MATLAB® Today

By Steve Eddins, Ph.D.

© 2015 The MathWorks, Inc.

True or false:

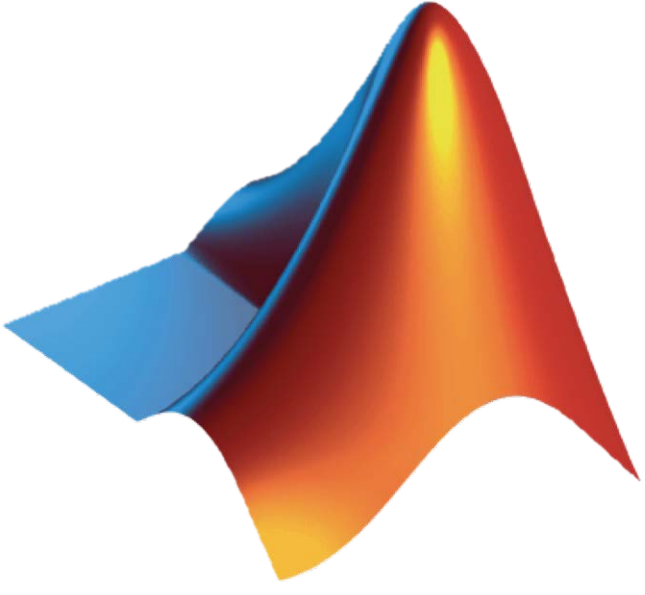
Every cell in the human body gets replaced every 7 years

Continuously Rebuilding the Human Body

- Mostly true
  - Skin cells – 35 days
  - Red blood cells – 120 days
  - Skeleton – 10 years
  - Neocortical neurons – lifetime

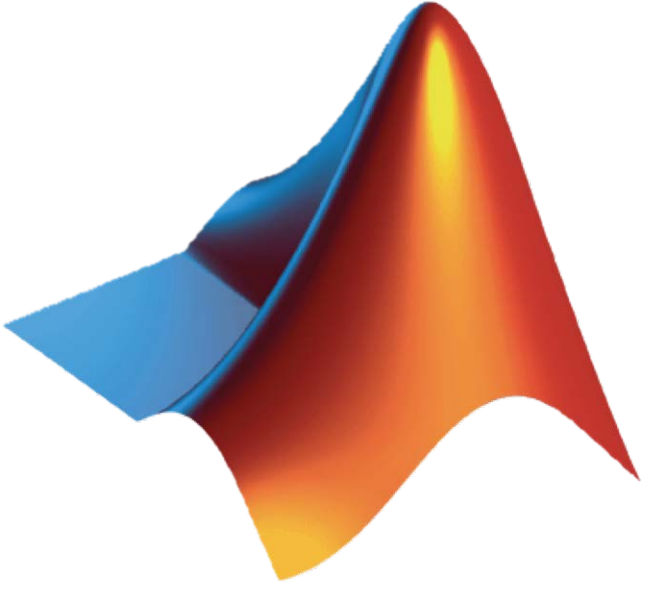
Public domain

Continuously Rebuilding MATLAB



Almost everything in MATLAB is completely new or has been rebuilt over the last 7-10 years.

Continuously Rebuilding MATLAB



Why?

Your needs change.

Computational technologies and environments change.

The pace of change in MATLAB has never been more rapid.

81137_92238v00_RainbowColorMap_57312.pdf (page 1 of 18)

Rainbow Color Map Critiques: An Overview and Annotated Bibliography

By Steve Eddins, MathWorks

Overview

A rainbow color map is based on the order of colors in the spectrum of visible light—the same colors that appear in a rainbow. Figure 1 shows one particular form of rainbow color map. Since multiple methods exist for computing rainbow color maps, they do not all look identical, but they all feature the same general ordering of colors.

Figure 1. *Visible light spectrum.*

Data credit: "Spectral and XYZ Color Functions," Jeff Mather, MATLAB Central File Exchange.

Rainbow color maps commonly appear in data visualizations in many different scientific and engineering communities. Technical computing software often provides a rainbow color map as the default choice.

Before Release 2014b, MATLAB® used a rainbow color map as the default for figures. In MATLAB, the rainbow color map is known as

Rainbow colormap paper (demonstrating annotations).pdf (page 1 of 18)

Search

Rainbow Color Map Critiques:
...view and Annotated Bibliography

A rainbow

shows one

identical,

...e order of colors in the spectrum of visible light—the same colors that appear in a rainbow. Figure 1
...w color map. Since multiple methods exist for computing rainbow color maps, they do not all look
...me general ordering of colors.

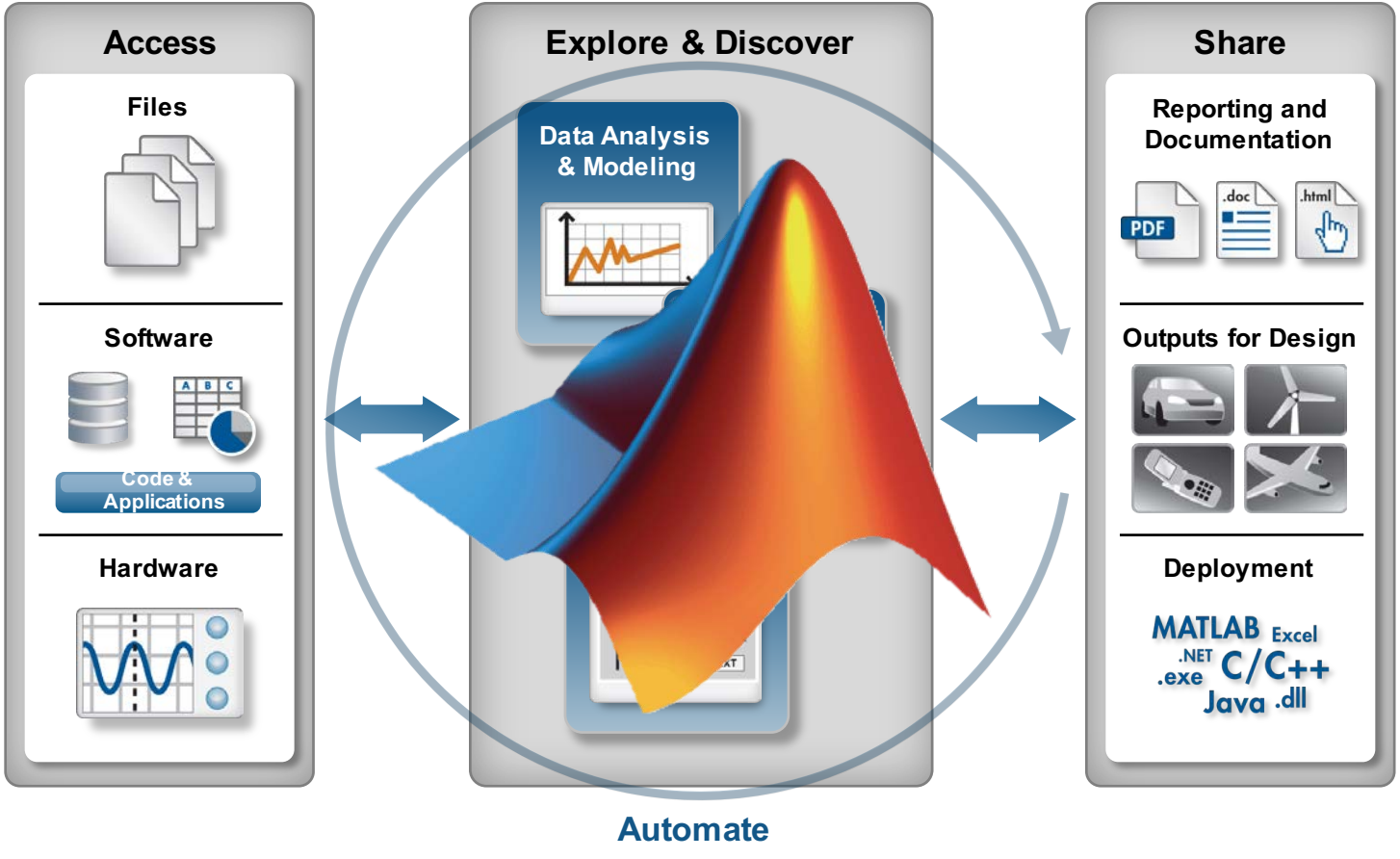
Talk to Jeff about this

...ectral and XYZ Color Functions," Jeff M...her, MATLAB Central File Exchange.

Rainbow color maps commonly appear in data visualizations in many different scientific and engineering communities. Technical computing software often provides a rainbow color map as the default choice.

Before Release 2014b, MATLAB® used a rainbow color map as the default for figures. In MATLAB, the rainbow color map is known as

Technical Computing Workflow



So Here's Where We're Going in the Next Hour

- Access
  - Files
  - Software
  - Hardware
- Graphics
- Math
- Software Development
- Image Processing
- Computer Vision

Getting Your Data

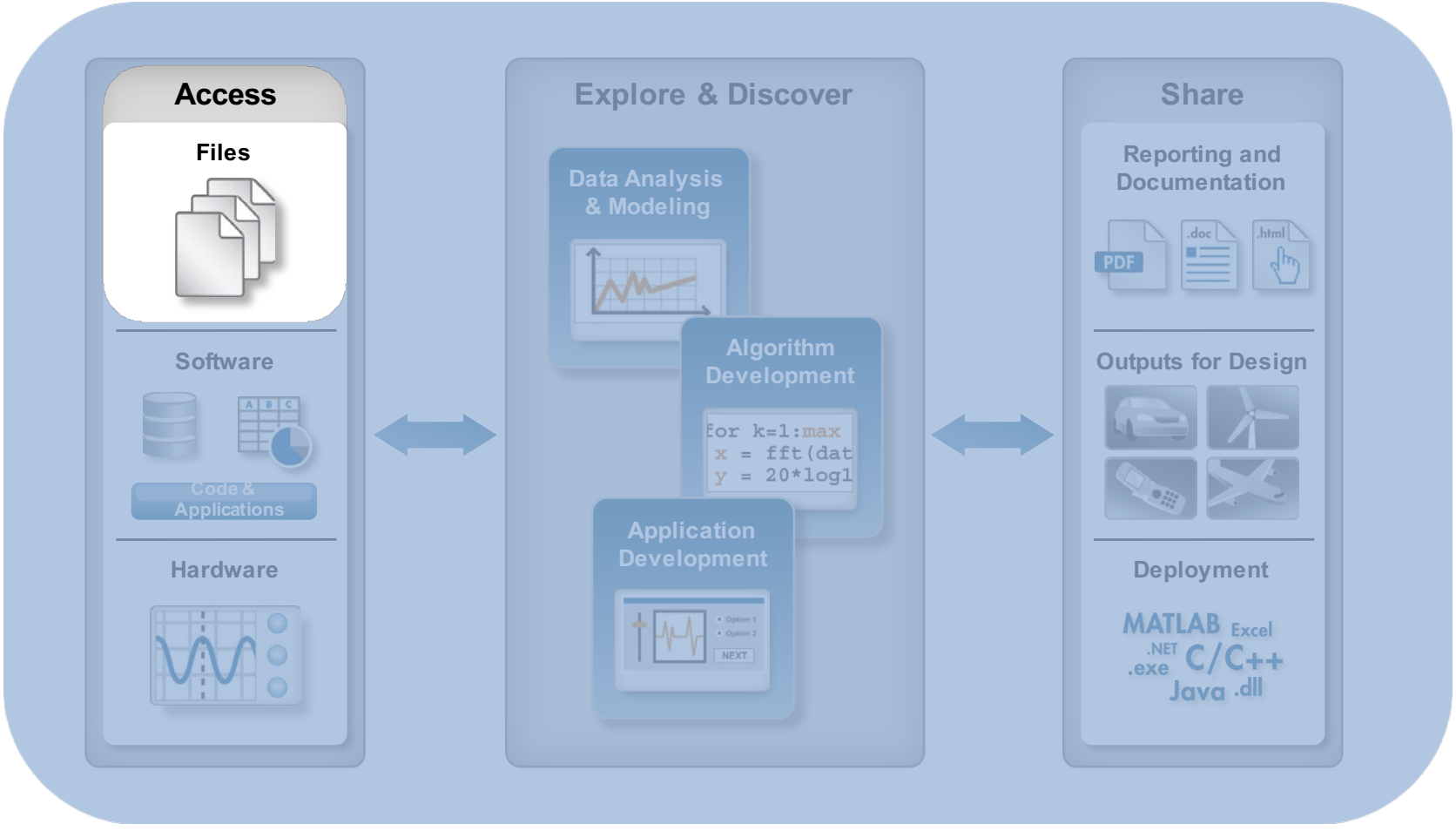


Image File Enhancements

TIFF:

- LogL and LogLUV HDR files
- BigTIFF

NITF JPEG

DPX

DICOM anonymizer (privacy requirements)

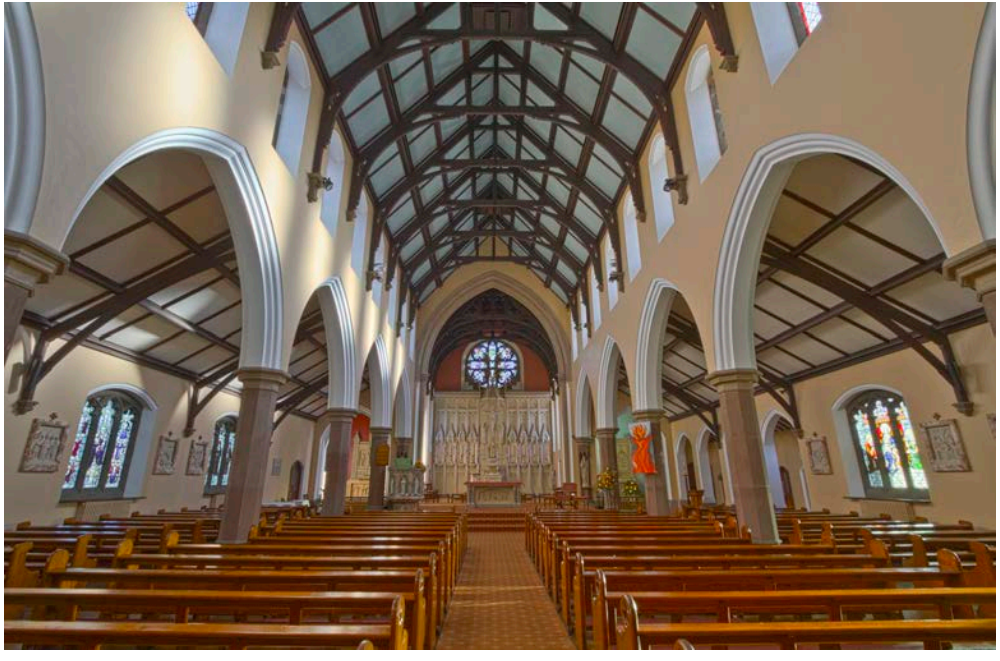
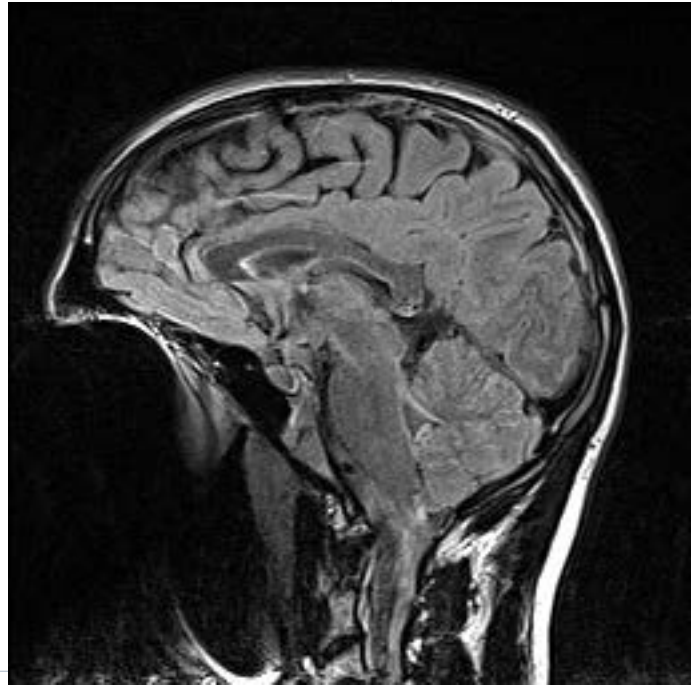


