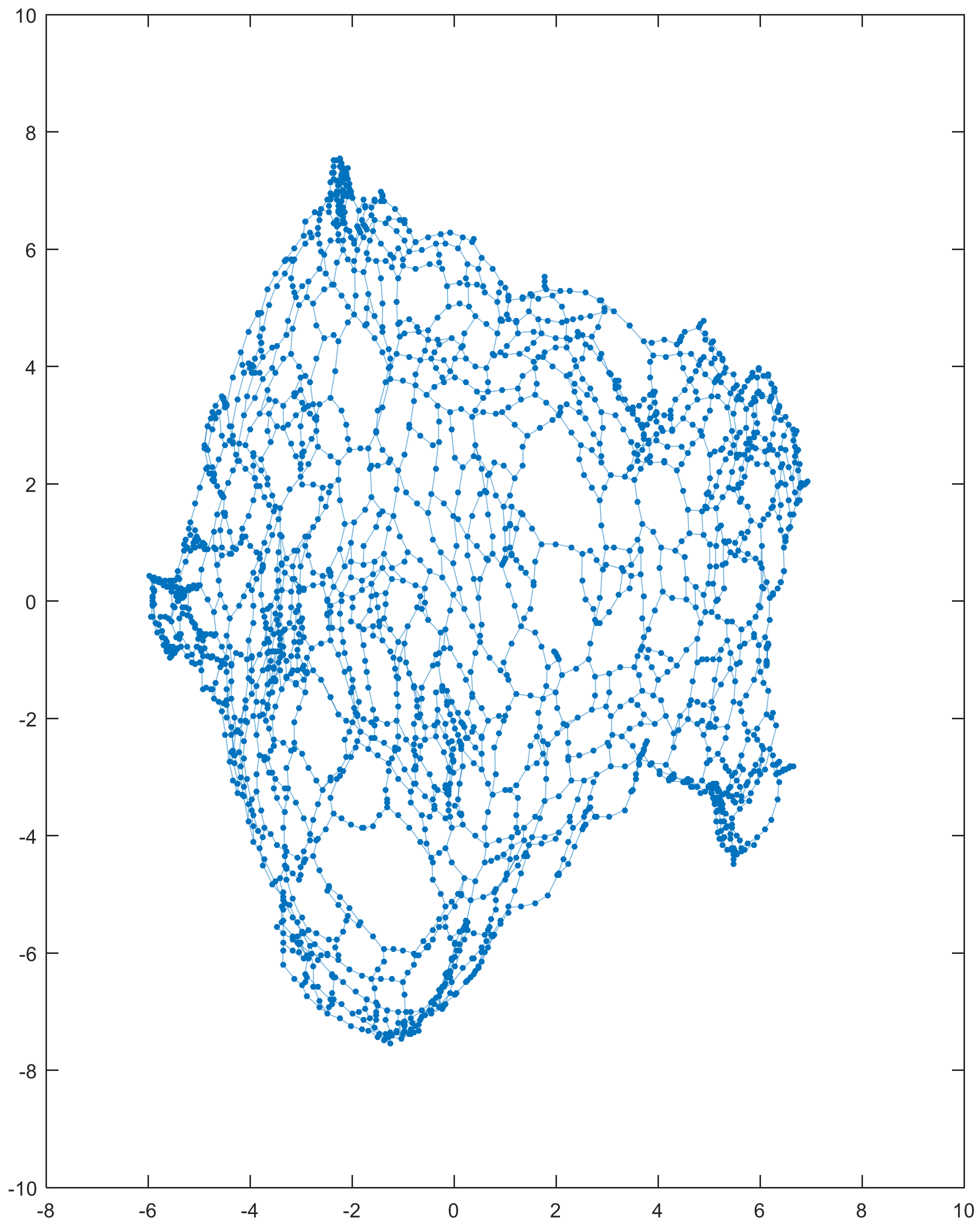
Plot a Graph

```
layout( P, 'force' );
```



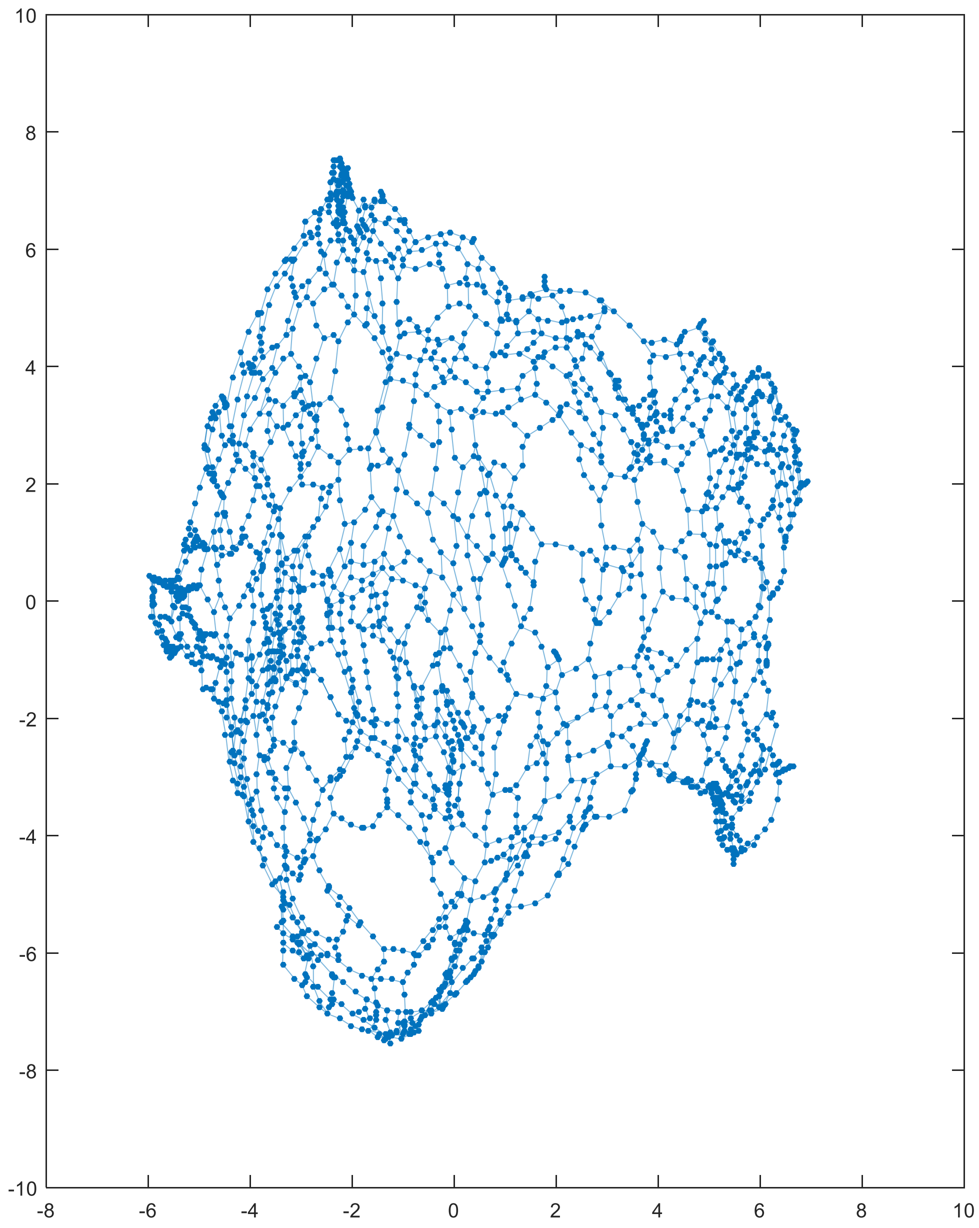
Graphs in MATLAB

```
load('MinnesotaRoads');
plot(G);
```



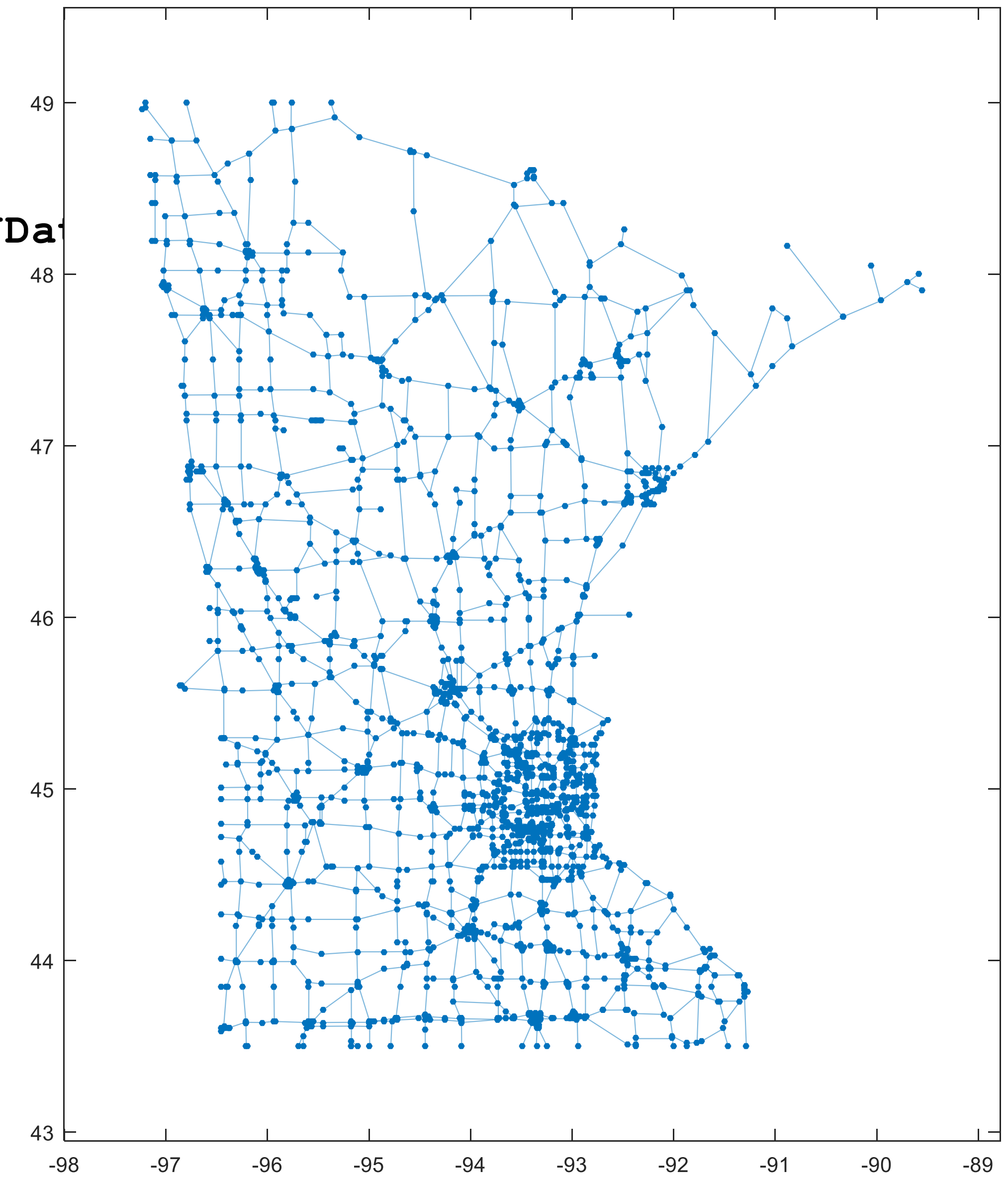
Graphs in MATLAB

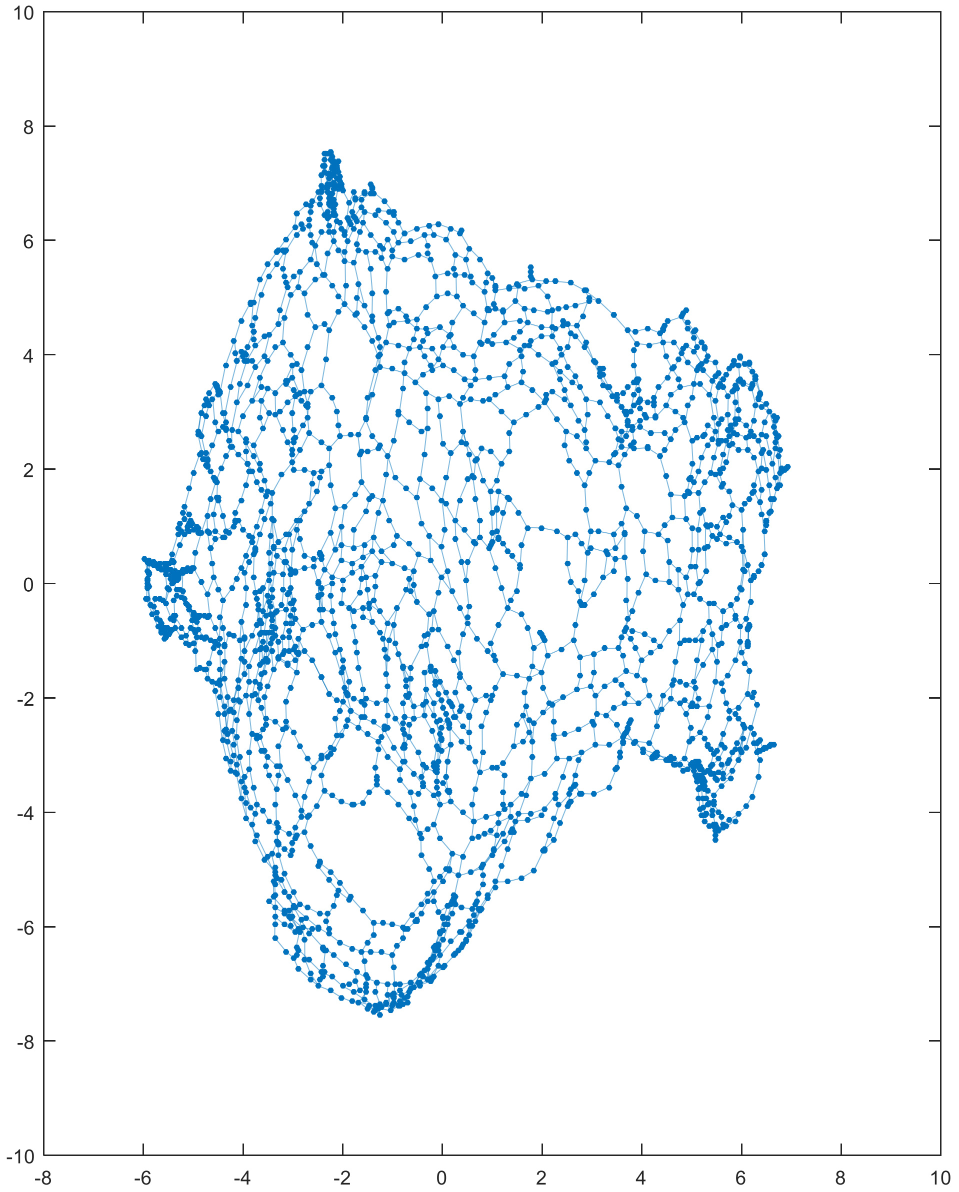
```
G.Nodes( 1:7,: )

ans =

    Longitude    Latitude
    _________    ________
    -97.207      49.001
    -96.801          49
    -95.957          49
    -95.931          49
    -95.766          49
    -95.378      48.999
      -97.2      48.972
```