Image credit: Michael Beckwith

Command Window

```
>> dicomanon('31237418','31237418a')
>> info = dicominfo('31237418a');
>> info.PatientName

ans =

     FamilyName: ''
      GivenName: ''
     MiddleName: ''
     NamePrefix: ''
     NameSuffix: ''
```

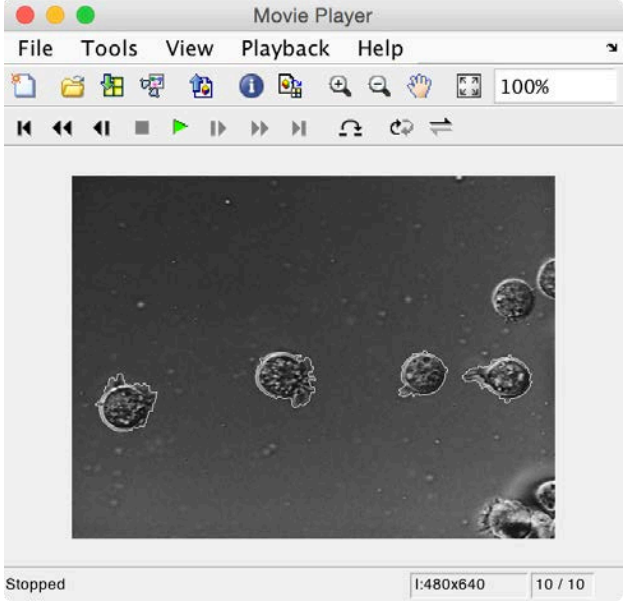
31237418

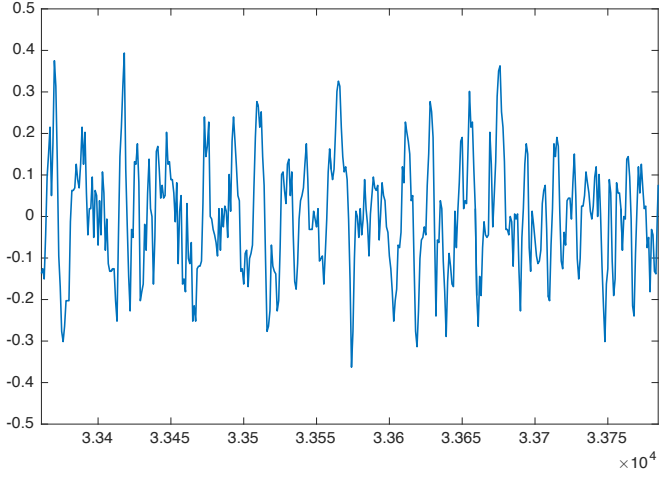


Video and Audio File Enhancements

MPEG-4

Mac support for MPEG-4 H.264





MP3
MPEG-4 audio
MPEG-4 AAC
Wave enhancements

Scientific Data File Enhancements

HDF5, netCDF

OPeNDAP online servers

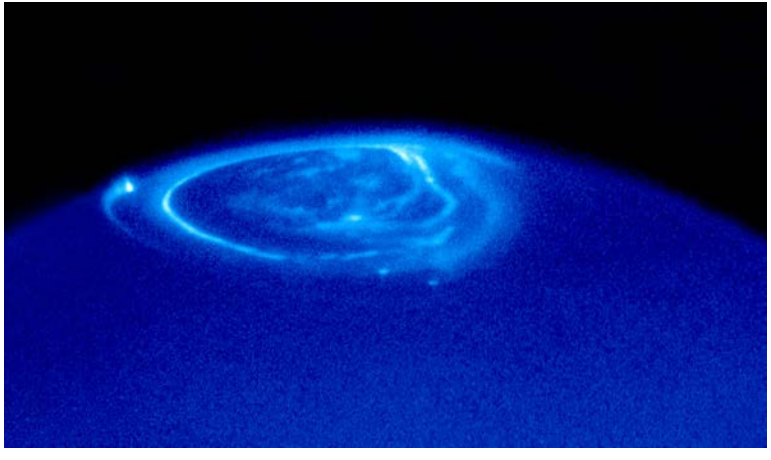


Image credit: NASA, ESA & John T. Clarke (Univ. of Michigan)

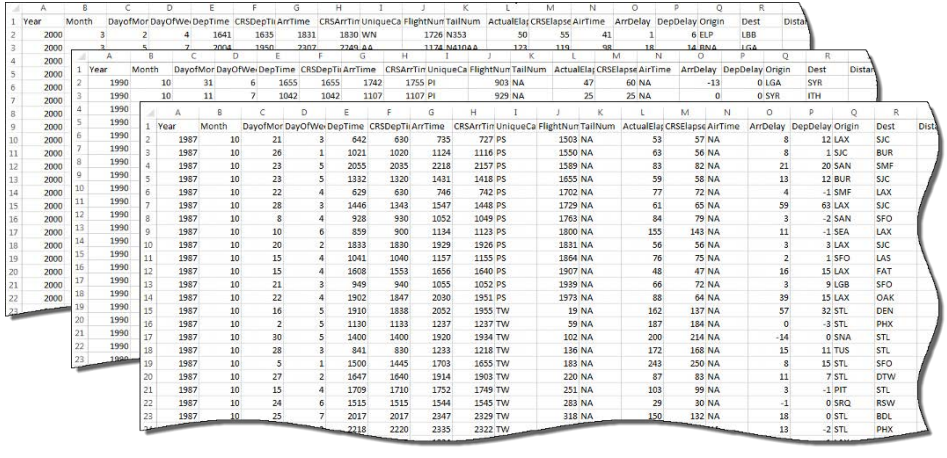
FITS export

Datastores

Datastore – for reading structured data in one or many files

Good for big data, Hadoop, HDFS

Tabular text datastores
Image datastores



Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier	FlightNum	TailNum	ActualElapsedTime	CRSElapsedTime	AirTime	ArrDelay	DepDelay	Origin	Dest	Distance
1987	10	21	3	642	630	735	727	PS	1503	NA	53	57	NA	8	12	LAX	SJC	308
1987	10	26	1	1021	1020	1124	1116	PS	1550	NA	63	56	NA	8	1	SJC	BUR	324
1987	10	23	5	2055	2035	2218	2157	PS	1589	NA	83	82	NA	21	20	SAN	SMF	480
1987	10	23	5	1332	1320	1431	1418	PS	1655	NA	59	58	NA	13	12	BUR	SJC	324
1987	10	22	4	629	630	746	742	PS	1702	NA	77	72	NA	4	-1	SMF	LAX	387
1987	10	28	3	1446	1343	1547	1448	PS	1729	NA	61	65	NA	59	63	LAX	SJC	308
1987	10	8	4	928	930	1052	1049	PS	1763	NA	84	79	NA	3	-2	SAN	SFO	447
1987	10	10	6	859	900	1134	1123	PS	1800	NA	155	143	NA	11	-1	SEA	LAX	954
1987	10	20	2	1833	1830	1929	1926	PS	1831	NA	56	56	NA	3	3	LAX	SJC	308
1987	10	15	4	1041	1040	1157	1155	PS	1864	NA	76	75	NA	2	1	SFO	LAS	414
1987	10	15	4	1608	1553	1656	1640	PS	1907	NA	48	47	NA	16	15	LAX	FAT	234
1987	10	21	3	949	940	1055	1052	PS	1939	NA	66	72	NA	3	9	LGB	SFO	337
1987	10	22	4	1902	1847	2030	1951	PS	1973	NA	88	64	NA	39	15	LAX	OAK	337
1987	10	16	5	1910	1838	2052	1955	TW	19	NA	162	137	NA	57	32	STL	DEN	862
1987	10	2	5	1130	1133	1237	1237	TW	59	NA	187	184	NA	0	-3	STL	PHX	1216
1987	10	30	5	1400	1400	1920	1934	TW	102	NA	200	214	NA	-14	0	SNA	STL	1363
1987	10	28	3	841	830	1233	1218	TW	136	NA	172	168	NA	15	11	TUS	STL	1109
1987	10	5	1	1500	1445	1703	1655	TW	183	NA	243	250	NA	8	15	STL	SFO	1843
1987	10	27	2	1647	1640	1914	1903	TW	220	NA	87	83	NA	11	7	STL	DTW	628
1987	10	15	4	1709	1710	1752	1749	TW	251	NA	103	99	NA	3	-1	PIT	STL	720
1987	10	24	6	1515	1515	1544	1545	TW	283	NA	29	30	NA	-1	0	SRQ	RSW	1021
1987	10	25	7	2017	2017	2347	2329	TW	318	NA	150	132	NA	18	0	STL	BDL	1023
				2218	2220	2335	2322	TW						13	-2	STL	PHX	

Web Service Interfaces

- Often called *RESTful interfaces*
- webread
- webwrite

Example RESTful API: NASA Landsat Imagery

HTTP REQUEST

```
GET https://api.nasa.gov/planetary/earth/imagery
```

QUERY PARAMETERS

Parameter	Type	Default	Description
lat	float	n/a	Latitude
lon	float	n/a	Longitude
dim	float	0.025	width and height of image in degrees
date	YYYY-MM-DD	*today*	date of image; if not supplied, then the most recent image (i.e., closest to today) is returned
cloud_score	bool	False	calculate the percentage of the image covered by clouds
api_key	string	DEMO_KEY	api.nasa.gov key for expanded usage

https://api.nasa.gov/api.html#imagery

Command Window

```
>> url = 'https://api.nasa.gov/planetary/earth/imagery';
>> result = webread(url,...
                    'lat',42.300,...
                    'lon',-71.351,...
                    'api_key','DEMO_KEY')

result =

    date: '2015-08-28T15:26:54'
     url: 'https://earthengine.googleapis.com/api/thum…'
      id: 'LC8_L1T_TOA/LC80120312015240LGN00'

>> mw = webread(result.url);
>> imshow(mw)
```

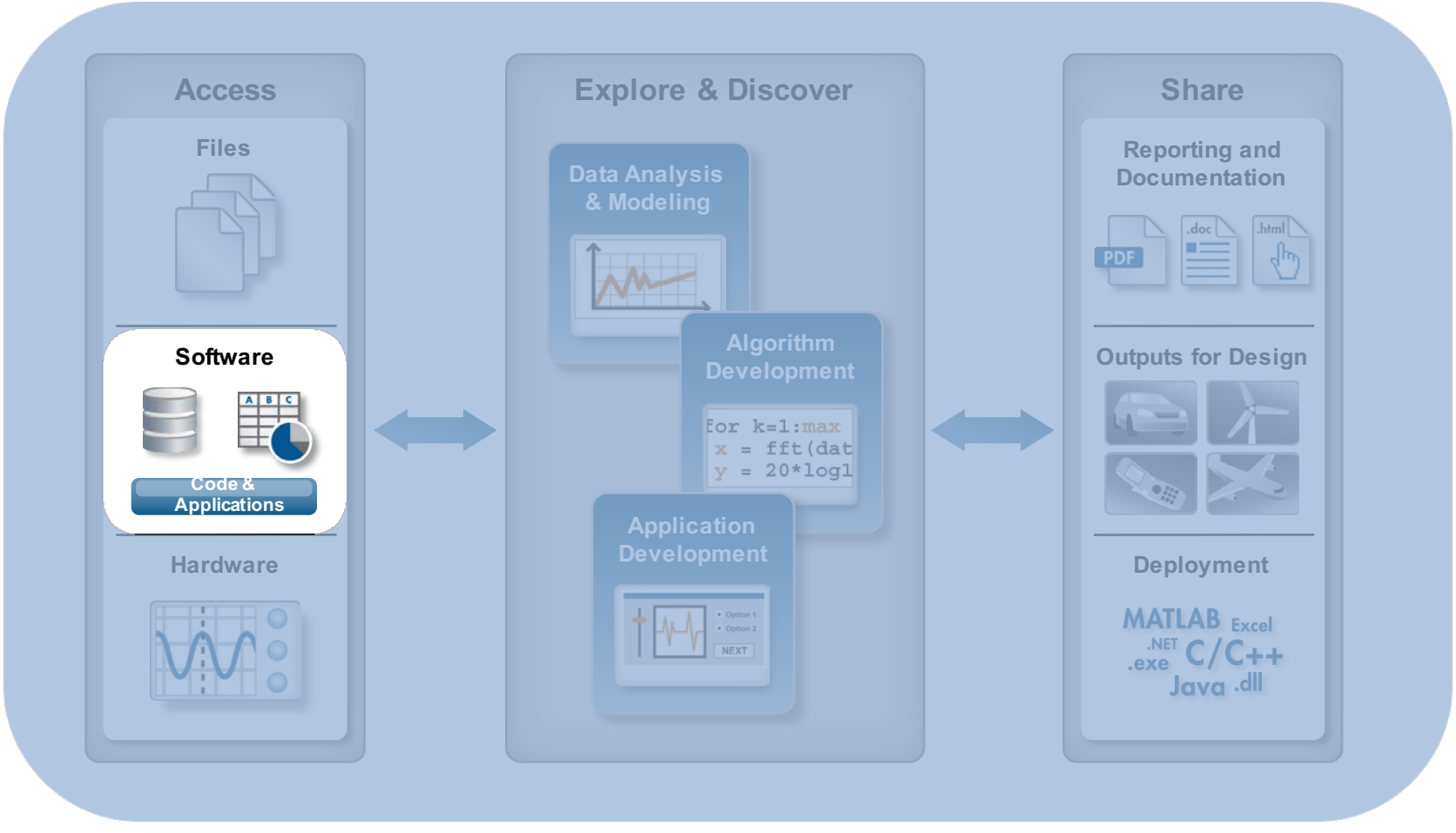
Command Window

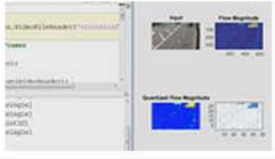
```
>> url = 'https://api.nasa.gov/planetary/earth/imagery';
>> result = webread(url,...
                    'lat',42.300,...
                    'lon',-71.351,...
                    'api_key','DEMO_KEY'

result =

    date: '2015-08-28T15:26:54'
     url: 'https://earthengine.googleapi
      id: 'LC8_L1T_TOA/LC80120312015240L

>> mw = webread(result.url);
>> imshow(mw)
```

Web Service Interface options

- Character encoding
- Timeout (how long before giving up on request)
- Username, password
- Request method (GET or POST)
- Desired content type (text, audio, image, table, etc.)

Using Other Software and Apps





Computer Vision System Toolbox OpenCV Interface

by MathWorks Computer Vision System Toolbox Team
03 Oct 2014 (Updated 04 Sep 2015)

Use OpenCV algorithms in MATLAB

Watch this File

2.5 | 2 ratings
Rate this file

208 Downloads (last 30 days)
File Size: 15.1 KB
File ID: #47953
Version: 1.5

OpenCV Interface: What's Included

- Prebuilt OpenCV binaries
- Build script to create OpenCV based MEX-files
- Data type conversions between MATLAB and OpenCV
- Examples to help get started with common workflows

Python

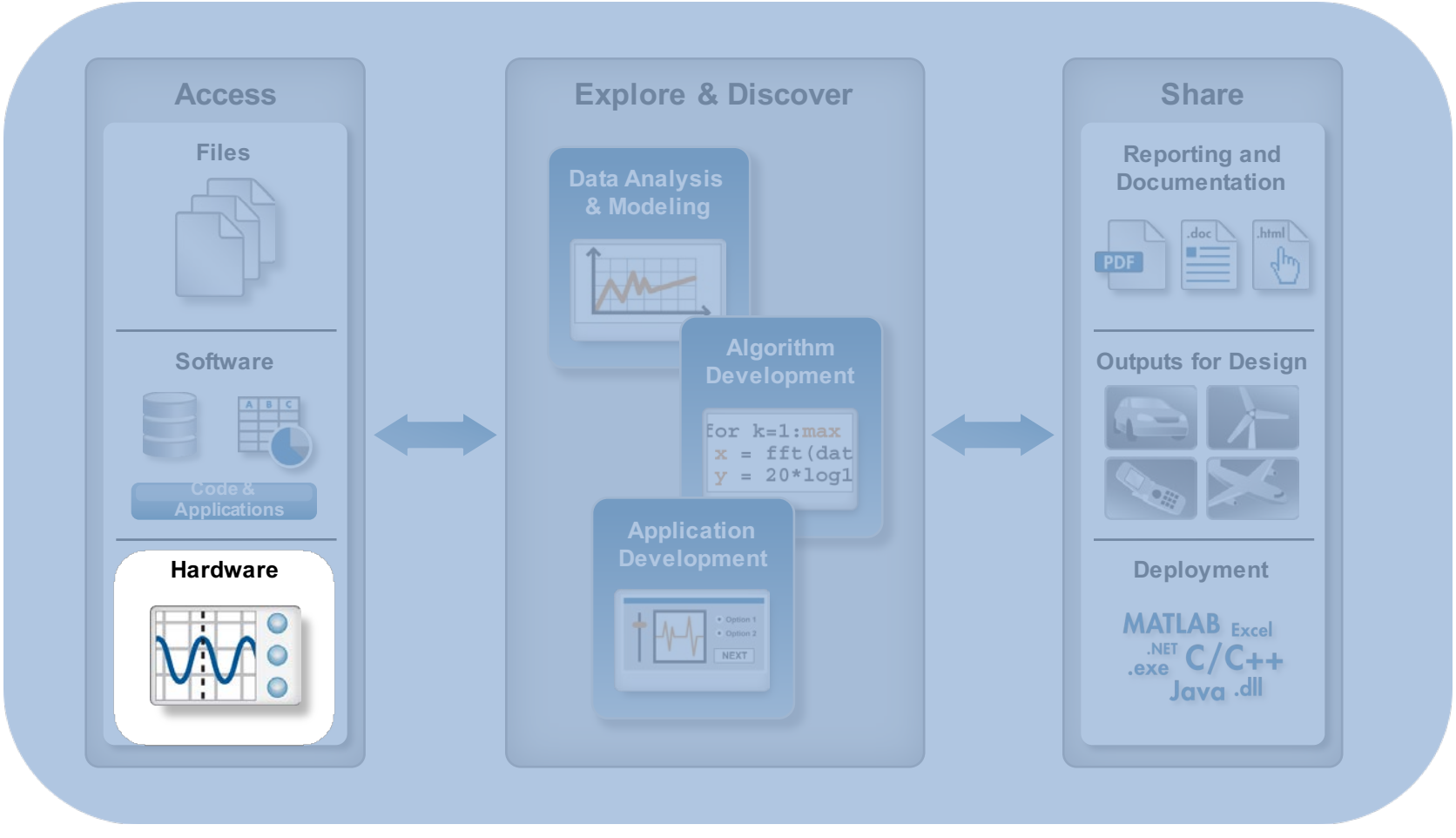
- Call Python from MATLAB
- Call MATLAB from Python

Command Window

```
>> T = 'A watershed is the ridge that divides areas drai
>> tw = py.textwrap.TextWrapper(pyargs(...
         'initial_indent', '% ', ...
         'subsequent_indent', '% ', ...
         'width', int32(30)));
>> wrapped = wrap(tw,T);
>> wrapped = cellfun(@(pystr) {char(pystr)},wrapped);
>> fprintf('%s\n',wrapped{:})

% A watershed is the ridge
% that divides areas drained
% by different river systems.
% A catchment basin is the
% geographical area draining
% into a river or reservoir.
% The watershed transform
% applies these ideas to gray-
% scale image processing in a
% way that can be used to
% solve a variety of problems.
```

```
>>> import matlab.engine
>>> eng = matlab.engine.start_matlab()
>>> t = eng.gcd(18.0,4.0,nargout=3)
>>> print(t)
(2.0, 1.0, -4.0)
```

Hooking Up Your Hardware



Explosion in Hardware Support

117 manufacturers

Application areas:

- Communications
- Control systems
- DSP
- Embedded systems
- FPGA
- Image processing and computer vision
- Internet of things
- Mechatronics

Protocols and standards:

- Audio
- Bluetooth
- CAN
- Ethernet
- I2C/SPI
- Safety
- Serial
- USB
- Video

Explos…pport

Adimec
Use Adim…
Vendors: A…
Tags: Math…

Allied V…
Use AV c…
Vendors: A…
Tags: Math…

Altera S…
Use Altera…
Vendors: A…
Tags: Conn…

Analog …
Use analo…
processin…
Vendors: A…
Panasonic,…
Tags: Math…

Automa…
Use Auto…
Vendors: Automation Technology
Tags: MathWorks Supported

Basler Camera Support from Image Acquisition…
Use Basler cameras with MATLAB and Simulink to acquire …
Vendors: Basler
Tags: MathWorks Supported

Baumer Camera Support from Image Acquisiti…
Use Baumer cameras with MATLAB and Simulink to acquir…
Vendors: Baumer
Tags: MathWorks Supported

BeagleBoard Support fr…
Low-cost, single-board comput…
Vendors: BeagleBoard
Tags: C/C++ Code Generation, Math…
Installer Enabled

BeagleBone Black Supp…
Generate code optimized for B…
Vendors: ARM, BeagleBoard, Texas …
Tags: C/C++ Code Generation, Math…

BeagleBone Black Supp…
Acquire sensor and image data…
Vendors: BeagleBoard, Texas Instru…
Tags: MathWorks Supported, Project-…

BitFlow Frame Grabber …
Use BitFlow Frame Grabbers w…
Vendors: BitFlow
Tags: MathWorks Supported

BitFlow Neon Support fr…
Use BitFlow Neon with Simulin…
medical, or semiconductor ima…
Vendors: BitFlow
Tags: MathWorks Supported, Suppor…

Camera Link Support from Image Acquisition Toolb…
Use Camera Link frame grabbers with MATLAB and Simulink to s…
image processing and computer vision workflows.
Vendors: BitFlow, Data Translation, Imperx, Matrox Imaging, National Instrume…
Tags: MathWorks Supported

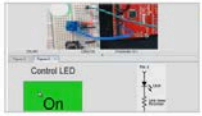
Cohu C…
Use Coh…
Vendors: …
Tags: Mat…

Data T…
Use Data…
Vendors: …
Tags: Mat…

DCAM …
Use DCA…
image pr…
Vendors: …
Tags: Mat…

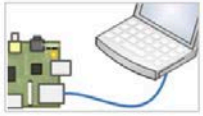
FLIR C…
Use FLIR…
Vendors: …
Tags: Mat…

GenTL…
Use Gen…
processi…
Vendors: …
Tags: Mat…



Arduino Support from MATLAB
Student-priced microcontroller board for introducing electrical enginee…
mechatronics
Vendors: Arduino
Tags: Project-Based Learning, Support Package Installer Enabled

Arduino Support from Simulink
Student-priced microcontroller board for introducing electrical enginee…
mechatronics.
Vendors: Arduino
Tags: Support Package Installer Enabled, C/C++ Code Generation, MathWorks Sup…
Target Hardware



Raspberry Pi Support from MATLAB
Use MATLAB to acquire sensor and image data from your connected…
Vendors: Raspberry Pi
Tags: MathWorks Supported, Project-Based Learning, Support Package Installer En…

Raspberry Pi Support from Simulink
Credit-card sized, low-cost, single-board computer with audio and vid…
teaching.
Vendors: Raspberry Pi
Tags: C/C++ Code Generation, MathWorks Supported, Project-Based Learning, Run…
Installer Enabled

ThingSpeak Support from Desktop MATLAB
Prototype Internet of Things (IoT) applications using ThingSpeak and…
Vendors: ioBridge

USB Webcam and IP Cameras

USB Webcam Support with MATLAB

Use webcams with MATLAB to acquire images and video on PCs.

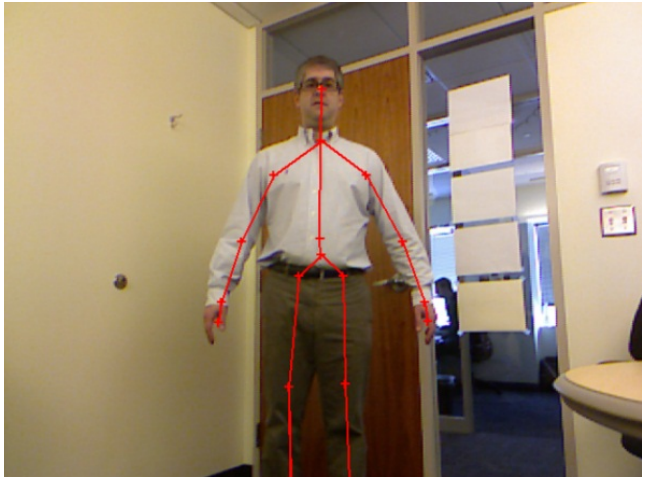
- USB Webcam Support with MATLAB
- IP Camera Support from MATLAB

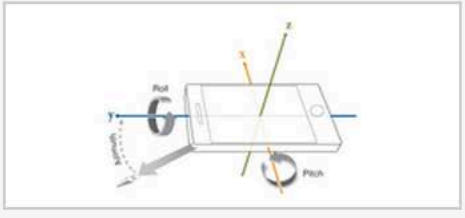
Using UVC compliant webcams with MATLAB®, you can explore and develop live image processing and computer vision applications on PCs.

MATLAB provides webcam support through a Hardware Support Package available via the Support Package Installer.

Microsoft Kinnect

- Image Acquisition Toolbox (Windows only)
  - Depth map
  - Video
  - Skeleton data for up to 6 people in field of view





MATLAB Support Package for Apple iOS Sensors

by MathWorks Mobile Sensor Connectivity Team
15 Jul 2015 (Updated 21 Jul 2015)

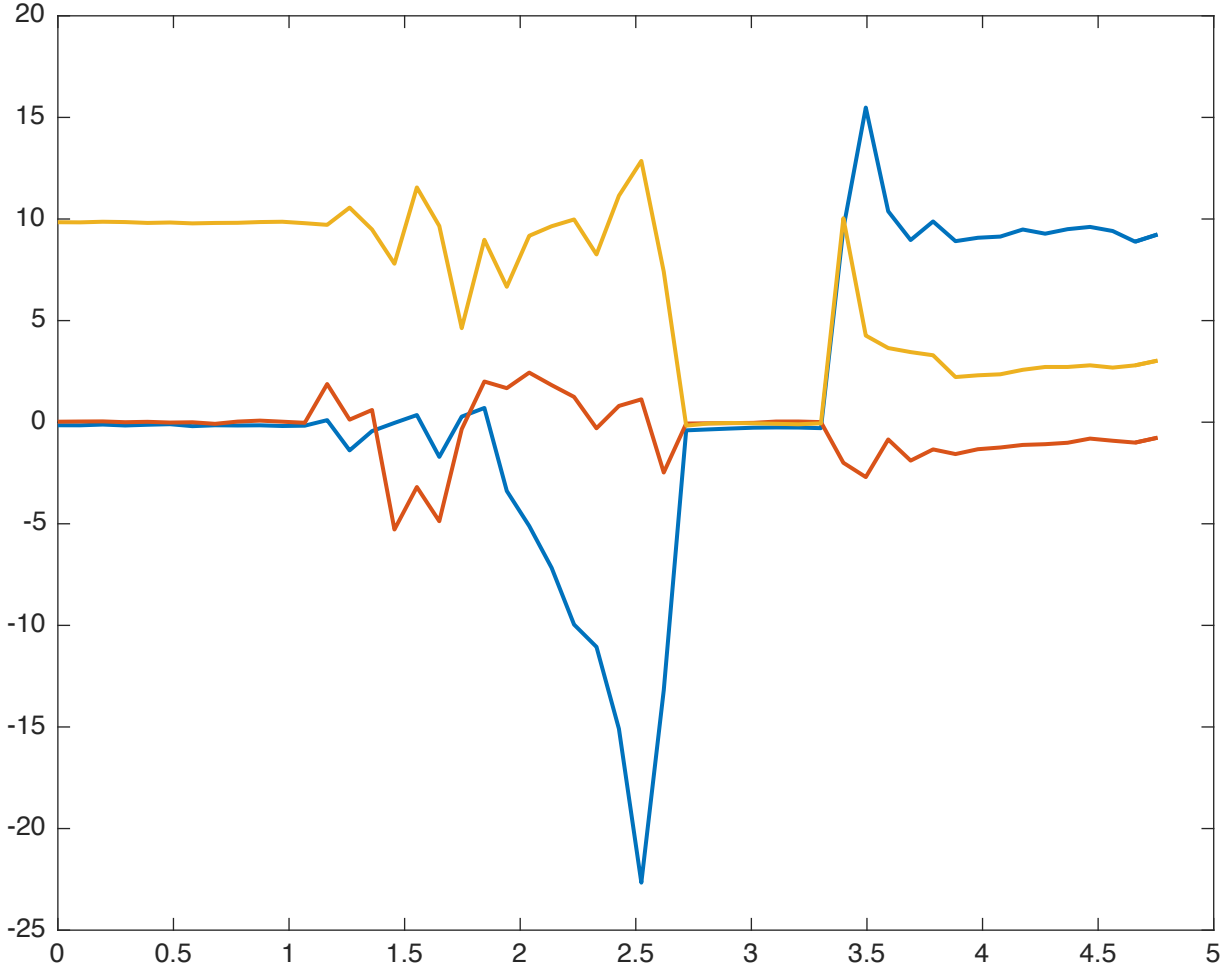
Use MATLAB to acquire sensor data from built-in sensors on your Apple iOS device