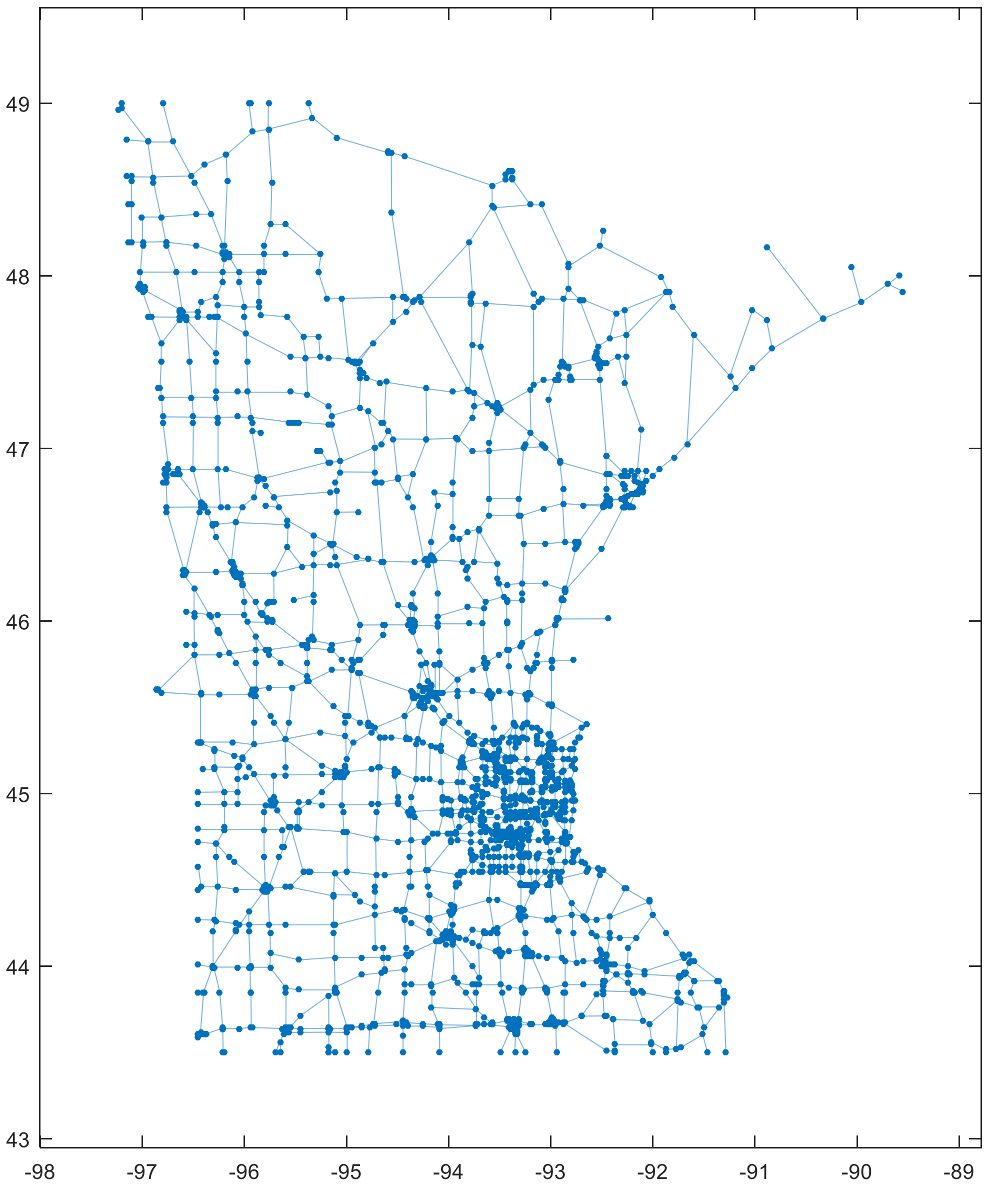


Graphs in MATLAB

```
P = plot(G, 'XData', G.Nodes.Longitude, 'YDat
```




10
8
6
4
2
0
-2
-4
-6
-8
-10
-8
-6
-4
-2
0
2
4
6
8
10

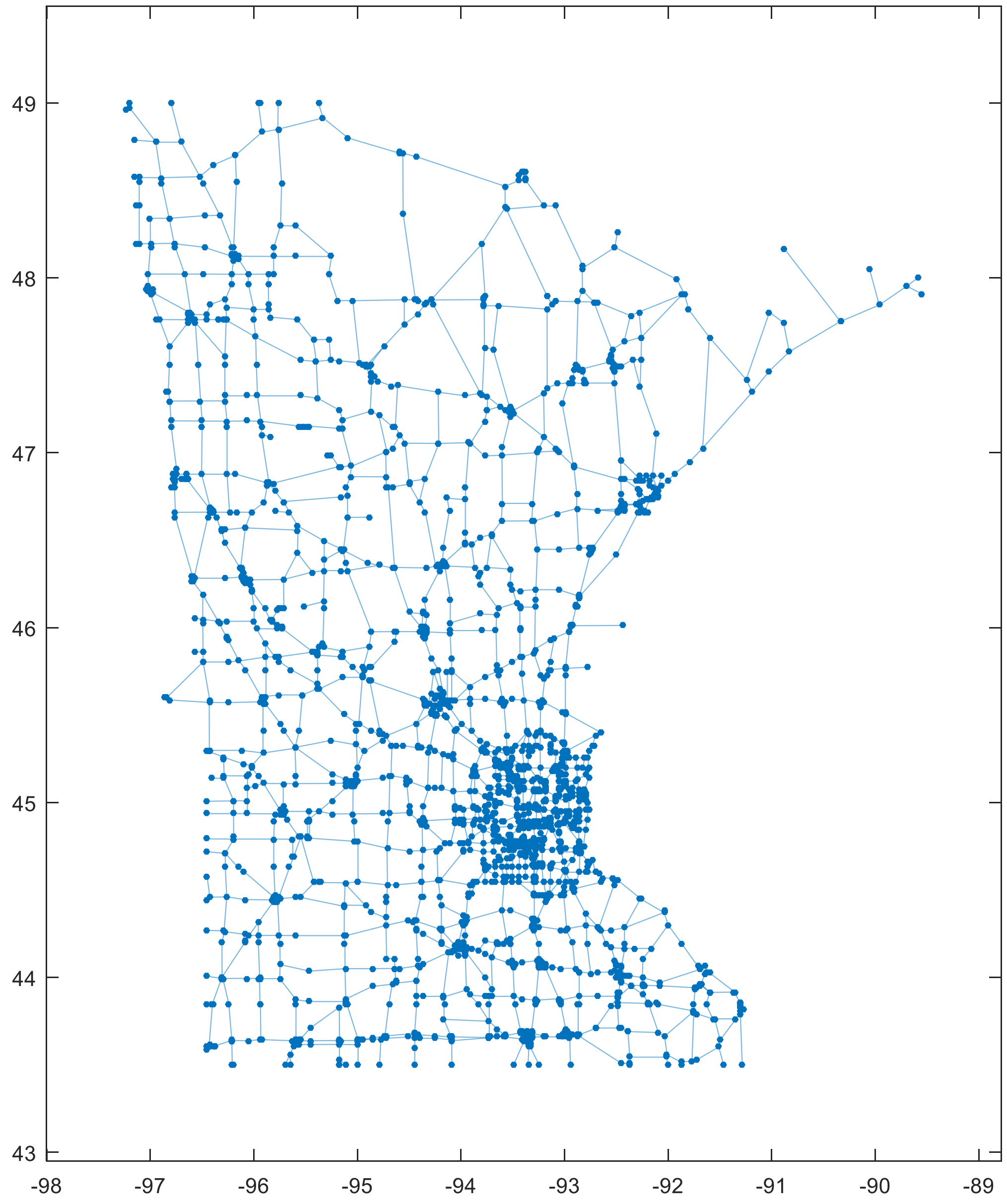

49
48
47
46
45
44
43
-98
-97
-96
-95
-94
-93
-92
-91
-90
-89

Useful Graph Algorithms

shortestpath	Shortest path between two single nodes
shortestpathtree	Shortest path tree from node
distances	Shortest path distances of all node pairs
bfsearch	Breadth-first graph search
dfsearch	Depth-first graph search
maxflow	Maximum flow in graph
conncomp	Connected graph components
minspantree	Minimum spanning tree of graph
toposort	Topological order of directed acyclic graph
isdag	Determine if graph is acyclic
transclosure	Transitive closure
transreduction	Transitive reduction

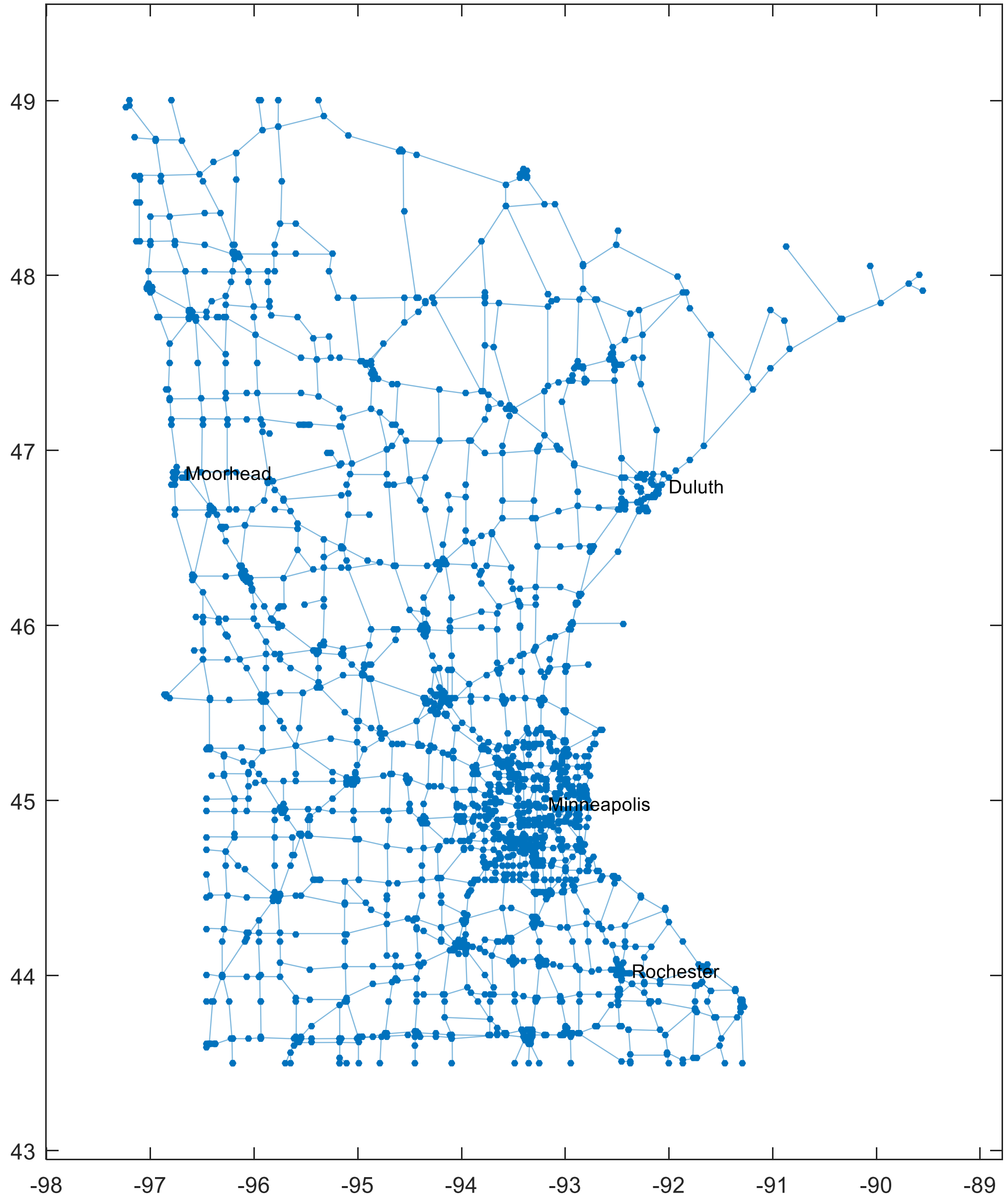
Graphs in MATLAB

```
P.labelnode(cityIDs, cityNames);
```



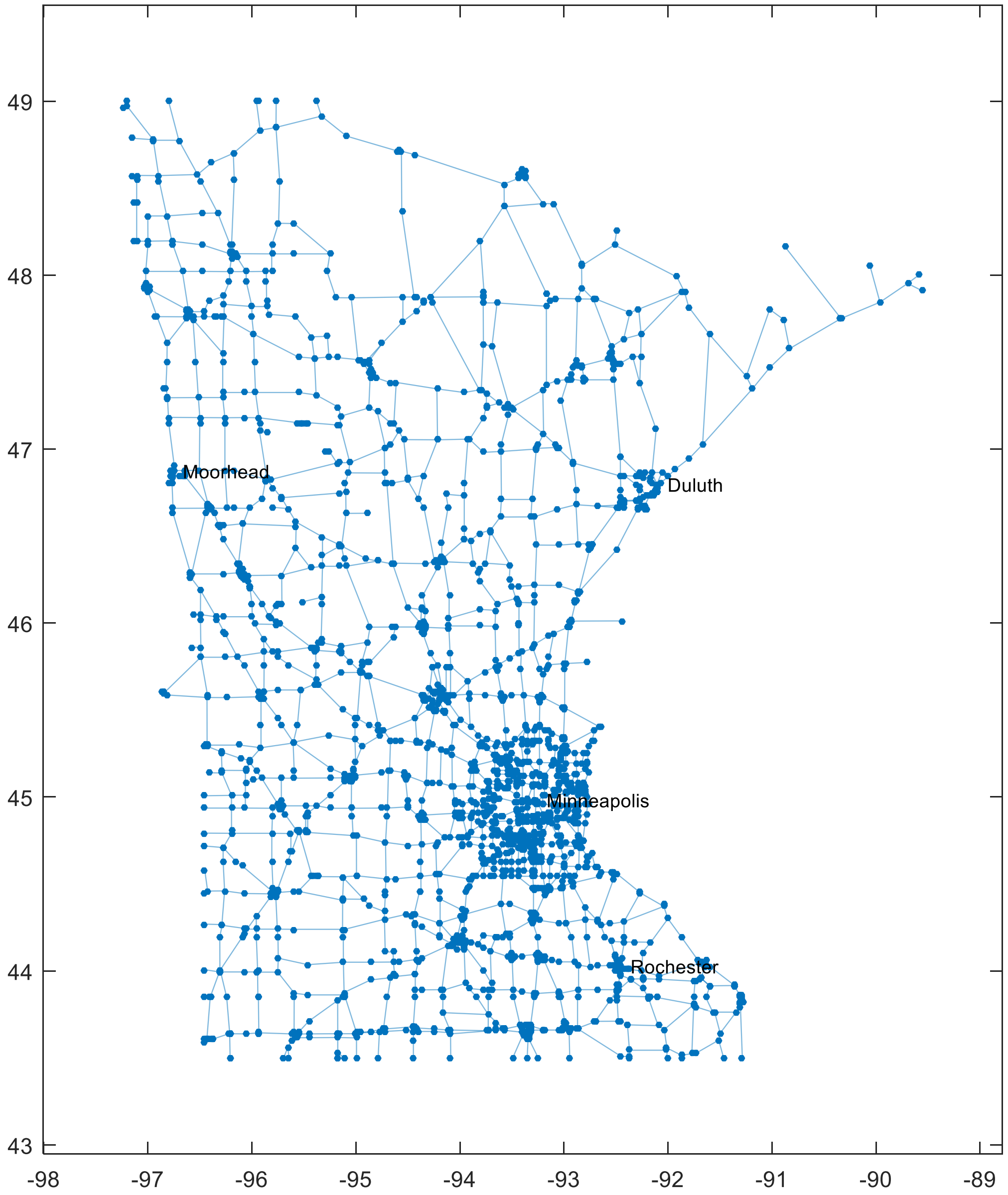
Graphs in MATLAB

```
P.labelnode(cityIDs, cityNames);
```



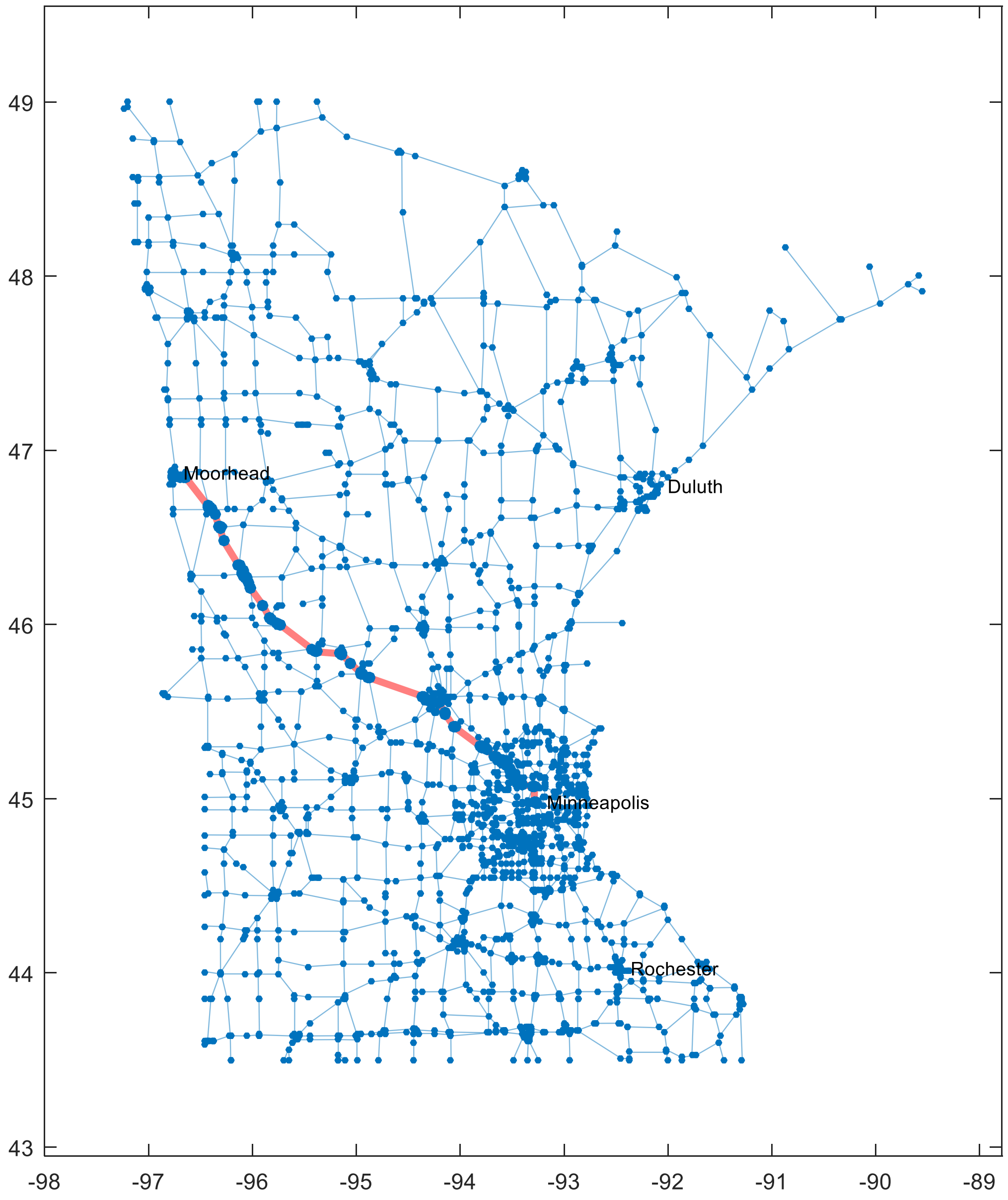
Graphs in MATLAB

```
T = shortestpath(G,Minneapolis,Moorhead);
P.highlight(T,'EdgeColor','r');
```



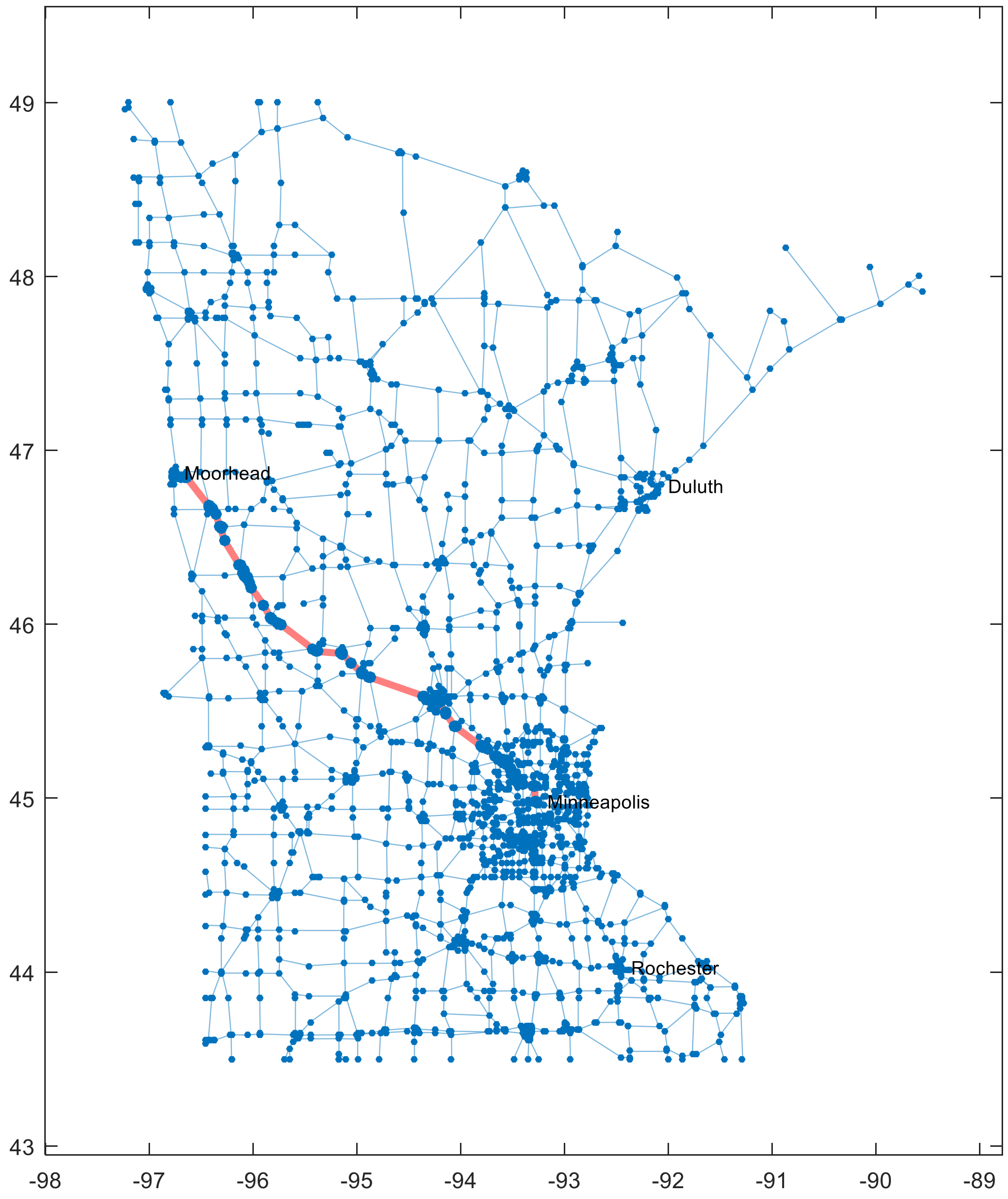
Graphs in MATLAB

```
T = shortestpath(G,Minneapolis,Moorhead);
P.highlight(T,'EdgeColor','r');
```