Watch this File

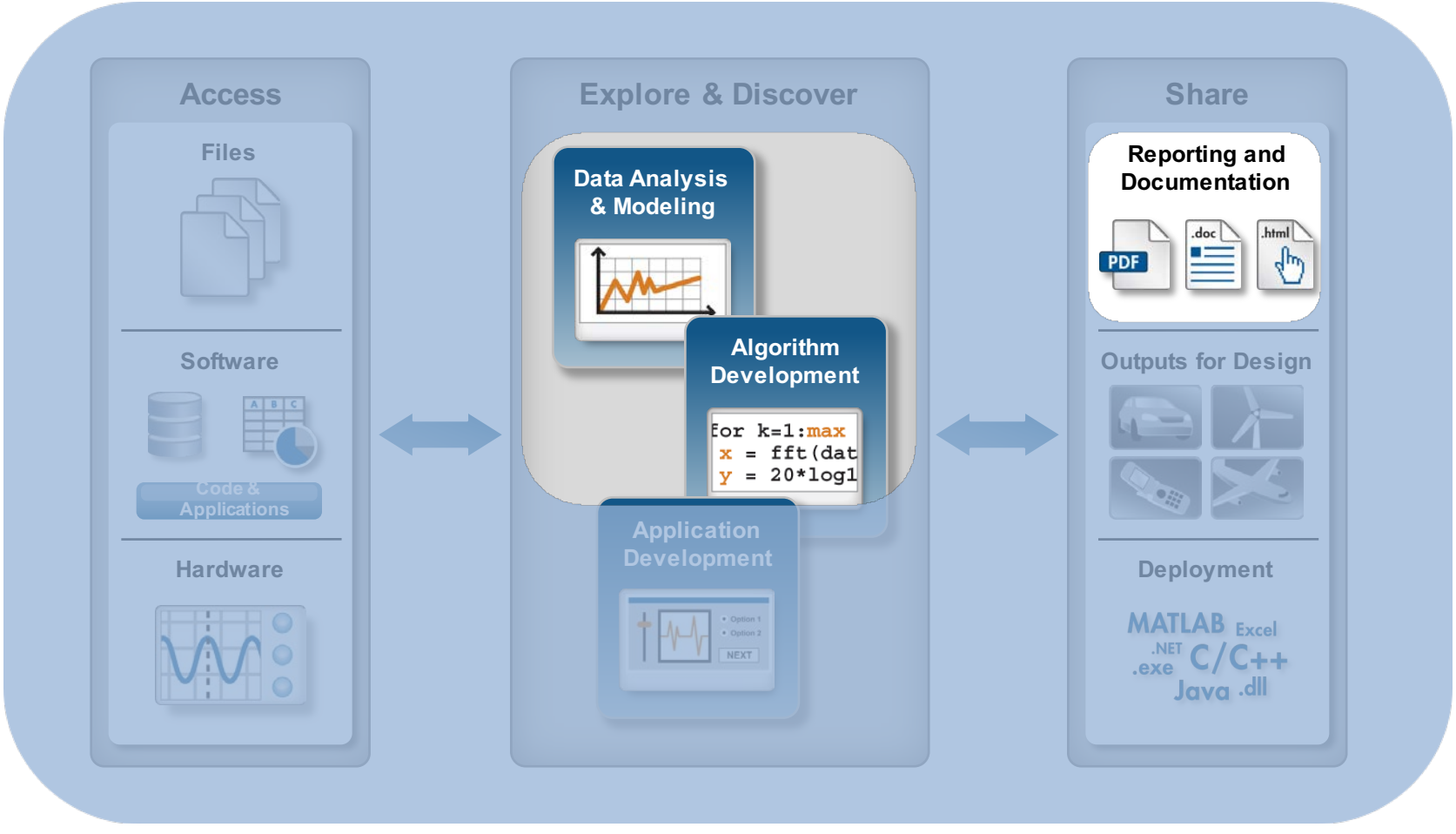
5.0 | 1 rating
Rate this file

413 Downloads (last 30 days)
File Size: 15.1 KB
File ID: #51235
Version: 1.0

Foolish Demo



MATLAB Graphics

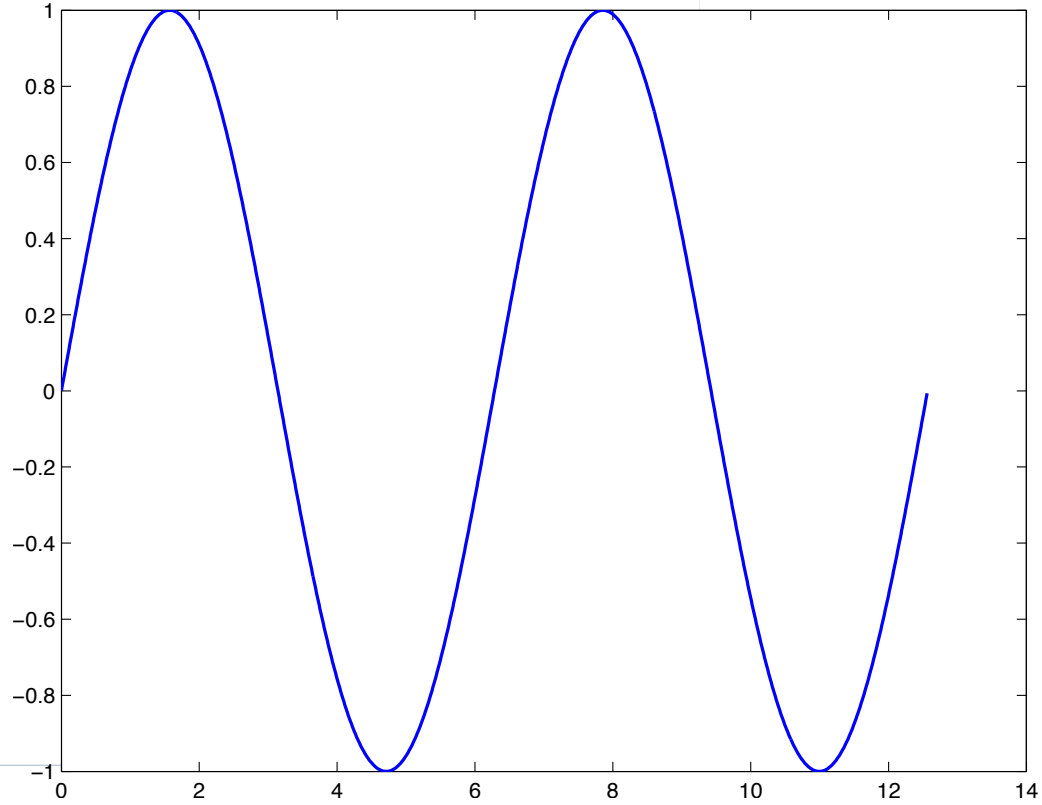


MathWorks

R2014a

Command Window

```
>> t = 0:0.01:4*pi;
>> p = plot(t,sin(t))

p =

  174.0016
```



MathWorks

R2014b

Command Window

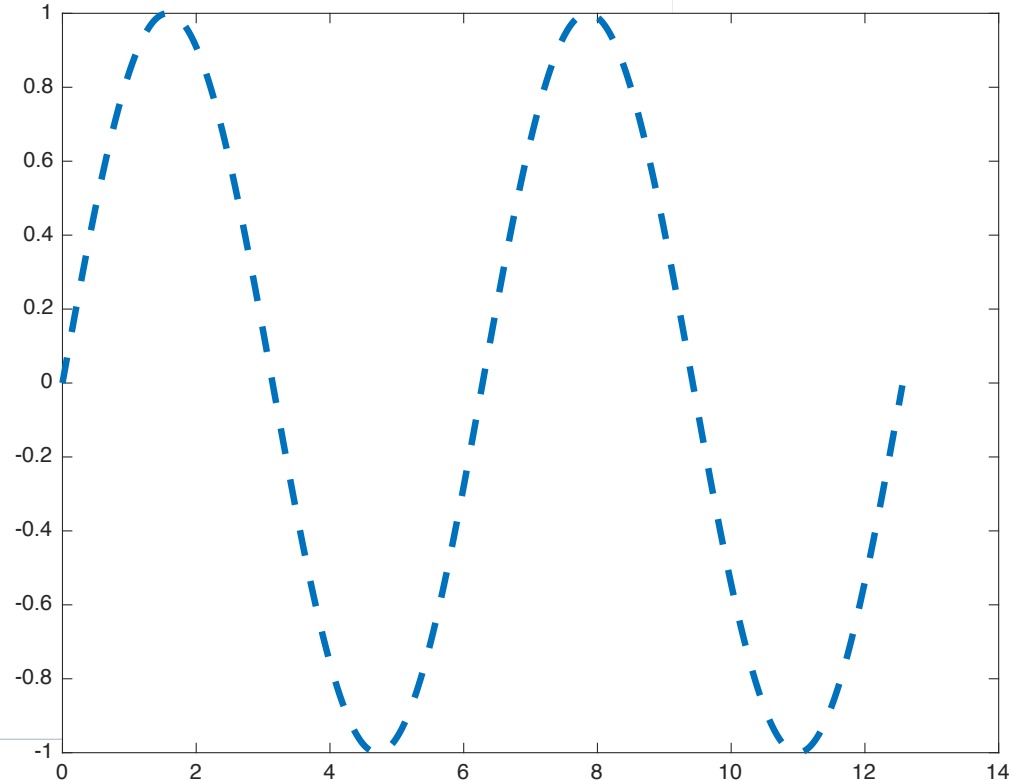
```
>> t = 0:0.01:4*pi;
>> p = plot(t,sin(t))

p =

  Line with properties:

            Color: [0 0.4470 0.7410]
        LineStyle: '-'
        LineWidth: 1.5000
           Marker: 'none'
       MarkerSize: 6
  MarkerFaceColor: 'none'
            XData: [1x1257 double]
            YData: [1x1257 double]
            ZData: [1x0 double]

  Show all properties
```

MathWorks

R2014b

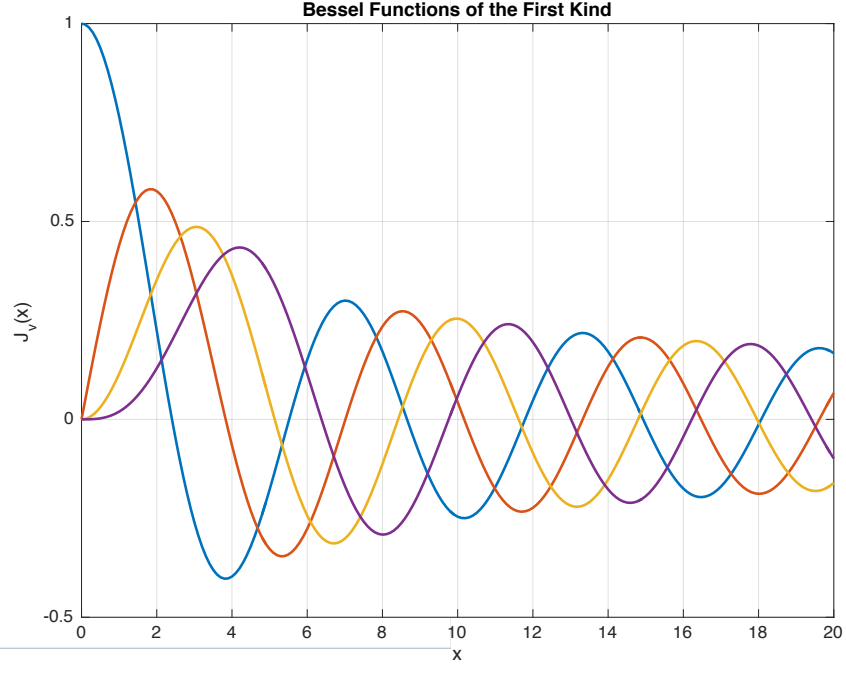
Command Window

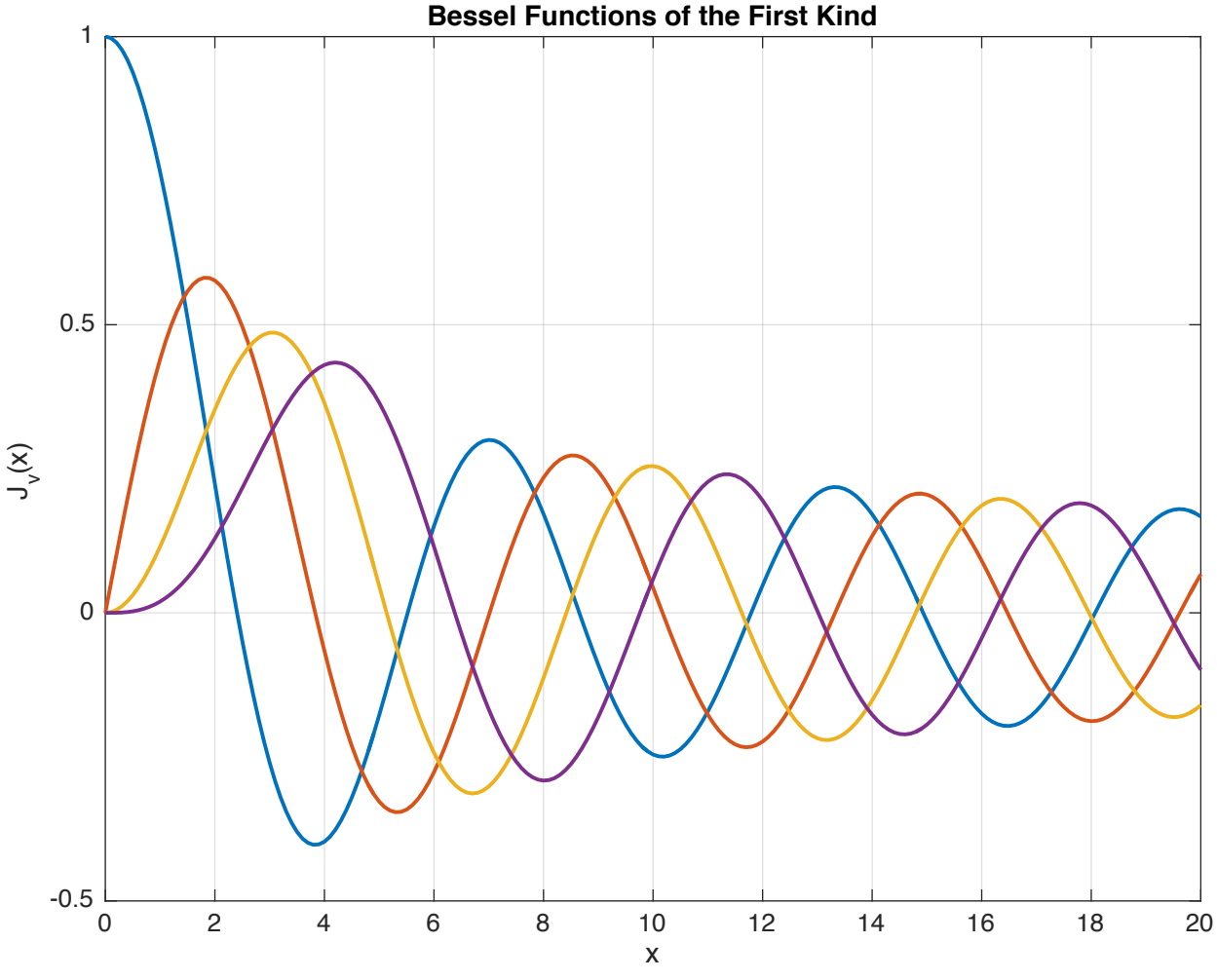
```
>> p.LineStyle = '--';
>> p.LineWidth = 3;
```



Command Window

```
>> x = 0:0.1:20;
>> plot(x,besselj(0,x));
>> grid on
>> title('Bessel Functions of the First Kind')
>> xlabel('x')
>> ylabel('J_v(x)')
>> hold on
>> plot(x,besselj(1,x))
>> plot(x,besselj(2,x))
>> plot(x,besselj(3,x))
>> hold off
```





Antialiased lines, text

Subtle grid color
that doesn't distract

Line colors easier to
distinguish

Better "hold on" behavior

Better title formatting

exponent on axis (2*10-2 instead of 0.02)

Subject: exponent on axis (2*10-2 instead of 0.02)
From: Rachel van Ooteghem
Date: 19 Dec, 2005 11:04:16
Message: 1 of 6

Reply to this message
Add author to My Watch List
View original format
Flag as spam

When I give: "plot([0 1],[0 0.02])" I get a graph where on the y-axis the ticks range from 0 to 0.02.
Due to space limitations I would rather have it like it does with: "plot([0 1],[0 0.002])", where the ticks range from 0 to 2, with an exponent above the y-axis. Matlab does this automatically for smaller values on the axes.
I would like to have a y-axis from 0 to 2 with an exponent of 10^-2.
Is this possible?

"I would like to have a y-axis from 0 to 2 with an exponent of 10^-2"

How do I format tick labels?