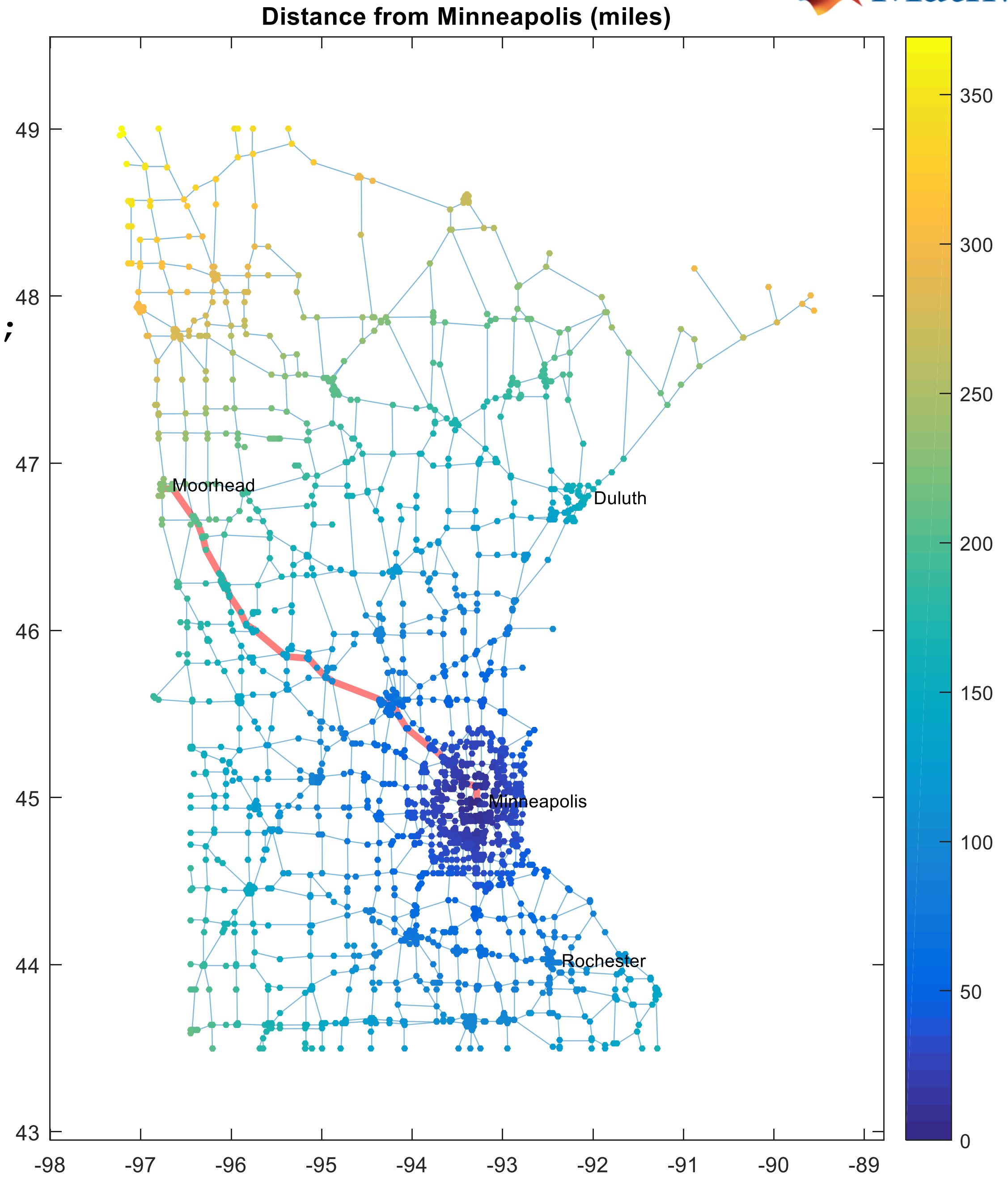
Graphs in MATLAB

```
P.NodeCData = distances(G, Minneapolis);
title('Distance from Minneapolis (miles)');
colorbar
```



Graphs in MATLAB

```
P.NodeCData = distances(G, Minneapolis);
title('Distance from Minneapolis (miles)');
colorbar
```



Minnesota gets a lot of snow.

You plow the snow

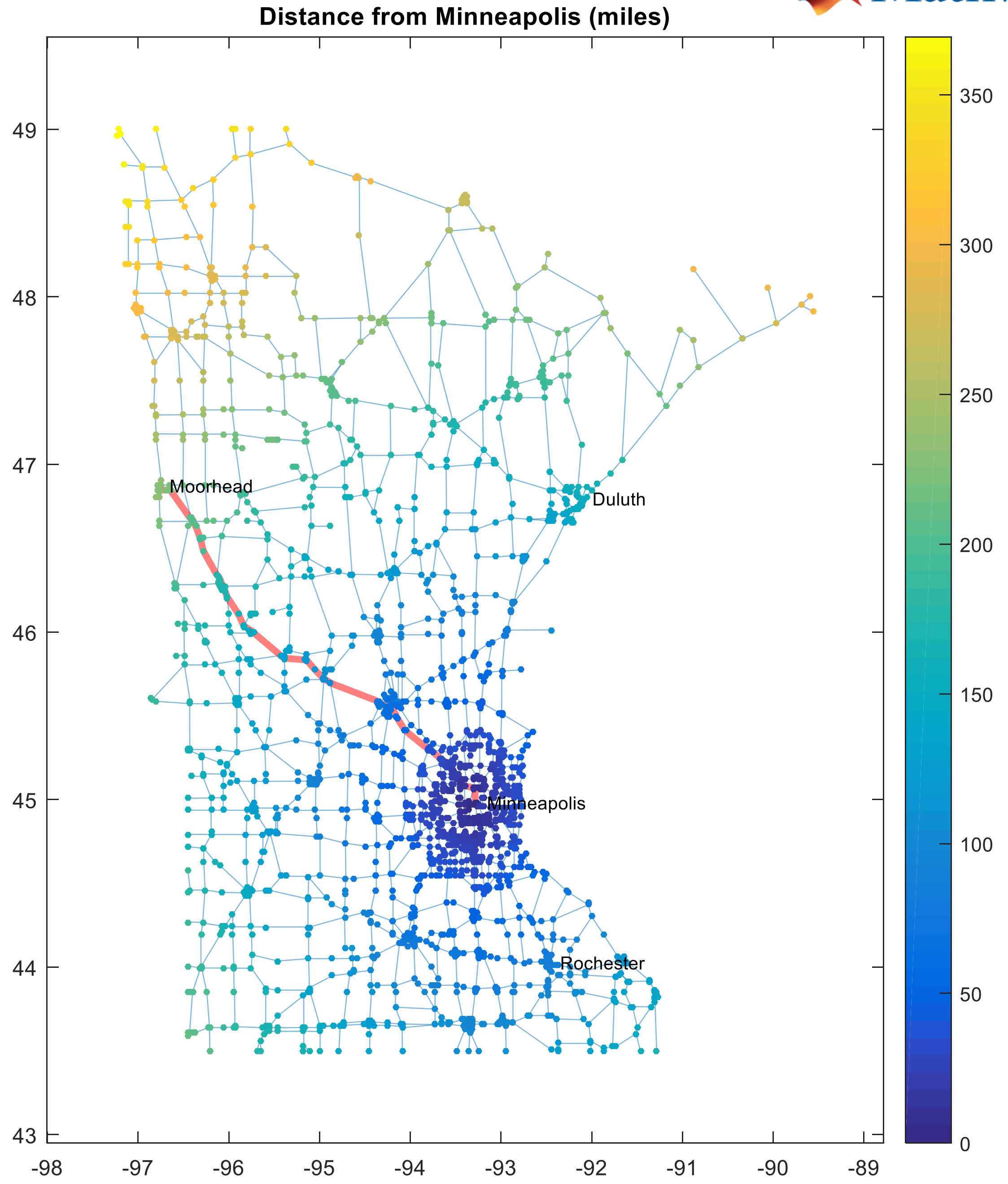
Your equipment is in Minneapolis

You don't have to plow every road

Drivers must be able to get from every town to every other town

What is the least you must plow?

```
tree = minspantree(G,'root',minneapolis);
highlight(P,tree, 'LineWidth', 3);
```



Minnesota gets a lot of snow.

You plow the snow

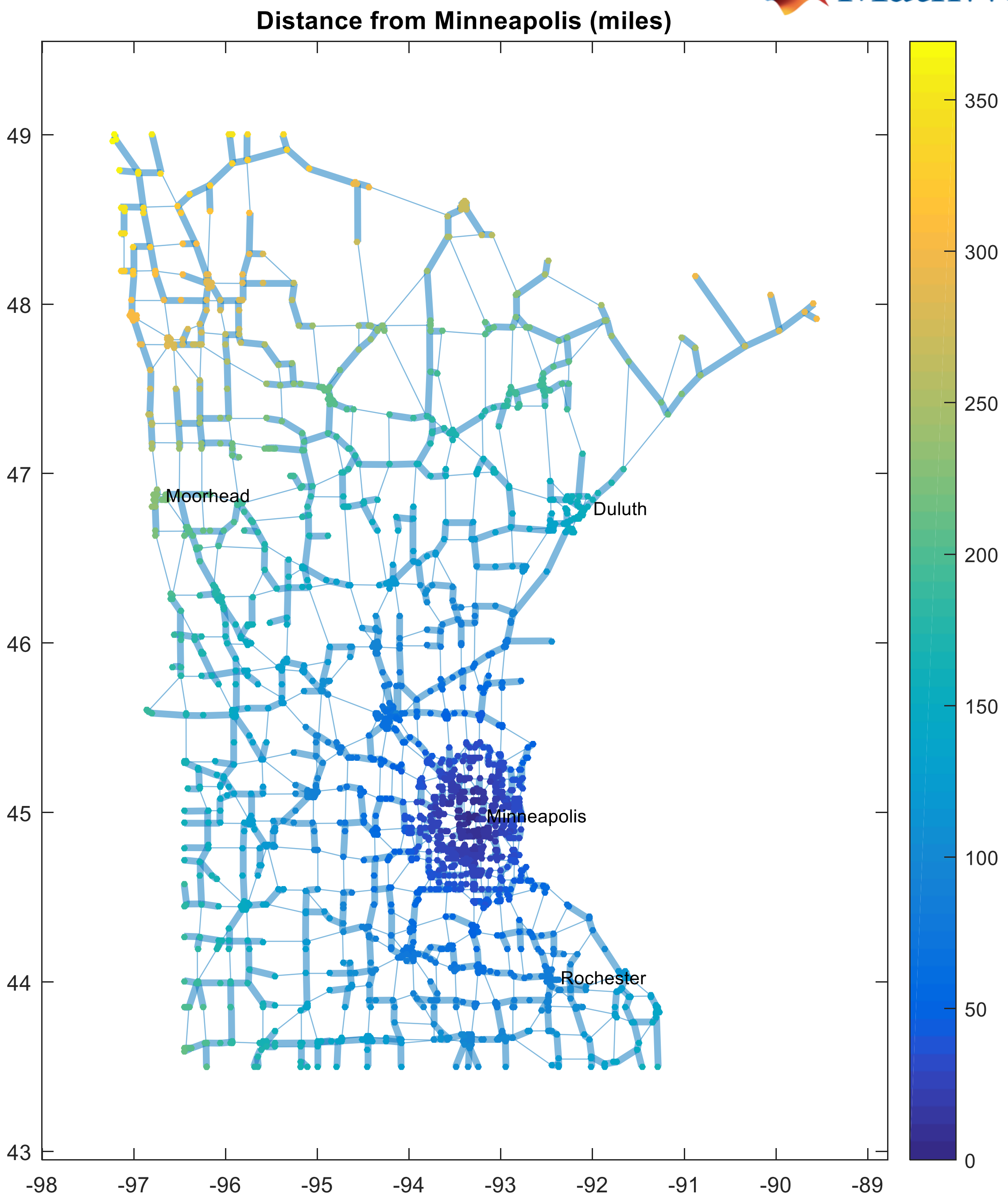
Your equipment is in Minneapolis

You don't have to plow every road

Drivers must be able to get from every town to every other town

What is the least you must plow?

```
tree = minspantree(G,'root',minneapolis);
highlight(P,tree, 'LineWidth', 3);
```

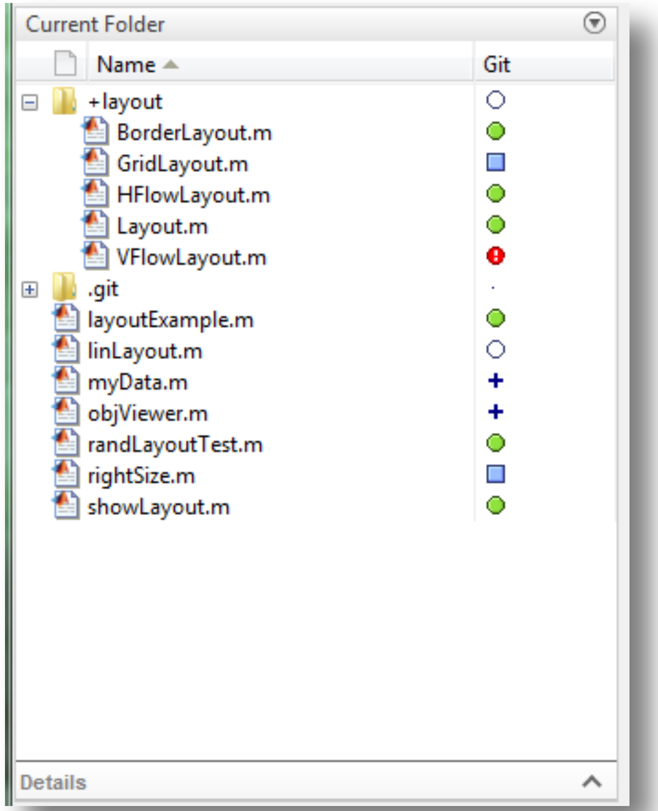


Agenda

- MATLAB Infrastructure
  - Editor
  - Graphics
- Workflows
  - Managing / Testing Code
  - Sharing Apps and Custom Toolboxes
- Performance
  - Acceleration Strategy
  - Execution Engine
  - Parallel computing and GPU computation
- Wrap up & QnA

Source Control Integration

- Manage your code from within the MATLAB Desktop
- Leverage modern source control capabilities
  - GIT and Subversion integration in Current Folder browser
- Use Comparison Tool to view and merge changes between revisions



Unit Testing Framework

- Write, run, and analyze tests for your MATLAB programs
  - Define how each test checks values and responds to failures
  - Setup and restore system before and after tests
  - Run tests individually or grouped into a test suite
  - Measure MATLAB code performance

```
s\MATLABFiles\Test1.m
lassdef Test1 < matlab.unittest.TestCase

  methods (Test)

      function testRealSolution(testCase)
          actSolution = quadraticSolver(1
          expSolution = [2,1];
          testCase.verifyEqual(actSolutic
      end
```

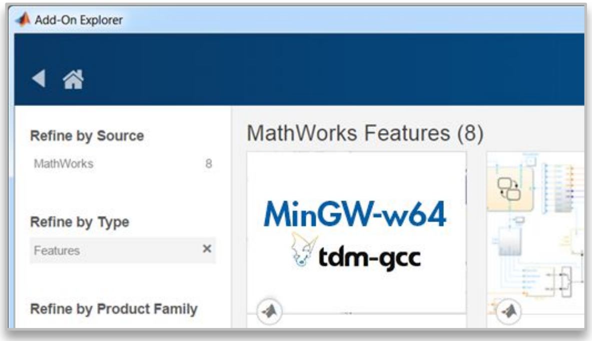
- Supports either script-based, function-based or object-based unit tests

Why use Unit Testing?

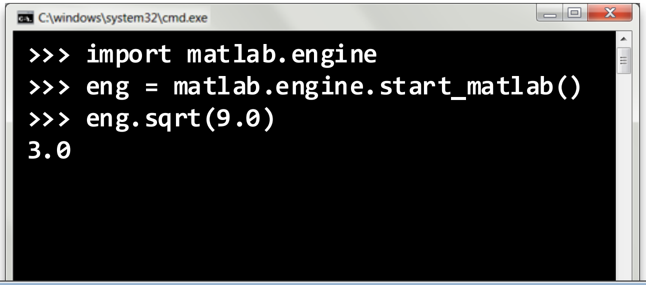
- Testing saves development time
- Testing makes development more enjoyable
  - Your time is spent making things, not fixing things.
  - Fewer nasty surprises and opportunities to make mistakes
- Framework is not trivial, but easily learnable
  - Well worth the effort if you maintain software.

Enhancements to MATLAB Interoperability

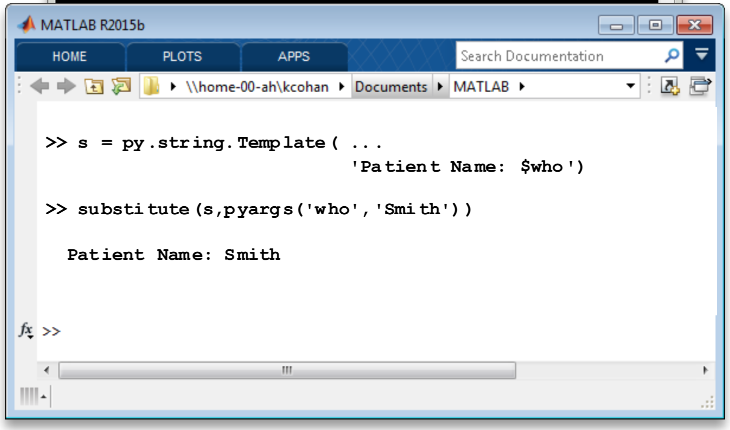
- MEX compiler support
  - Access to a free compiler (**MinGW-w64**) for 64-bit Windows *(from the Add-On Explorer)*



- MATLAB Engine API
  (for calling MATLAB from Python)
  - Call MATLAB functions and objects from Python by connecting to a running session of MATLAB

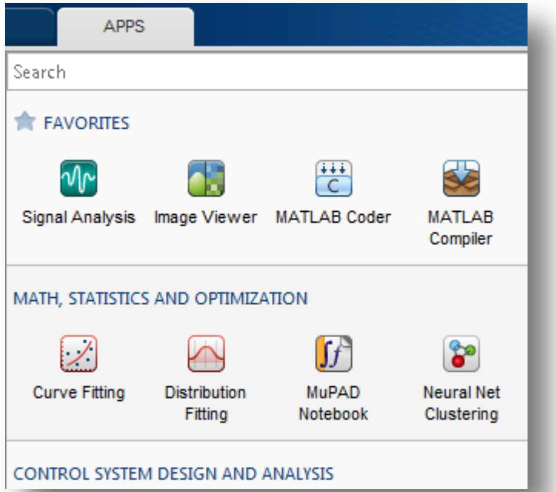


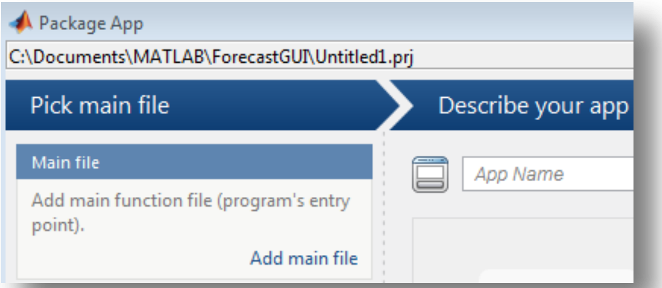
- MATLAB interface to Python
  (for calling Python from MATLAB)
  - Clear Python class definitions with `clear classes` command
    (useful when reloading revised Python classes)



MATLAB Apps

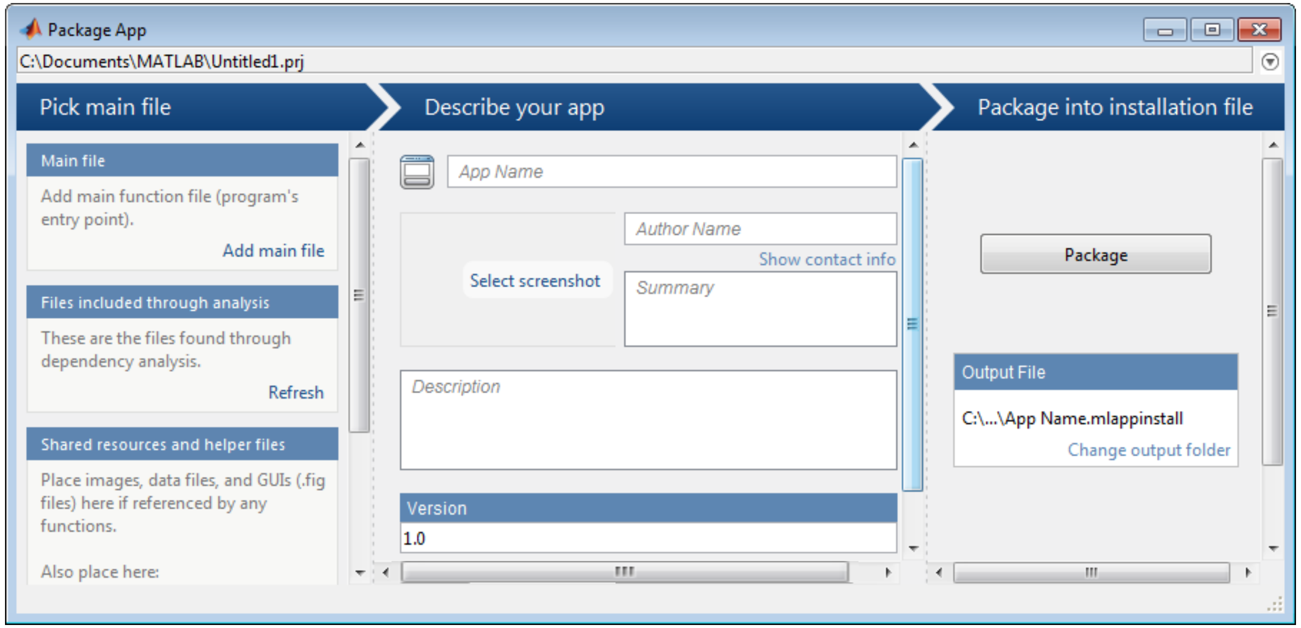
- Apps are self-contained tools, typically with a UI
  - Accessed in MATLAB Apps gallery
  - Included in many MATLAB Products
  - Can be authored by MATLAB users
- Apps from the MATLAB Community
  - Found on MATLAB File Exchange
  - Download and install into the MATLAB Apps gallery
- Making your own apps
  - Create single file for easier install and distribution





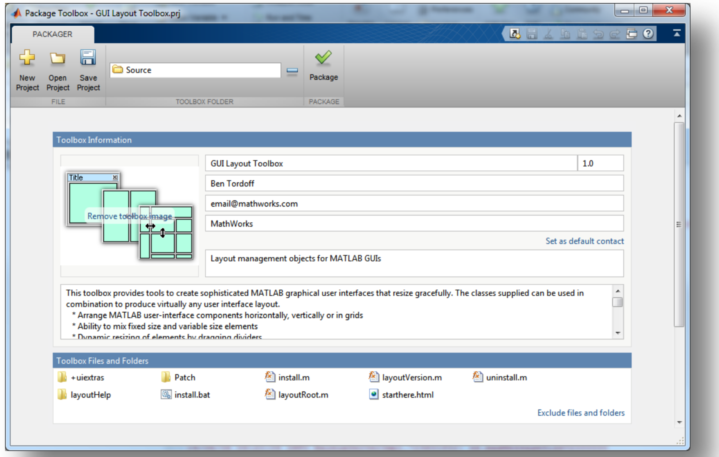
Packaging and Sharing MATLAB Apps

- Automatically includes all necessary files
- Documents required products
- Creates single installation file for easy distribution and installation into the MATLAB apps gallery

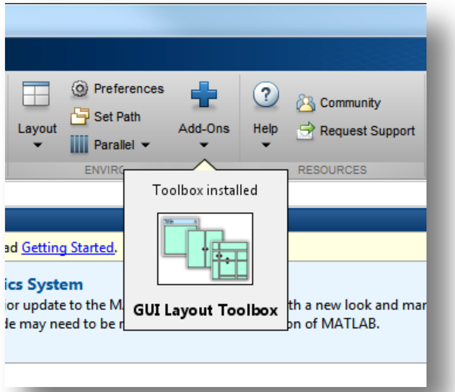


Toolbox Packaging

- Package your toolbox as a single installer file
  - Contains all of the code, data, apps, documentation, and examples
  - Checks for dependent files and automatically includes them
  - Documents required products



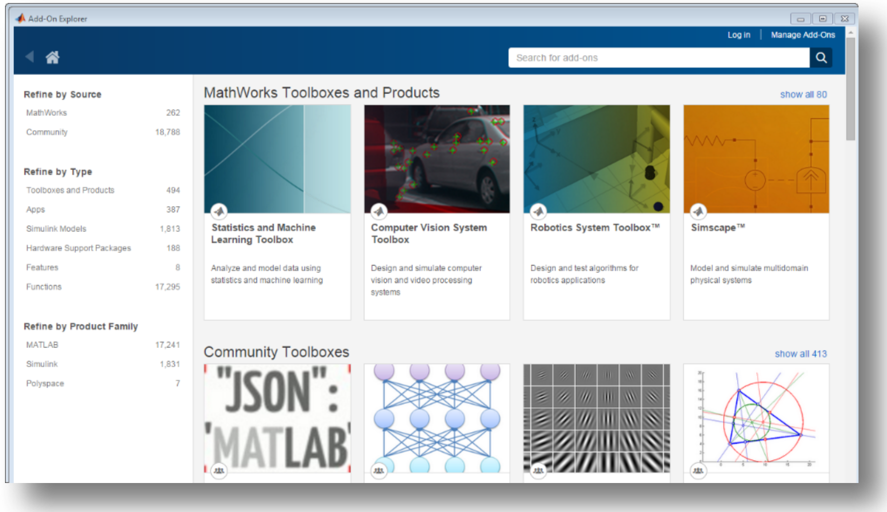
- Included folders and files automatically appear on path when installed