Asked by MathWorks Support Team on 5 Aug 2013

Latest activity Edited by MathWorks Support Team on 23 Dec 2014

Accepted Answer by MathWorks Support Team

I have an axis with tick labels and I want them all to have 3.1 format. For example, I do a plot (1:5) and I change my tick labels using the command

```
set(gca, 'XTick', [1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0])
```

However, they appear as:

```
 [1 1.5 2 2.5 3 3.5 4 4.5 5]
```

I would like to get them to appear as I typed them in the above command.

Also, I want to know if it is possible to specify the format of the plot axes tick label which is automatically placed by MATLAB.

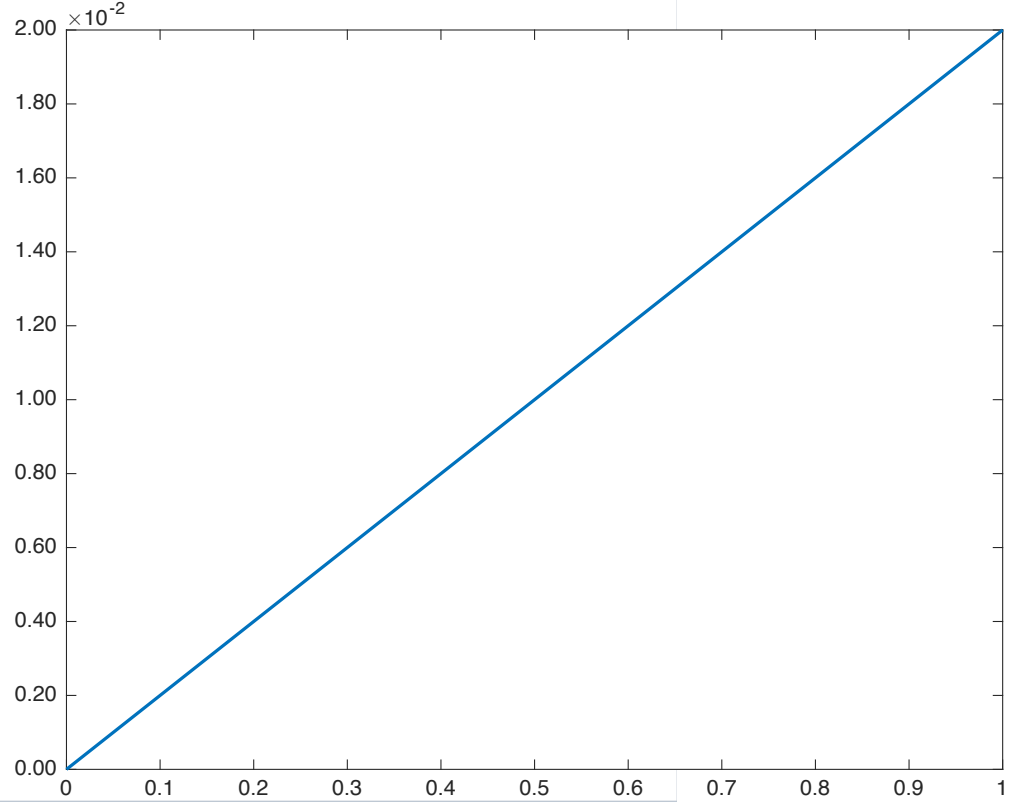
"I want all my axes tick labels to have one fractional digit."

Command Window

```
>> plot([0 1],[0 0.02])
>> ax = gca;
>> yruler = ax.YAxis

yruler =

NumericRuler with properties:

             Limits: [0 0.020000000000000]
              Scale: 'linear'
           Exponent: 0
         TickValues: [1x11 double]
    TickLabelFormat: '%g'

Show all properties
```

Command Window

```
>> yruler.Exponent = -2;
>> yruler.TickLabelFormat = '%4.2f';
```

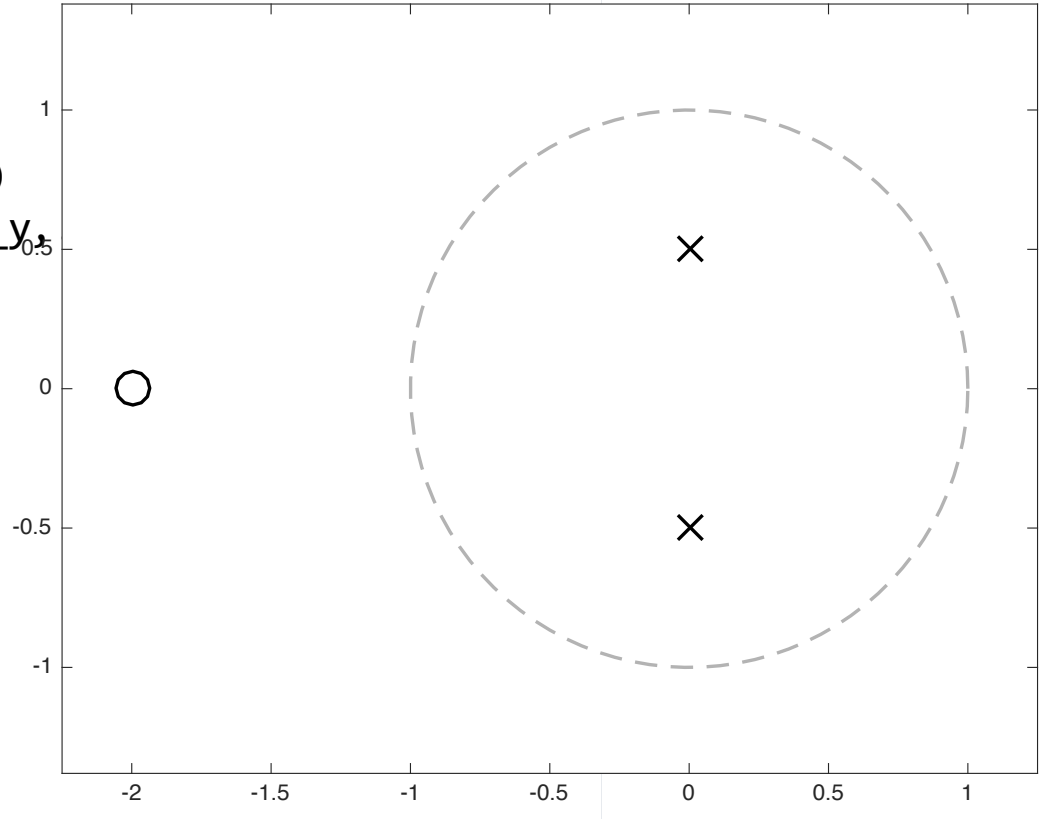


Command Window

```
>> theta = linspace(0,2*pi,100);
>> unit_circle_x = cos(theta);
>> unit_circle_y = sin(theta);
>> plot([0 0],[-0.5 0.5],'xk',...
   'MarkerSize',15)
>> hold on
>> plot(-2,0,'ok','MarkerSize',15)
>> plot(unit_circle_x,unit_circle_y,...
   'Color',[.7 .7 .7],...
   'LineStyle','--');
>> hold off
>> axis([-2.25 1.25 -1.25 1.25])
>> axis equal
```

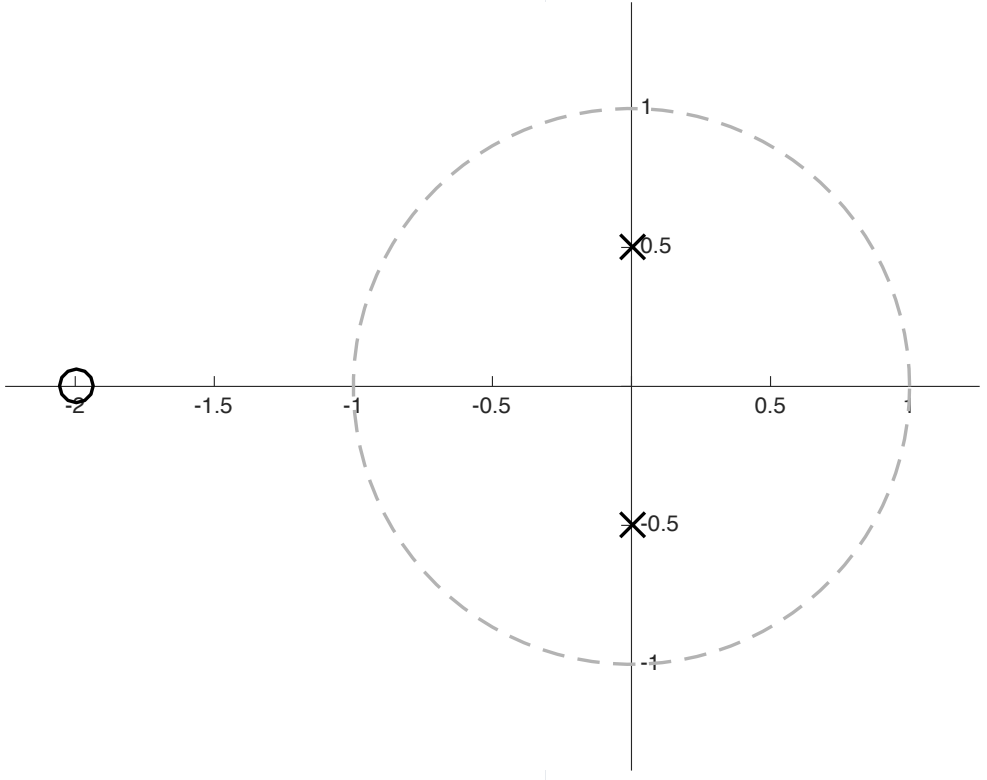
Command Window

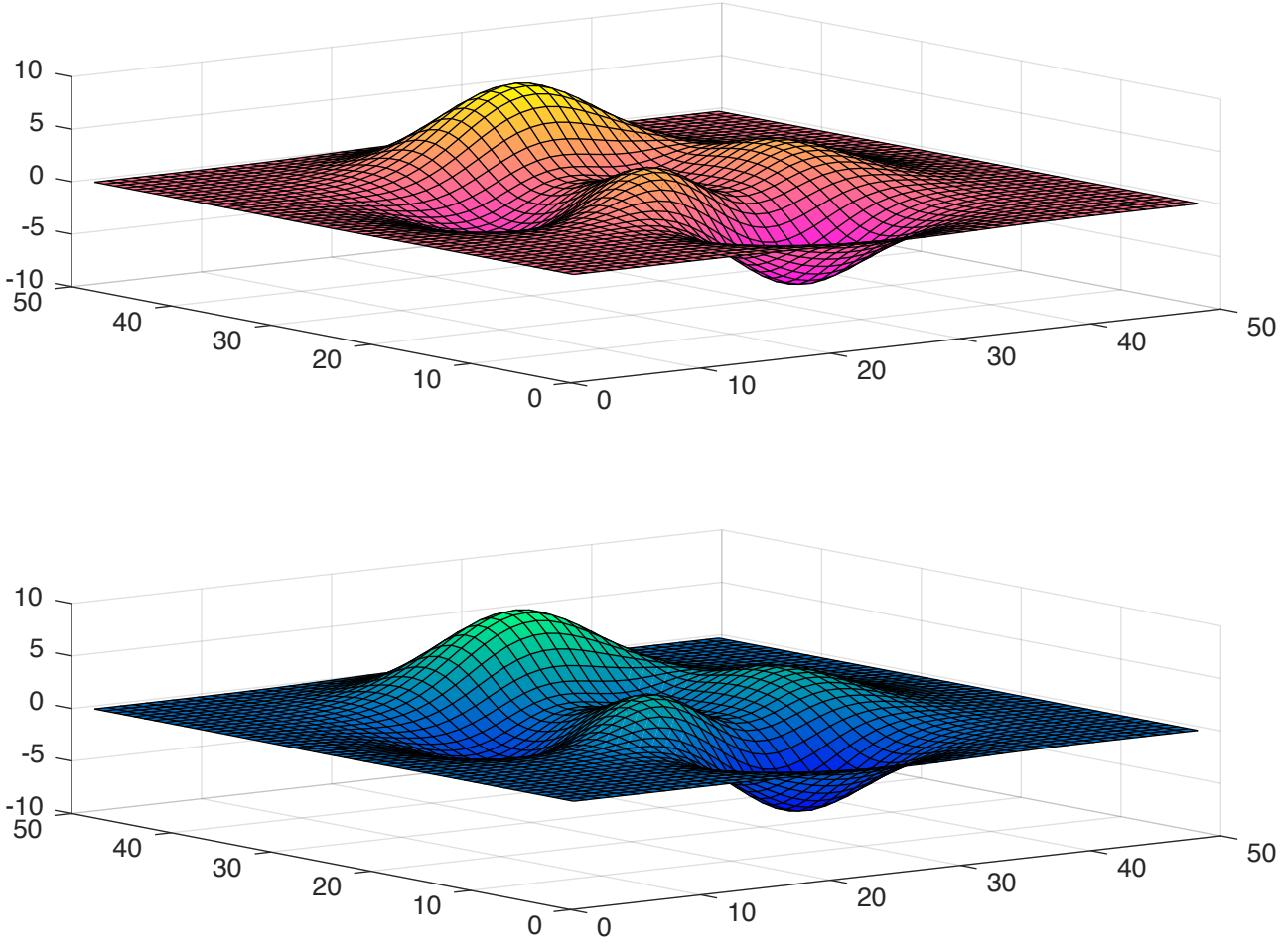
```
>> theta = linspace(0,2*pi,100);
>> unit_circle_x = cos(theta);
>> unit_circle_y = sin(theta);
>> plot([0 0],[-0.5 0.5],'xk',...
   'MarkerSize',15)
>> hold on
>> plot(-2,0,'ok','MarkerSize',15)
>> plot(unit_circle_x,unit_circle_y,
   'Color',[.7 .7 .7],...
   'LineStyle','--');
>> hold off
>> axis([-2.25 1.25 -1.25 1.25])
>> axis equal
```