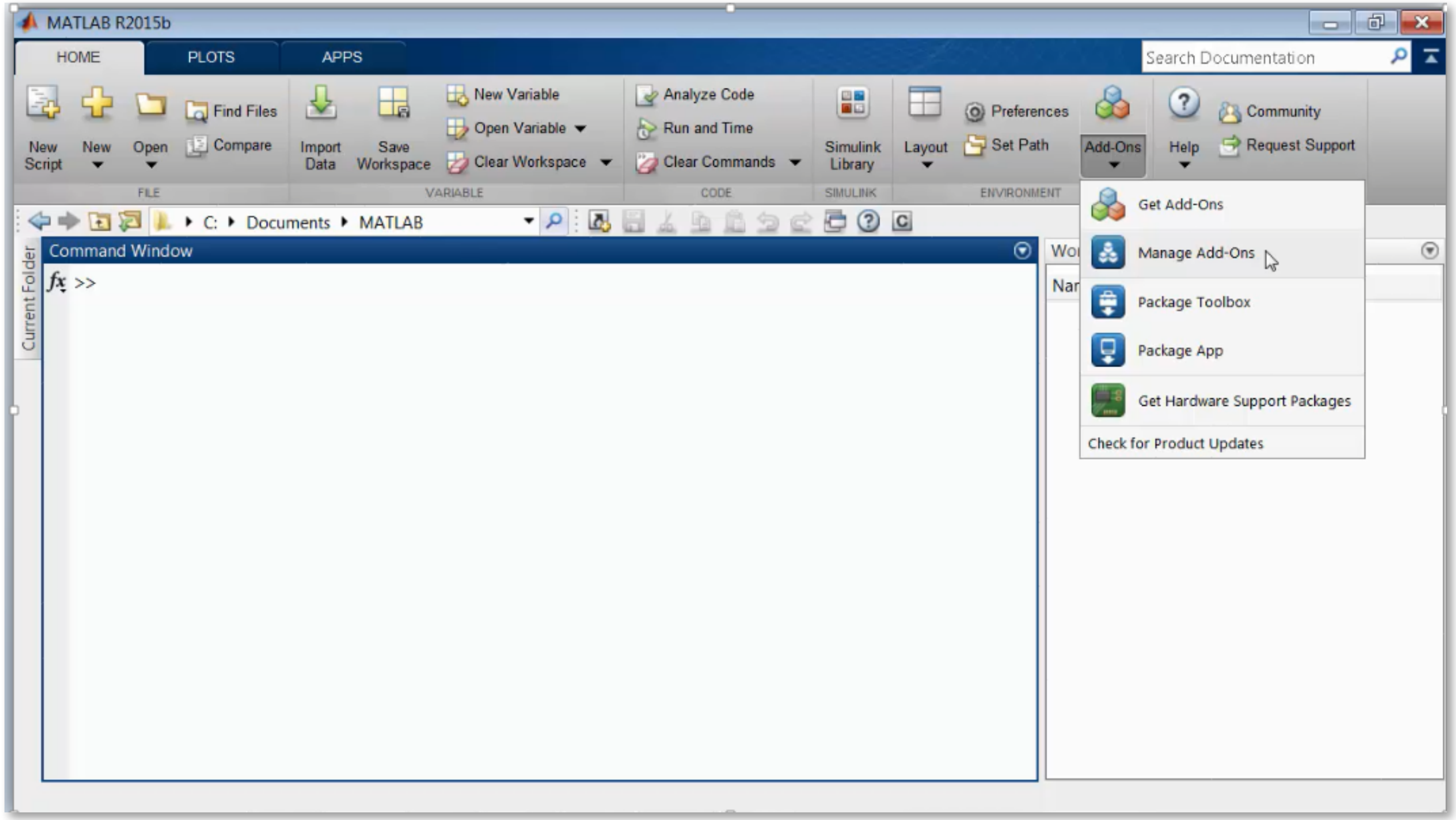
- View details and uninstall toolboxes with Manage Add-on Toolboxes dialog box

Add-On Explorer

- Add capabilities to MATLAB, including **community-authored** and **MathWorks** toolboxes, apps, functions, models, and hardware support
  - Browse and install add-ons directly from MATLAB
  - Access **community-authored** content from File Exchange



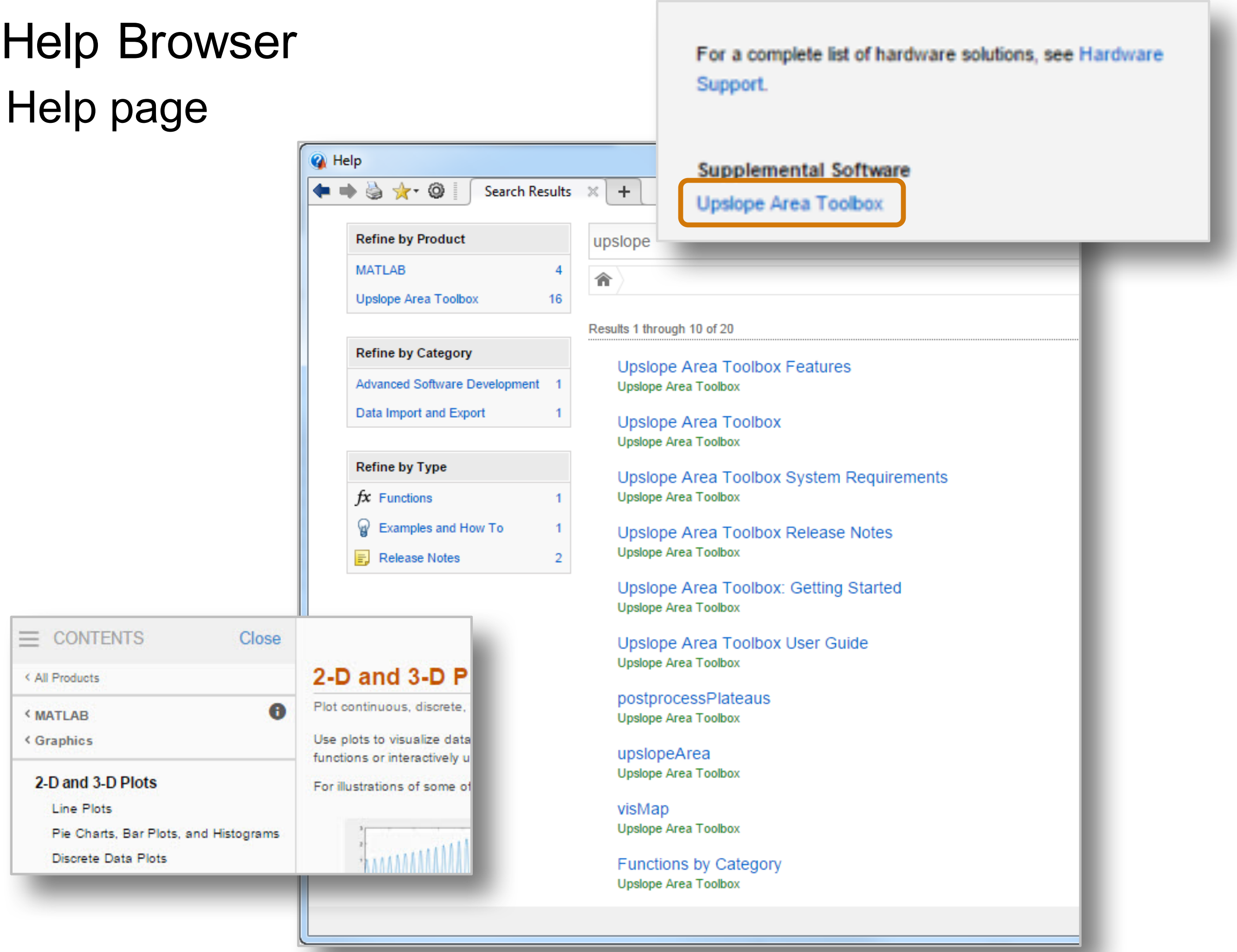
Add-On Explorer



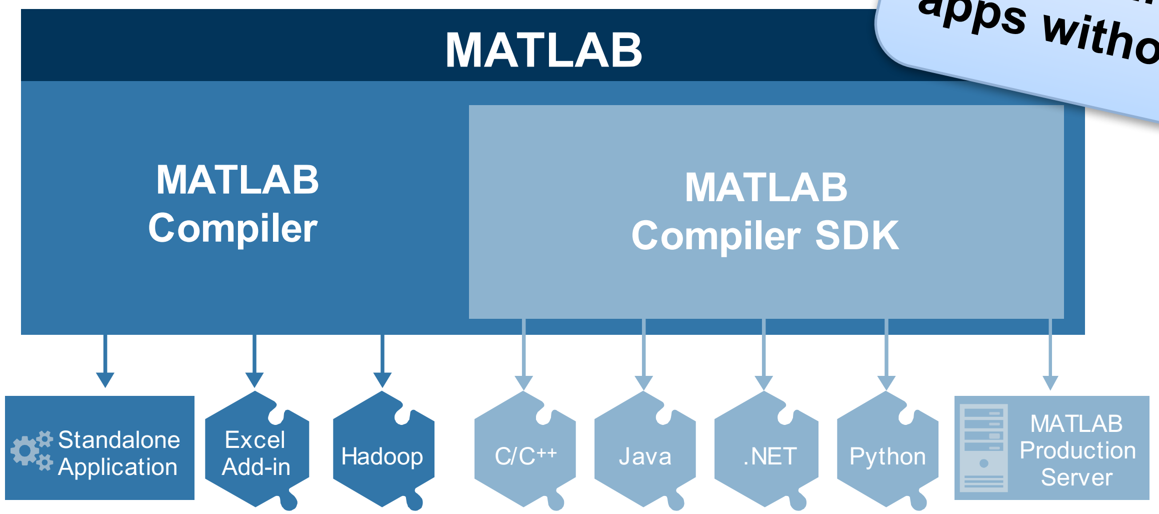
MATLAB

Documentation

- Integration of documentation for custom toolboxes into the MATLAB Help Browser
  - Link appears on the Home Help page
  - Help displays in the current window
  - Integrated search
- Redesigned help navigation



Application Deployment



Smaller MATLAB Runtime in R2016a for apps without graphics

- MATLAB Compiler
  - Application-specific MATLAB Runtime based on requirements for numeric, graphic, and GPU support
  - Support for MATLAB objects for Hadoop integration
- MATLAB Compiler SDK
  - Development and test framework for MATLAB Production Server for integration with web and enterprise systems

Agenda

- MATLAB Infrastructure
  - Editor
  - Graphics
- Workflows
  - Managing / Testing Code
  - Sharing Apps and Custom Toolboxes
- Performance
  - Acceleration Strategy
  - Execution Engine
  - Parallel computing and GPU computation
- Wrap up & QnA

Performance Updates in MATLAB & Toolboxes

- MATLAB
  - `median, cumsum, cumprod, cummin, cummax`
- Image Processing Toolbox
  - Image filtering and grayscale morphology
- Optimization Toolbox
  - `fminunc, fsolve, lsqcurvefit, lsqnonlin` (using Parallel Computing Toolbox)
- Database Toolbox
  - `fetch` – faster database read and write
  - Native SQLite interface

Performance Updates in MATLAB & Toolboxes

Statistics and Machine Learning Toolbox

- clustering using `kmeans`, `kmedoids`, and Gaussian mixture models faster when data has a large number of clusters
- Stable Distributions
  - Model financial and other data that requires heavy-tailed distributions
- Half-Normal Distributions
  - Model truncated data and create half-normal probability plots
- Linear Regression: `CompactLinearModel` object reduces memory footprint of linear regression model
- Robust covariance estimation for multivariate sample data using `robustcov`
- Squared Euclidean distance measure for `pdist` and `pdist2` functions
- Nearest neighbor search using kd-tree
- GPU support for extreme value distribution functions and `kmeans`
- Probability Distributions
- Fit kernel smoothing density to multivariate data using the `ksdensity` and `mvksdensity` functions

Performance Updates in MATLAB & Toolboxes

- GPU acceleration using Parallel Computing Toolbox
  - More than 90 GPU-enabled functions in Statistics and Machine Learning Toolbox, including:
    - Probability distributions
    - Descriptive statistics
    - Hypothesis testing
  - An additional 16 MATLAB functions supported using `gpuArray`
  - An additional 23 MATLAB functions supported using sparse `gpuArray`

MATLAB Execution Engine

Old system had two different execution mechanisms – a JIT and an Interpreter.
New system has a single execution mechanism.

Old JIT was designed for FORTRAN-like constructs within MATLAB.
New JIT is designed for the entire MATLAB language.

Old system had a monolithic architecture that was difficult to extend.
New system has a Modular, Thread-safe, and Platform re-targetable architecture.

MATLAB Execution Engine

Performance Improvement Highlights

Econometrics Toolbox: American Basket Demo executes **60% faster**

Image processing with active contours executes **32% faster**

SVM classification for Machine Learning executes **12% faster**

Examples used in “Speeding up MATLAB” webinar execute **30% faster**

k-NN classification for Machine Learning executes **37% faster**

Machine Learning classification executes **25% faster**

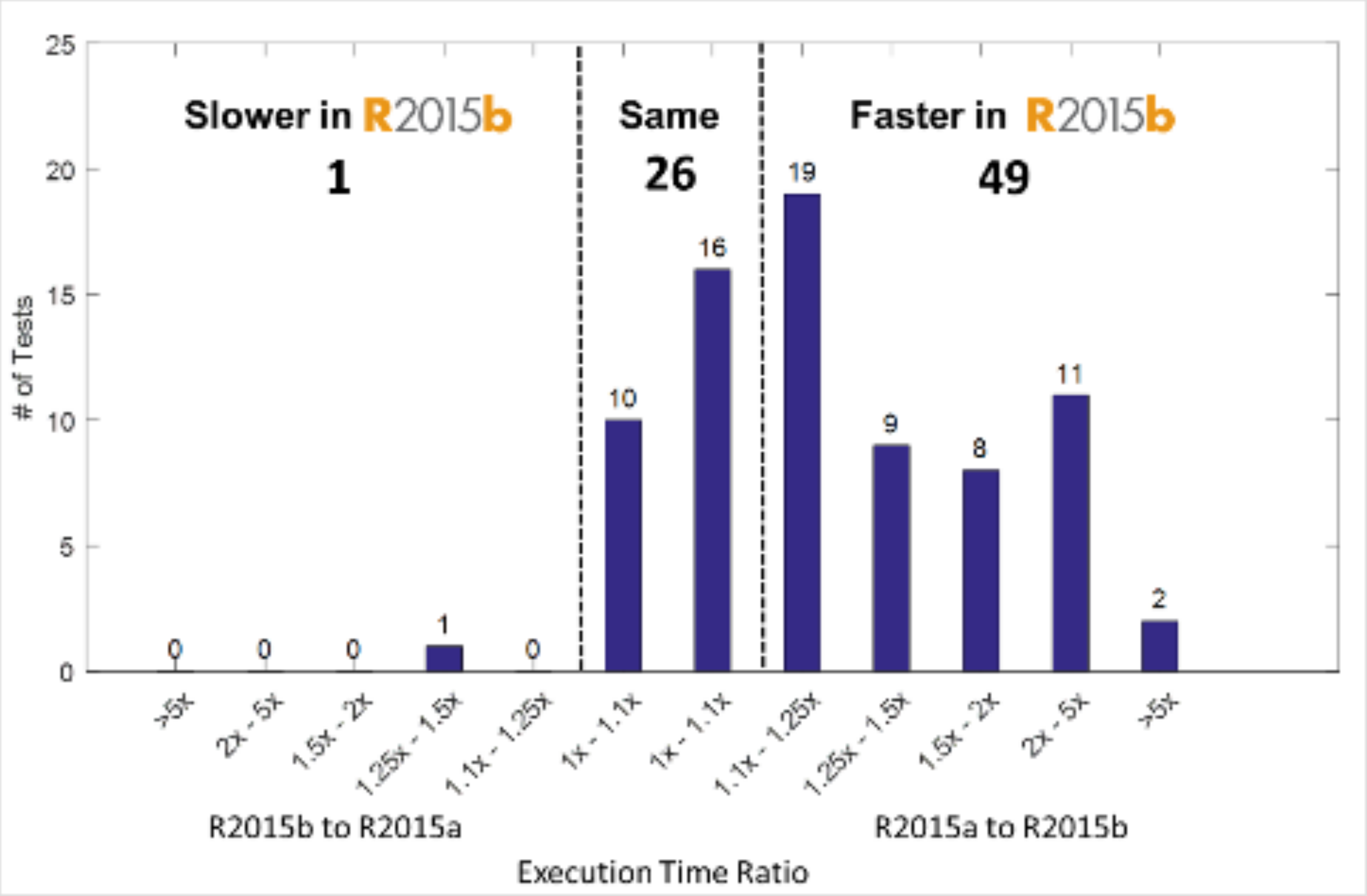
Image Processing executes **15% faster**

Performance in Object-Oriented MATLAB Code on File Exchange executes **10-40% faster**

Wireless Application demo executes **50% faster**

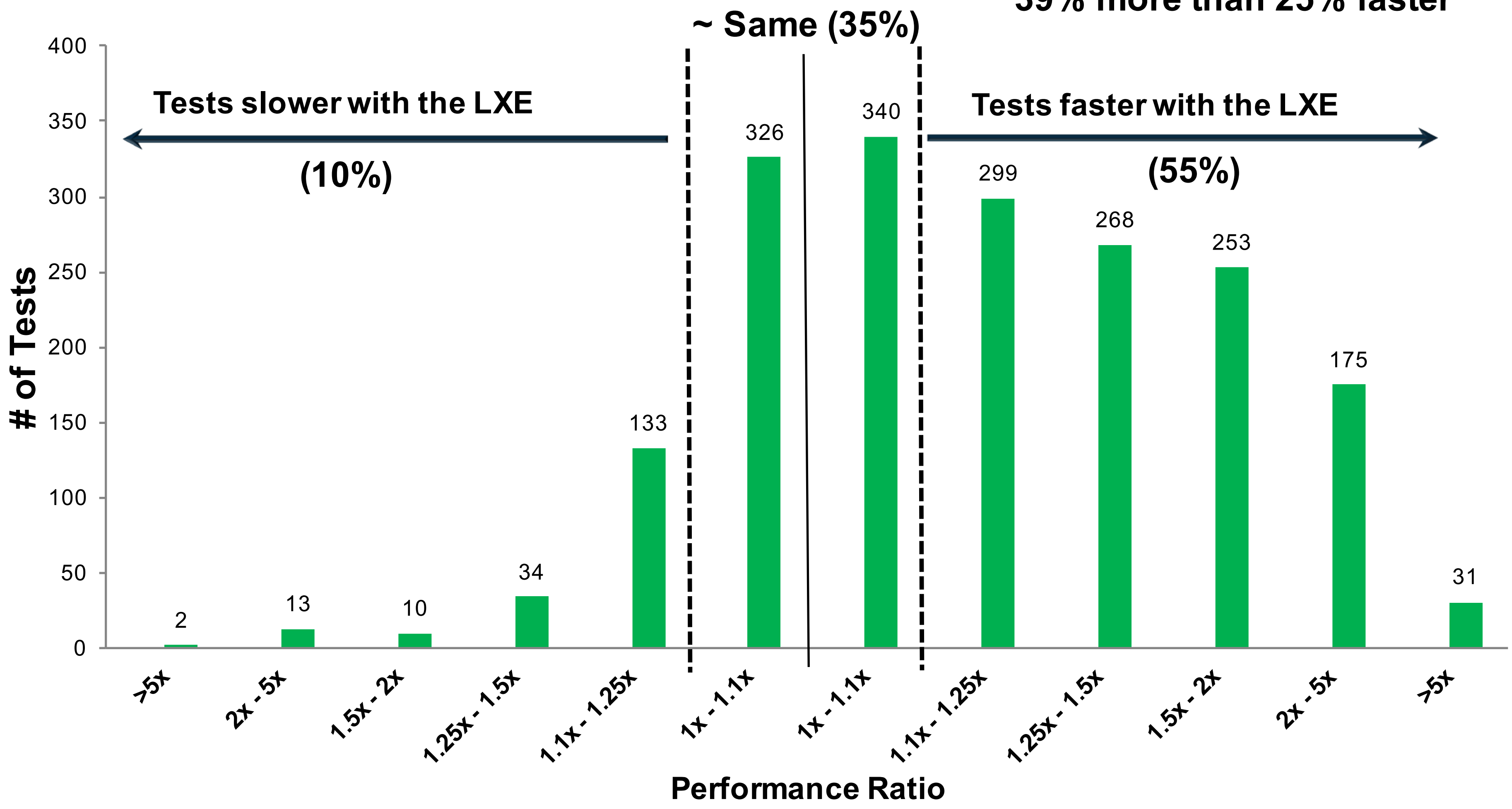
Application Level Benchmarks

99% on par or faster with LXE
64% more than 10% faster



Core and Toolbox UPS tests

90% on par or faster with LXE
55% more than 10% faster
39% more than 25% faster



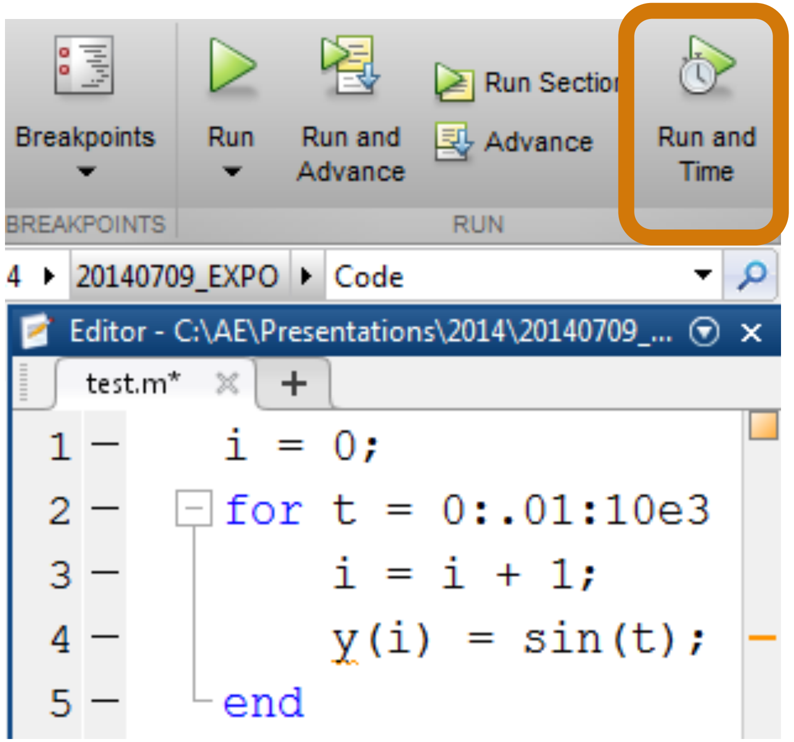
Lower-level tests show more variability

Acceleration Strategies Applied in MATLAB

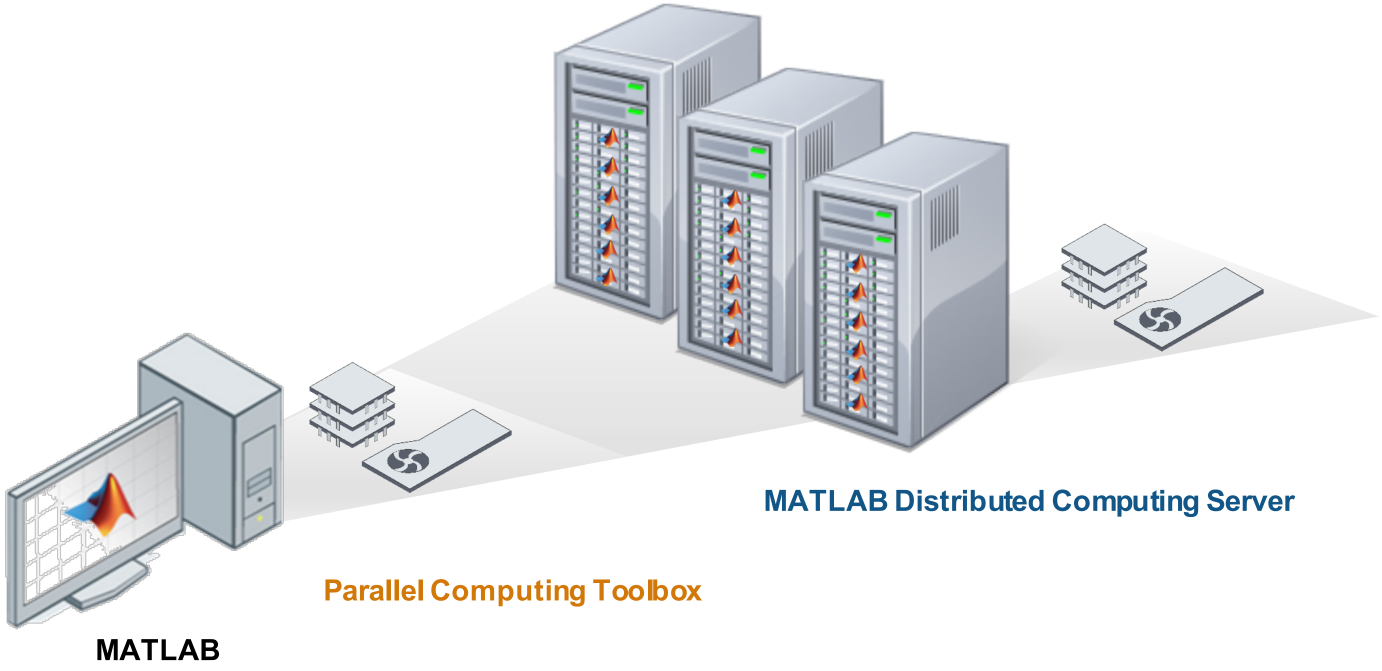
- **Best coding practices**
  - Use the Code Analyzer and Profiler
  - Preallocation
  - Vectorization

Lines where the most time was spent

Line Number	Code	Calls	Total Time	% Time	Time Plot
4	y(i) = sin(t);	1000001	0.198 s	52.5%	
3	i = i + 1;	1000001	0.093 s	24.7%	
5	end	1000001	0.086 s	22.8%	
2	for t = 0:.01:10e3	1	0 s	0%	
1	i = 0;	1	0 s	0%	
All other lines			0 s	0%	
Totals			0.377 s	100%	



Scale Compute Power

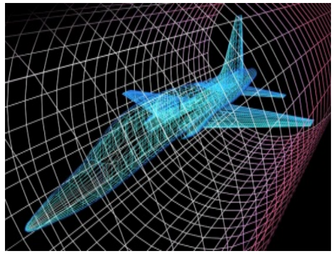


[Optimizing JIT Steel Manufacturing Schedule]

Cut simulation time from 1 hour to 5 minutes

[Heart Transplant Studies]

3-4 weeks reduced to 5 days

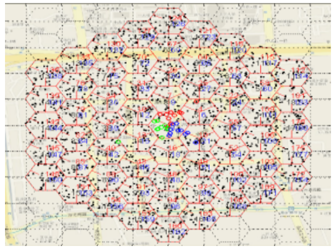


[Flight Test Data Analysis]

16x Faster

[Mobile Communications Technology]