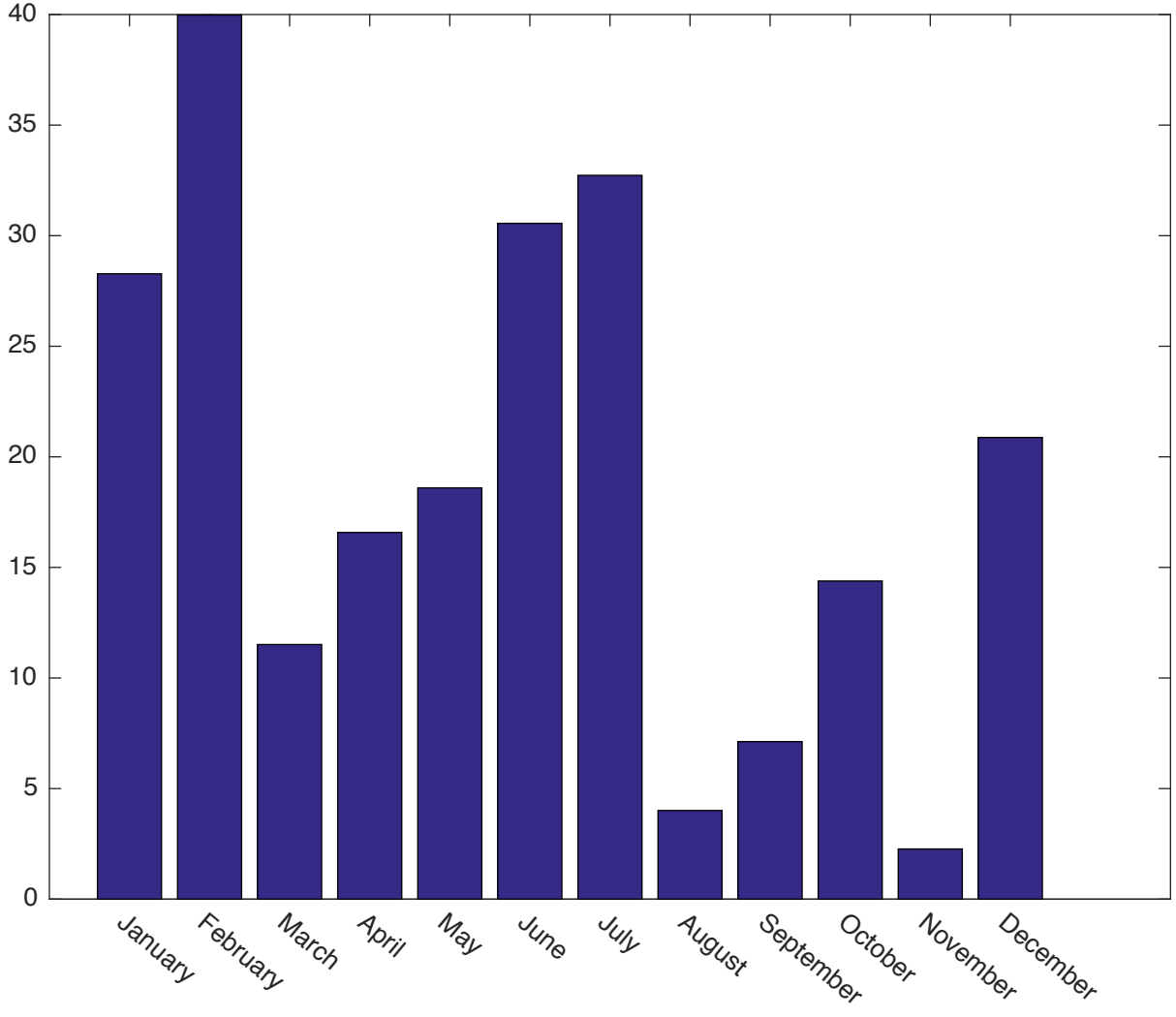


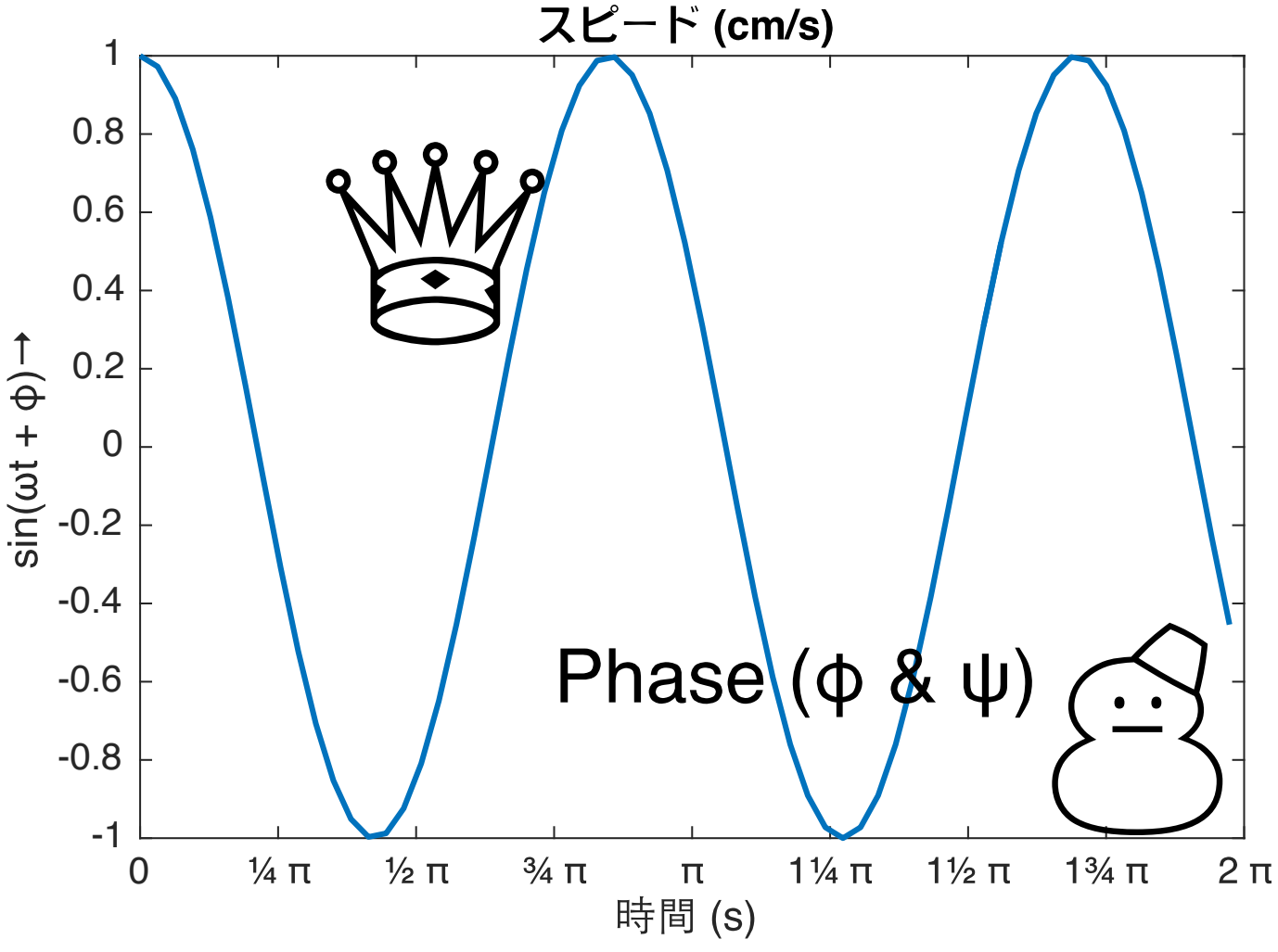
Command Window

```
>> ax = gca;
>> ax.XAxisLocation = 'origin';
>> ax.YAxisLocation = 'origin';
box off
```




10
5
0
-5
-10
50
40
30
20
10
0
0
10
20
30
40
50
10
5
0
-5
-10
50
40
30
20
10
0
0
10
20
30
40
50


40
35
30
25
20
15
10
5
0
January
February
March
April
May
June
July
August
September
October
November
December


スピード (cm/s)
sin(ωt + Φ) →
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
¼ π
½ π
¾ π
π
1¼ π
1½ π
1¾ π
2 π
時間 (s)
Phase (ϕ & ψ)

What's a Parula?

Image credit: Dario Sanches. Cropped by Steve Eddins

The "jet" colormap
MATLAB default from 1997 – 2014
Also known as a "rainbow" colormap

IEEE Computer Graphics and Applications, 2007
Borland & Taylor II

Rainbow Color Map (Still) Considered Harmful

Research has shown that the rainbow color map is rarely the optimal choice when displaying data with a pseudocolor map. The rainbow color map confuses viewers through its lack of perceptual ordering, obscures data through its uncontrolled luminance variation, and actively misleads interpretation through the introduction of non-data-dependent gradients.

Despite much published research on its deficiencies, the rainbow color map is prevalent in the visualization community. We present survey results showing that the rainbow color map continues to appear in more than half of the relevant papers in IEEE Visualization Conference proceedings; for example, it appeared on 61 pages in 2005. Its use is encouraged by its selection as the default color map used in most visualization toolkits that we inspected. The visualization community must do better.

In this article, we reiterate the characteristics that mercials, weather forecasts, and even the IEEE Visualization Conference 2006 call for papers, just to name a few. The problem with this wide use of the rainbow color map is that research shows that it is rarely, if ever, the optimal color map for a given visualization.[1-6] Here we will discuss the rainbow color map's characteristics of confusing the viewer, obscuring data, and actively misleading interpretation.

Confusing

For all tasks that involve comparing relative values, the color map used should exhibit perceptual ordering. A simple example of a perceptually ordered color map is the gray-scale color map. Increasing luminance from black to white is a strong perceptual cue that indicates values mapped to darker shades of gray are lower in value than values mapped to lighter shades of gray. This mapping is natural and intuitive.

IEEE Spectrum, 1998
Rogowitz & Treinish

Data visualization: the end of the rainbow

At the core of all good science and engineering is the respectful treatment of data: instruments are calibrated, algorithms are scrutinized, and the behavior of analytical or simulation models studied, on the assumption that the tools suit the data at hand.

But the knowledge to use one set of tools, data visualization software, is often lacking. Visualization should help make sense of the flood of output data. When applied without some insight into visual perception, however, it can introduce errors in understanding as surely as if a wrong analysis algorithm were used. In short, a picture can tell a thousand lies.

Blind spots in visualization are inevitable if attention is not paid to how the human visual system processes information, to the nature of the data, and to how the data are to be used. But once identified, these factors can be manipulated to good effect, as a variety of visualizations drawn from fields as diverse as geography, medicine, and physics will show.

The key is the colormap, which may be defined as a mapping from a data value to a color. (The term colormap is also sometimes used in other contexts specifically for the contents of the memory locations governing display.)

A rainbow colormap is often supplied as a default in visualization software, the vendor wishing perhaps to provide the greatest possible range of colors with which to work. In this kind of colormap, red is mapped to the highest data value, blue to the lowest, and the other data values are interpolated along the full extent of the spectrum. An example would be a temperature profile mapped over a land mass on a weather map.

But there is more to color than meets the eye. Color, after all, is a perceptual as well as physical phenomenon. What is commonly called color—hue—is only one of three parameters. Another is the brightness of the signal—intensity. The third is the admixture of white—saturation. Change

BERNICE E. ROGOWITZ & LLOYD A. TREINISH
IBM Corp., Thomas J. Watson Research Center

The End of the Rainbow? Color Schemes for Improved Data Graphics

PAGES 385, 391

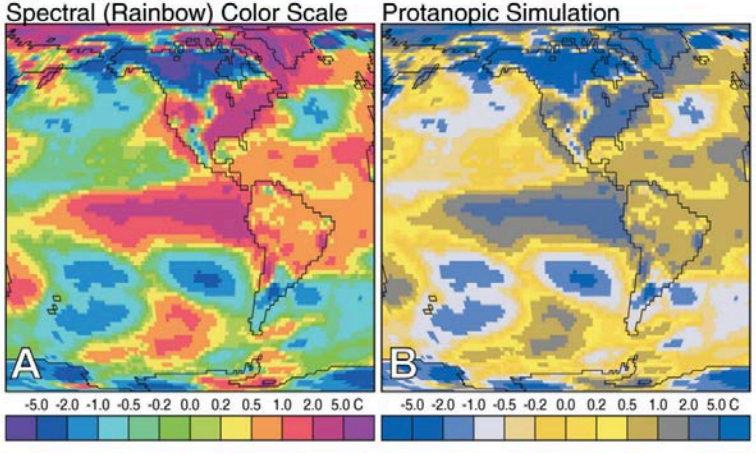
Modern computer displays and printers enable the widespread use of color in scientific communication, but the expertise for designing effective graphics has not kept pace with the technology for producing them. Historically, even the most prestigious publications have tolerated high defect rates in figures and illustrations [*Cleveland*, 1984], and technological advances that make creating and reproducing graphics easier do not appear to have decreased the frequency of errors. Flawed graphics consequently beget more flawed graphics as authors emulate published examples. Color has the potential to enhance communication, but design mistakes can result in color figures that are less effective than gray scale displays of the same data.

Empirical research on human subjects can build a fundamental understanding of visual perception [*Ware*, 2004] and scientific methods can be used to evaluate existing designs, but creating effective data graphics is a design task and not fundamentally a scientific pursuit. Like writing well, creating good data graphics requires a combination of formal knowledge and artistic sensibility tempered by experience: a combination of "substance, statistics, and design" [*Tufte*, 1983, p. 51].

Unlike writing, however, proficiency in creating data graphics is not a main component of secondary or postsecondary education. This article provides some concrete suggestions to one spectral color scheme, and a better alternative, may appear to color-blind readers.

Color-blind individuals see some colored data graphics quite differently from the general population. The human visual system normally perceives color through photosensitive cones in the eye that are tuned to receive wavelengths in the red, green, and blue portions of the visible spectrum. People who lack cones sensitive to one of the three wavelengths are called dichromats [*Fortner and Meyer*, 1997]. Individuals whose receptors are shifted toward one or the other end of the spectrum are called anomalous trichromats. The term "color deficient" encompasses dichromats and anomalous trichromats, as well as those who exhibit rarer forms of impaired color vision. Because a sex-linked recessive gene is implicated in the condition, color-vision deficiency is far more common in men than in women.

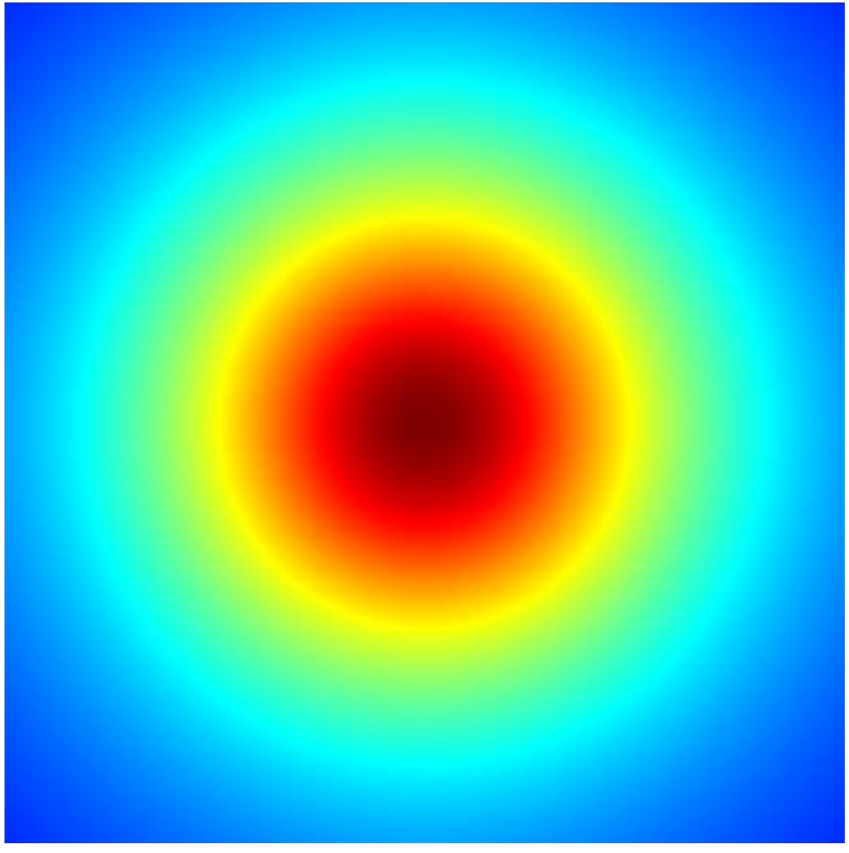
Algorithms based on psychophysical observations make it possible to simulate the appearance of colored images to color-deficient viewers [*Brettel et al.*, 1997]. Data maps of the type shown here serve two main purposes: detection of large-scale patterns and determination of specific grid-cell or point values. The saturated spectral scale (Figure 1a) creates a region of confusion centered on the North American continent where achieving either purpose becomes nearly impossible for dichromat