Simulation time reduced from weeks to hours, 5x more scenarios

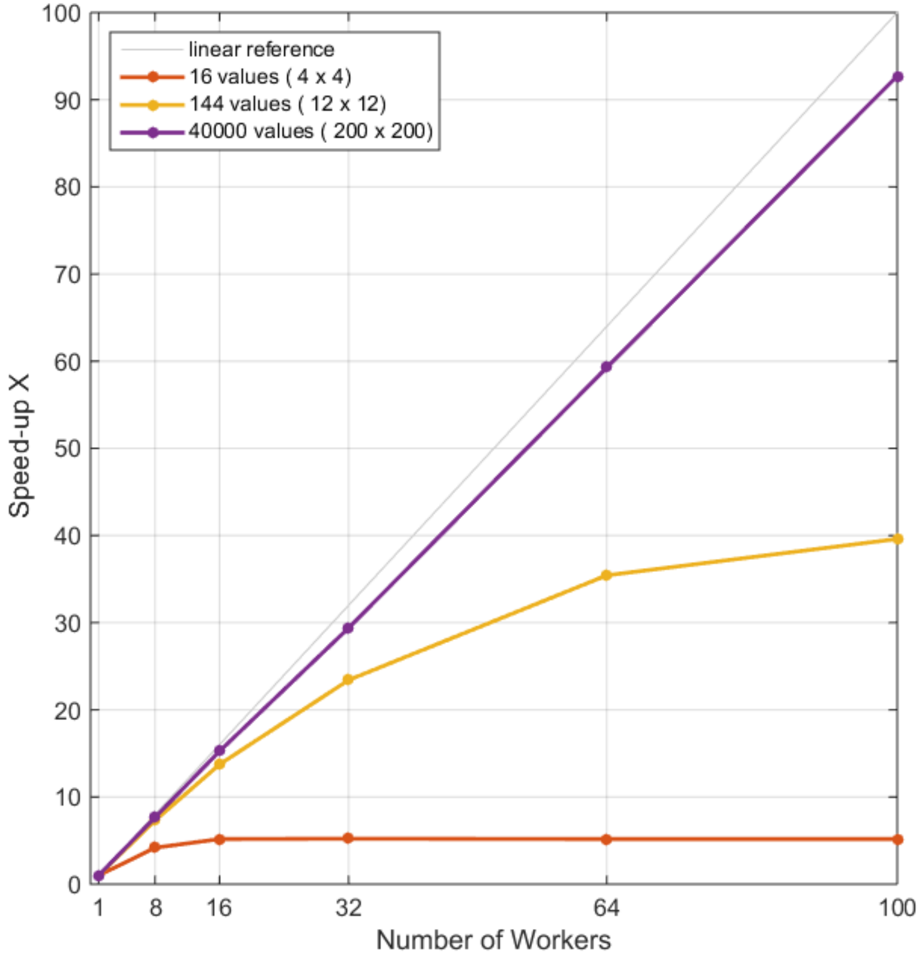


[Hedge Fund Portfolio Management]

Simulation time reduced from 6 hours to 1.2 hours

Benchmark: Parameter Sweep of ODEs

Scaling case study with a compute cluster



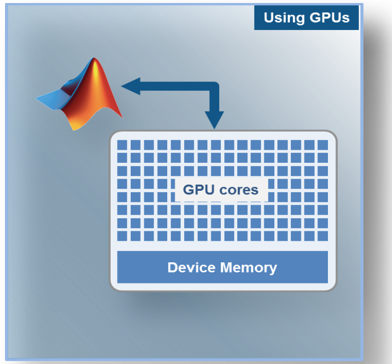
Workers in pool	Compute time (minutes)		
	200 x 200	12 x 12	4 x 4
1	241	0.90	0.11
8	32	0.12	0.03
16	16	0.07	0.02
32	8	0.04	0.02
64	4	0.03	0.02
100	3	0.02	0.02

Processor: Intel Xeon E5-class v2
16 physical cores per node

MATLAB code on the GPU

- Scaled parallel processing on workstation or cluster
- 200+ MATLAB functions supported on the GPU

Random number generation	Solvers	SVD
FFT	Convolutions	Cholesky and LU
Matrix multiplications	Min/max	factorization



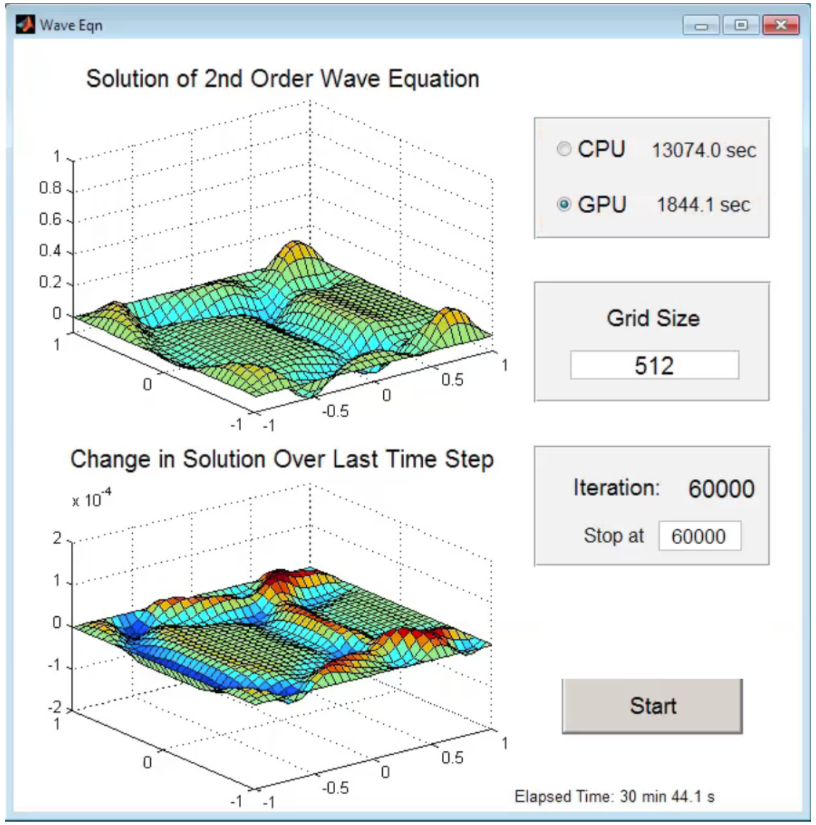
- Additional support in toolboxes

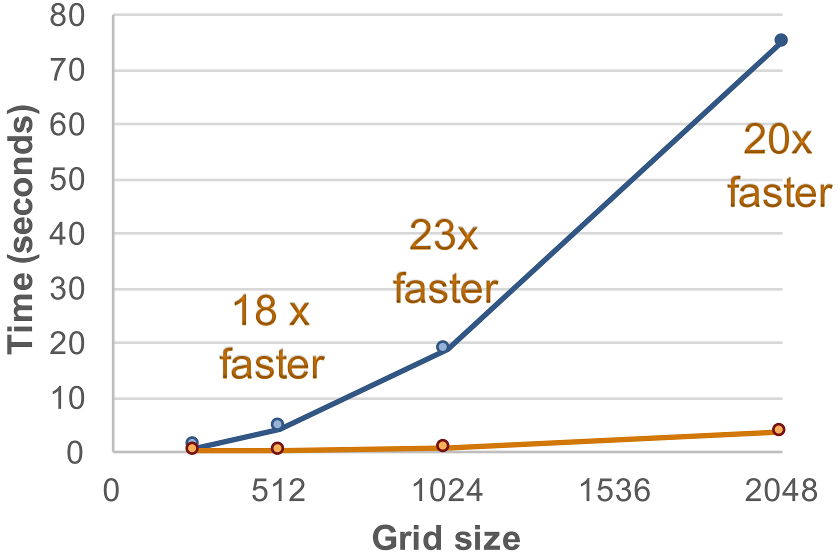
Image Processing	Communications	Signal Processing
Morphological filtering,	Turbo,	Cross correlation
2-D filtering	LDPC	FIR filtering
	Viterbi decoders	

Requires NVIDIA GPUs with Compute Capability 2.0 or higher.
See a complete listing at www.nvidia.com/object/cuda_gpus.html

Run Same Code on CPU and GPU

Solving 2D Wave Equation

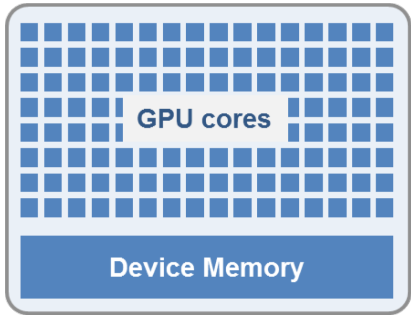




CPU	GPU
Intel(R) Xeon(R) W3550 3.06GHz 4 cores memory bandwidth 25.6 Gb/s	NVIDIA Tesla K20c 706MHz 2496 cores memory bandwith 208 Gb/s

Criteria for Good Problems to Run on a GPU

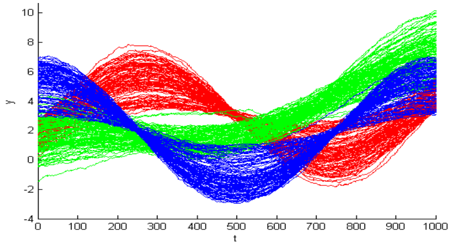
- **Massively parallel:**
  - Calculations can be broken into hundreds or thousands of independent units of work
  - Problem size takes advantage of many GPU cores



- **Computationally intensive:**
  - Computation time significantly exceeds CPU/GPU data transfer time

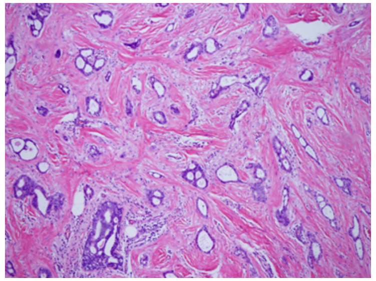
- **Algorithm consists of supported functions:**
  - Growing list of toolboxes with built-in support
    - Parallel Support in Toolboxes (pdf)
  - Subset of core MATLAB for `gpuArray`, `arrayfun`, `bsxfun`
    - MATLAB functions with gpuArray arguments (doc)
    - Run element-wise MATLAB code on a GPU (doc)

Speed up MATLAB code with NVIDIA GPUs

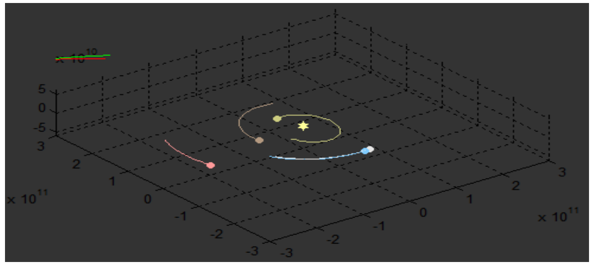


10x speedup *in data clustering via K-means clustering algorithm*

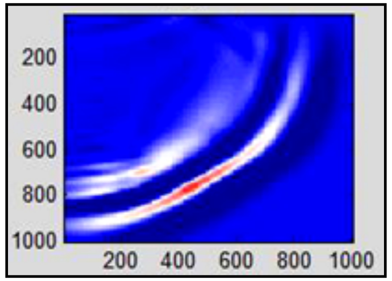
20x speedup *in wind tunnel acoustic data analysis (NASA Langley Research Center)*



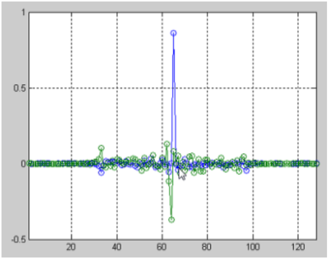
14x speedup *in template matching (part of cancer cell image analysis)*



17x speedup *in simulating the movement of 3072 celestial objects*

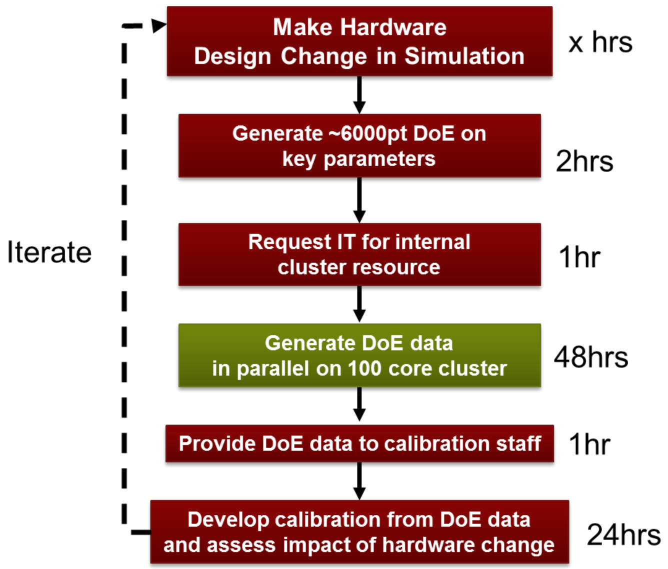


4x speedup *in wave equation solving (part of seismic data processing algorithm)*

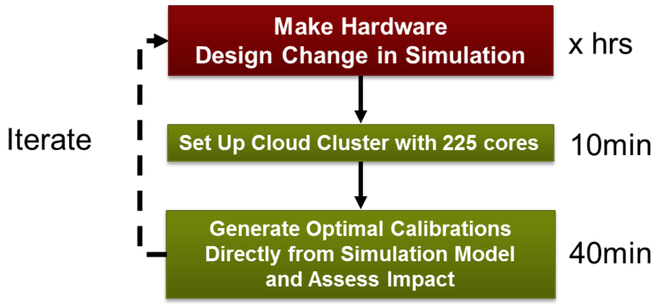


4x speedup *in adaptive filtering (part of acoustic tracking algorithm)*

Generating optimal solutions efficiently



Typical Engine Development Process (76 hrs)



Fully-automated VECO Process (50 min)

Intel® Xeon® processor E5 v2
16 physical cores per node

Human Labor
Automation

Agenda

- MATLAB Infrastructure
  - Editor
  - Graphics
- Workflows
  - Managing / Testing Code
  - Sharing Apps and Custom Toolboxes
- Performance
  - Acceleration Strategy
  - Execution Engine
  - Parallel computing and GPU computation
- Wrap up & QnA