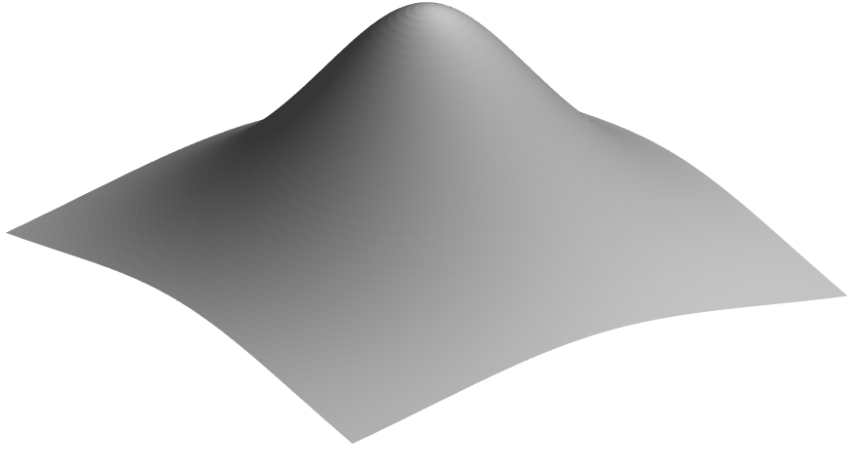


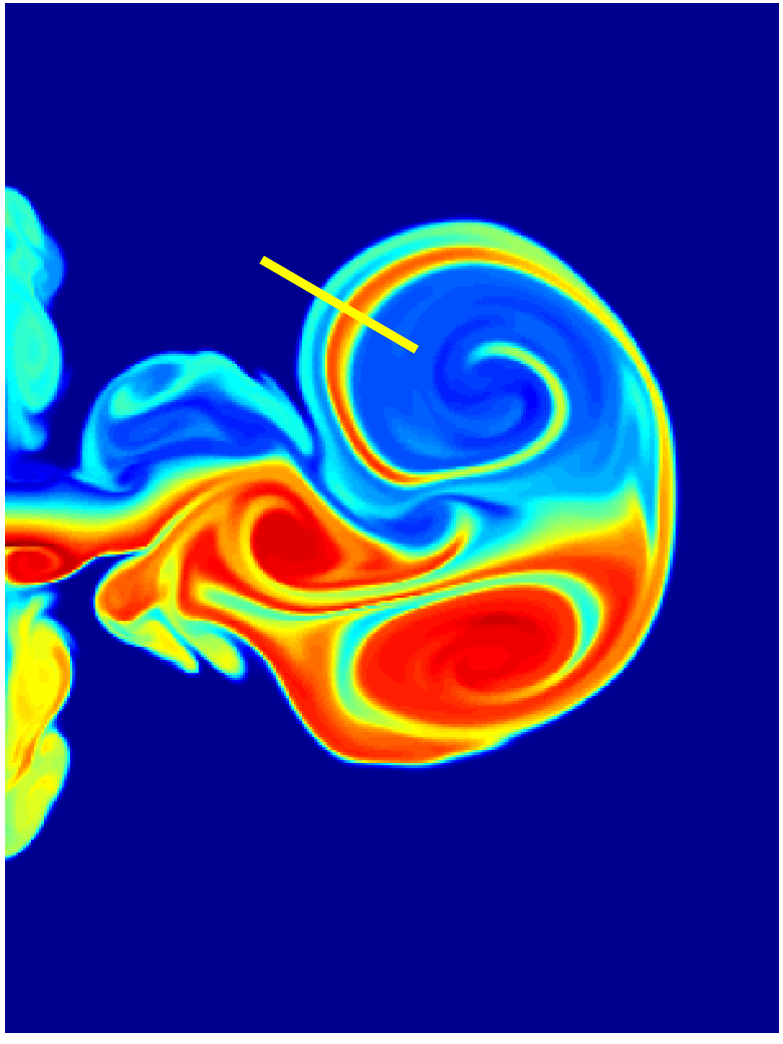
EOS, Transactions of American Geophysical Union, 2004
Light &Bartlein

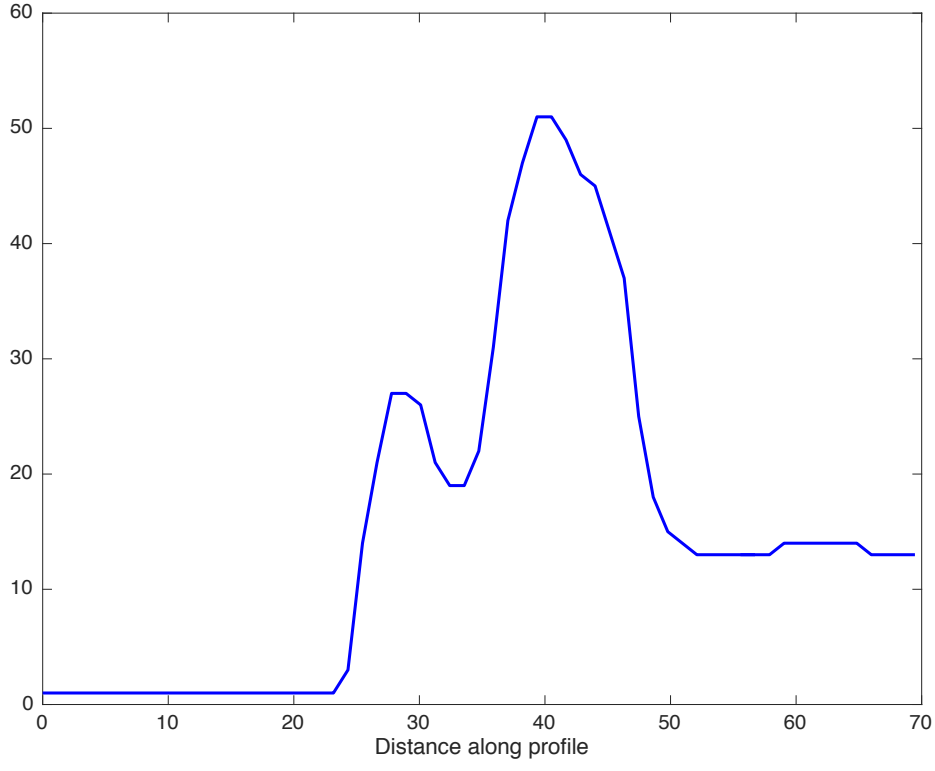
Rainbow criticisms

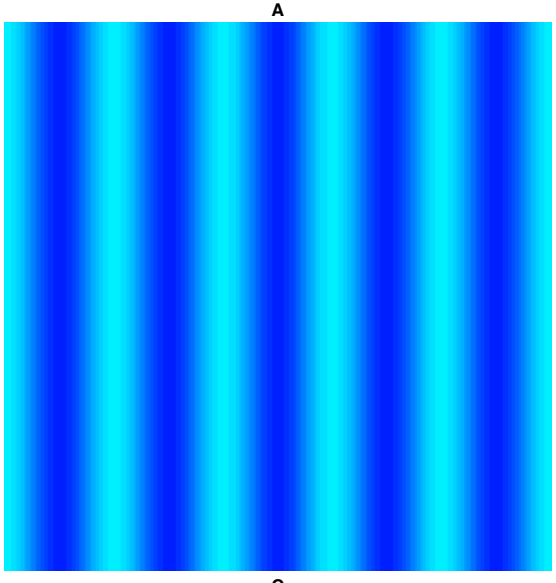
- Confuses viewers because lack of perceptual ordering of spectral colors
- Obscures data detail in some parts of colormap
- Conveys false data detail in other parts of colormap
- Causes difficulty for some color-impaired viewers
- Prints poorly to gray-scale devices

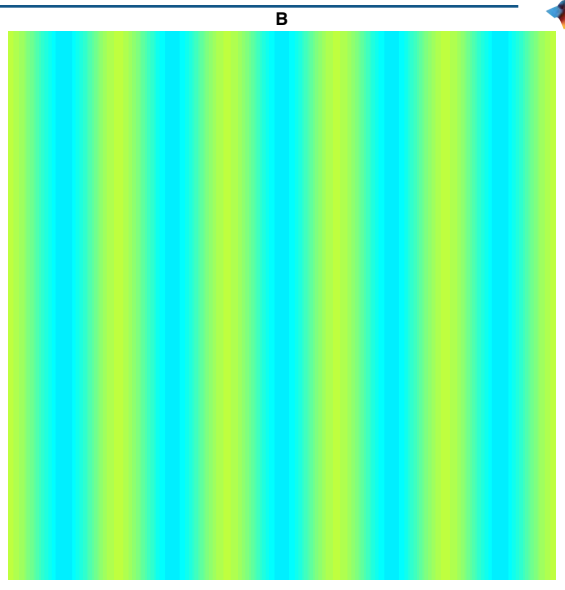


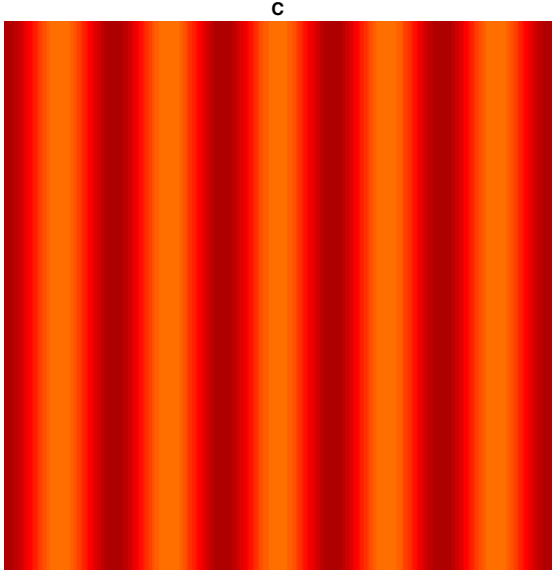


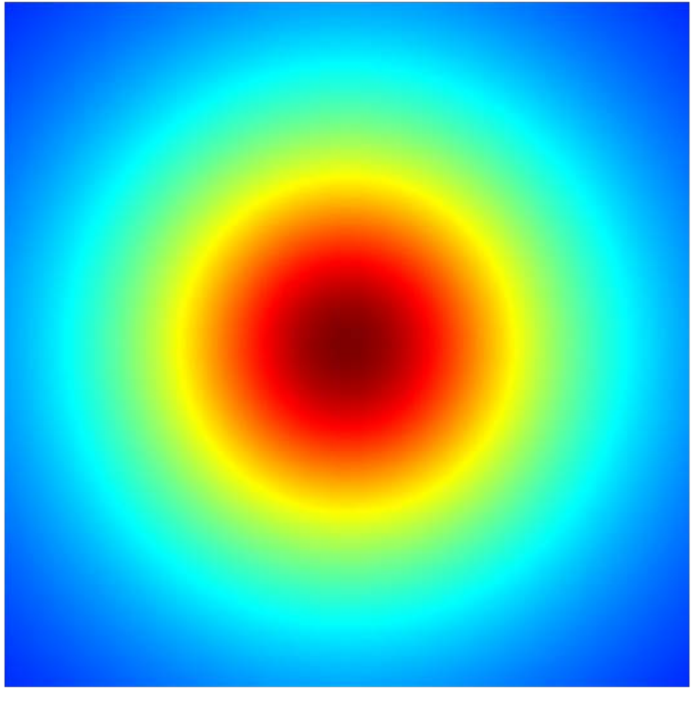


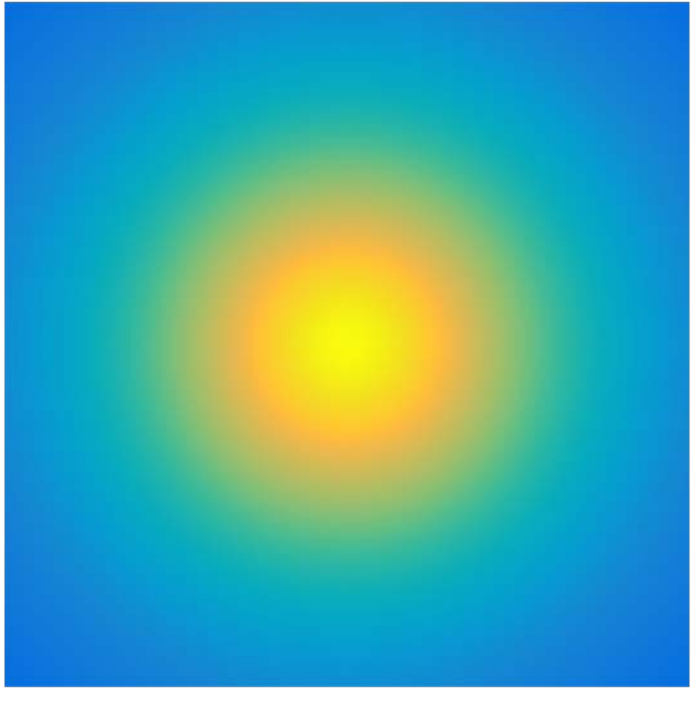

60
50
40
30
20
10
0
0
10
20
30
40
50
60
70
Distance along profile

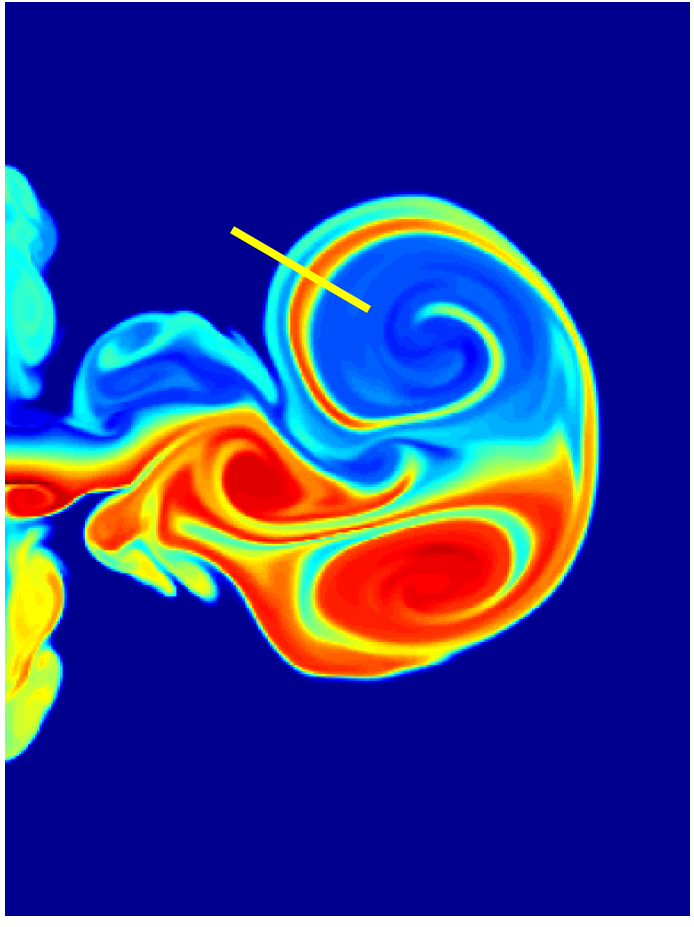

A

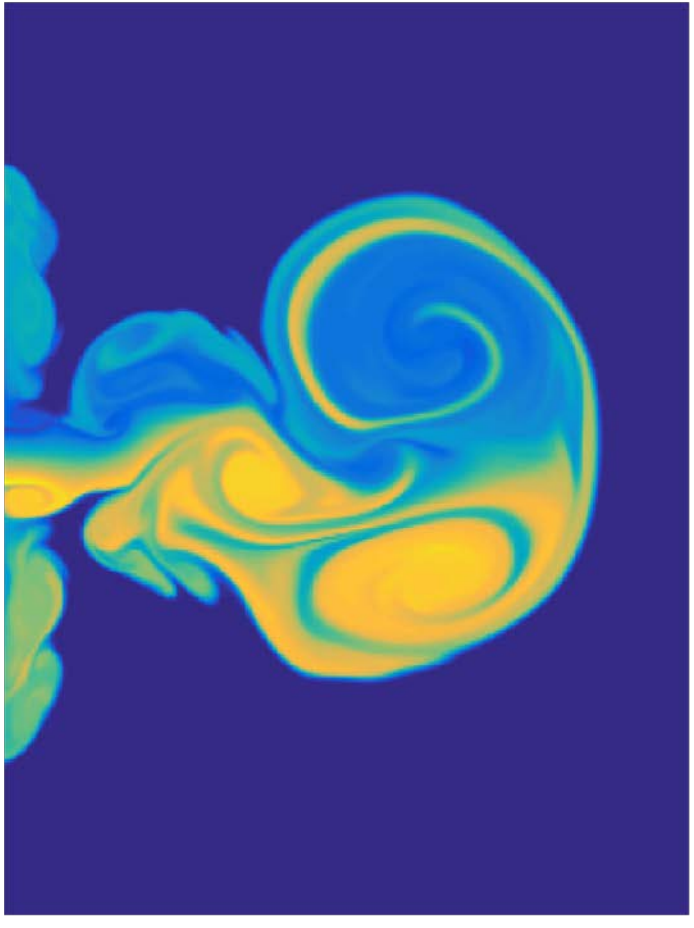

B

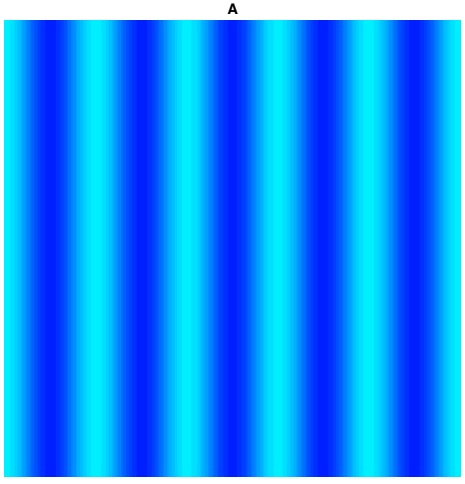

C

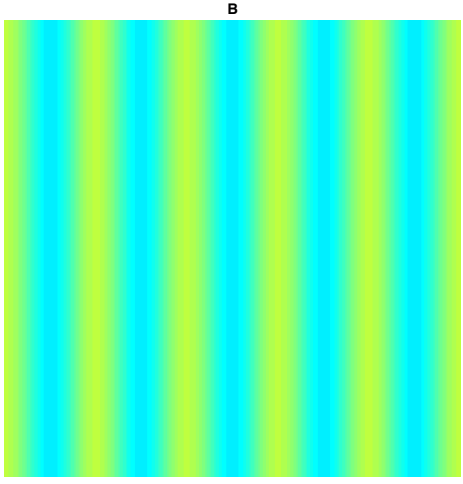


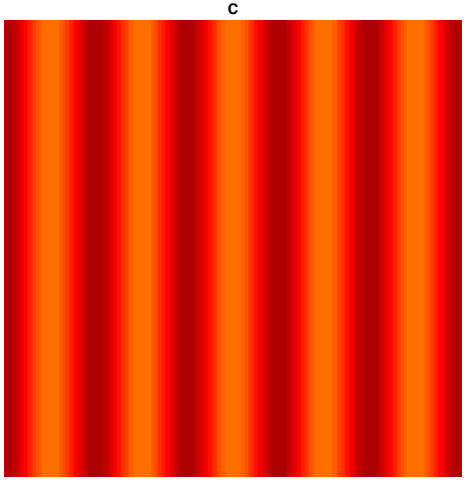




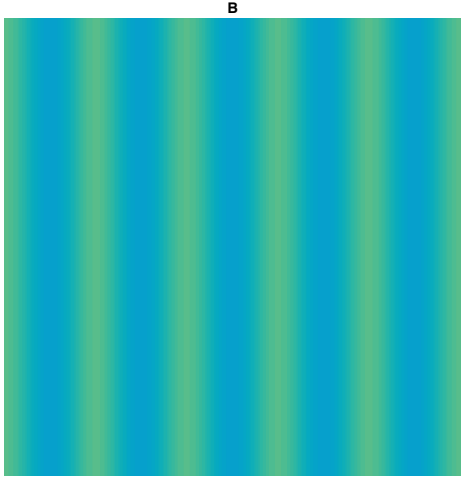


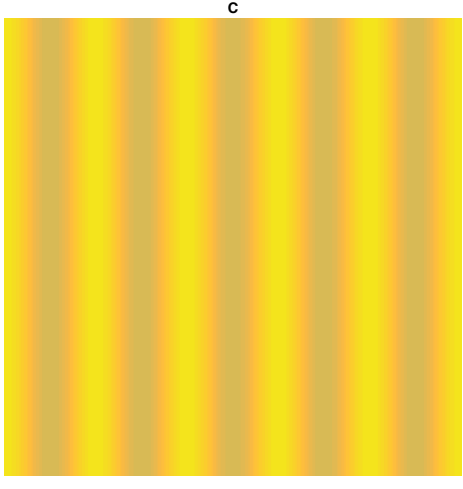

A


B

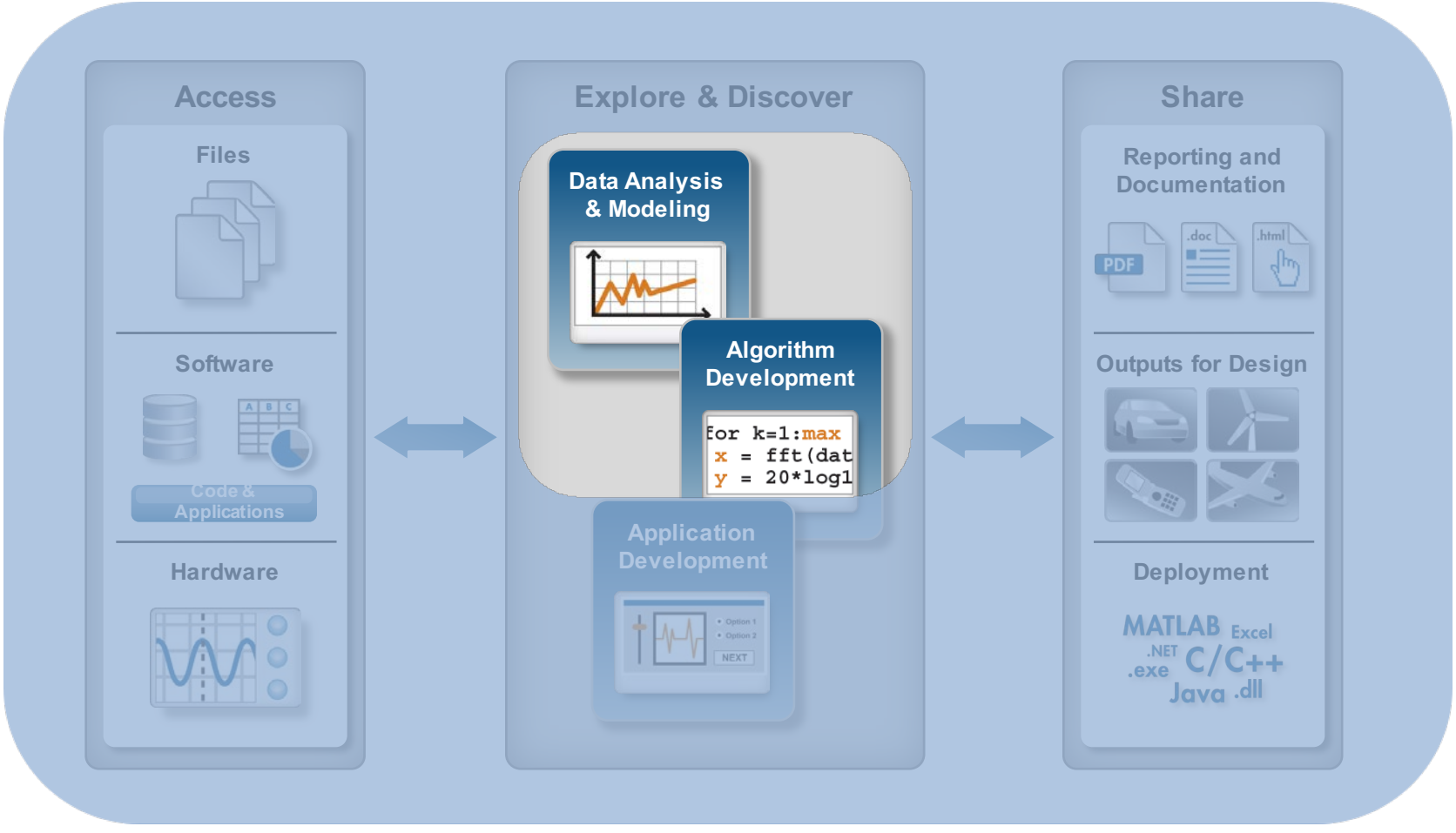

C


A


B


C

MATLAB Math


Access
Files
Software
Code & Applications
Hardware
Explore & Discover
Data Analysis & Modeling
Algorithm Development
for k=1:max
x = fft(dat
y = 20*log1
Application Development
Option 1
Option 2
NEXT
Share
Reporting and Documentation
PDF
.doc
.html
Outputs for Design
Deployment
MATLAB
Excel
.NET
.exe
C/C++
Java
.dll

MATLAB Math Topics

- Histograms
- Alpha shapes
- Graphs

Histograms: A New Project Begins

- Peter

  "Hey, you know we could really be doing a lot better with histograms in MATLAB."

- The rabble

  "What do you mean? **`hist`** has been in MATLAB since the dawn of time. What's the problem?"

- Peter (later, and at regular intervals)

  "Hey, you know we could really be doing a lot better with histograms in MATLAB."

- The rabble (eventually)

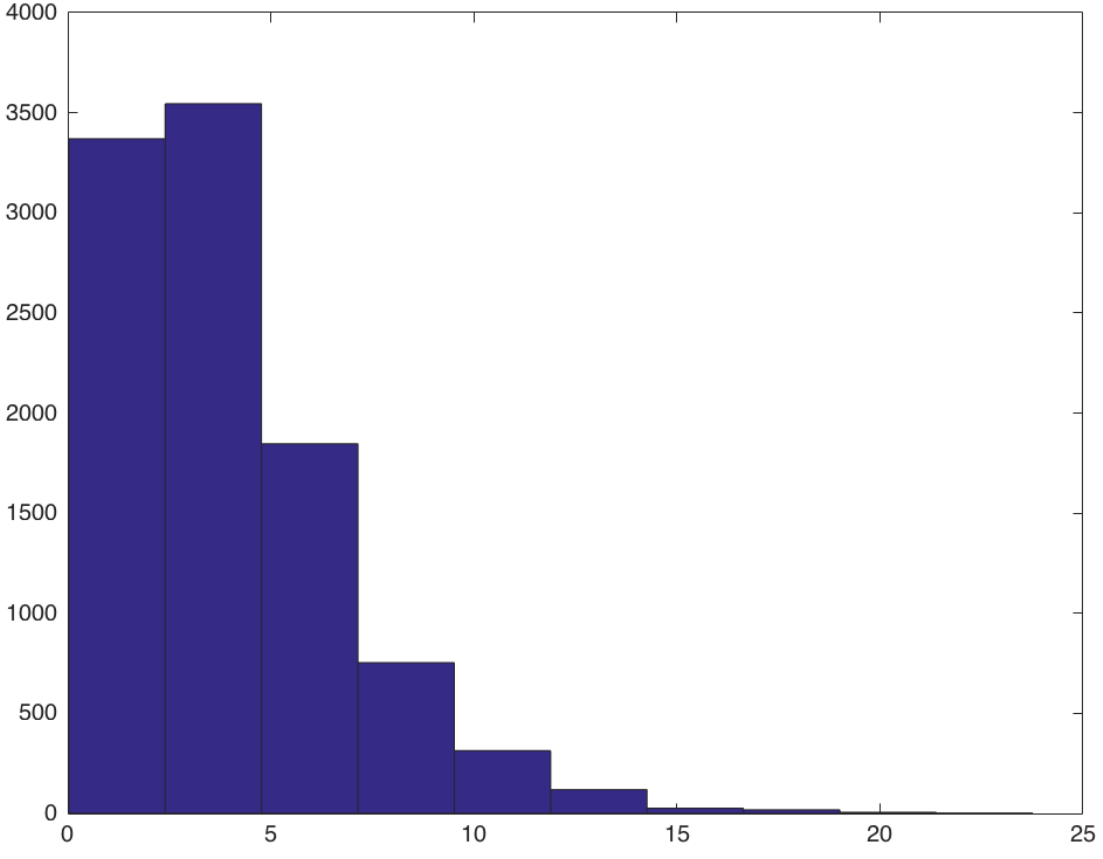
  "OK, let's do some research on this."

Issues with `hist`

- Confusion over hist and histc
- Default selection of 10 bins
- Floating-point input only
- Awkward behavior at left-most and right-most bin edges
- Histogram chart looks bad for multiple histograms
- Modifying histogram chart appearance too difficult
- Normalization computation subtle, mistake-prone

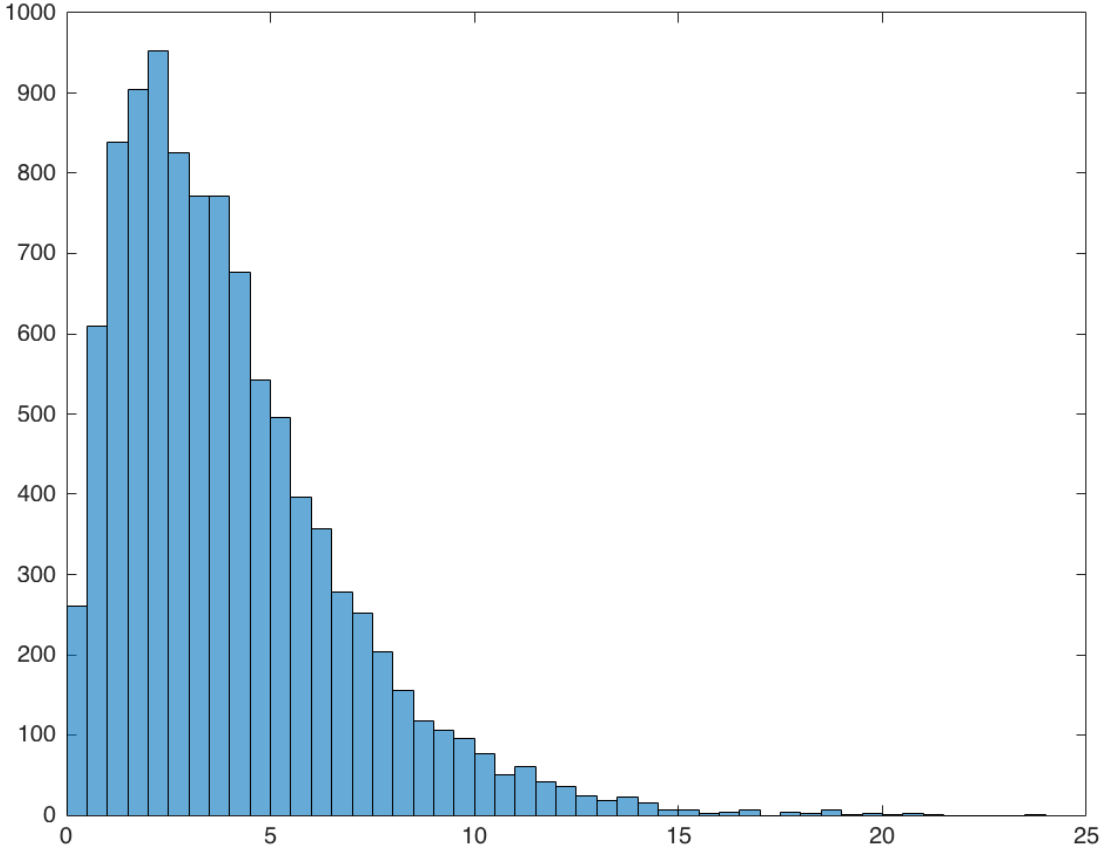
Command Window

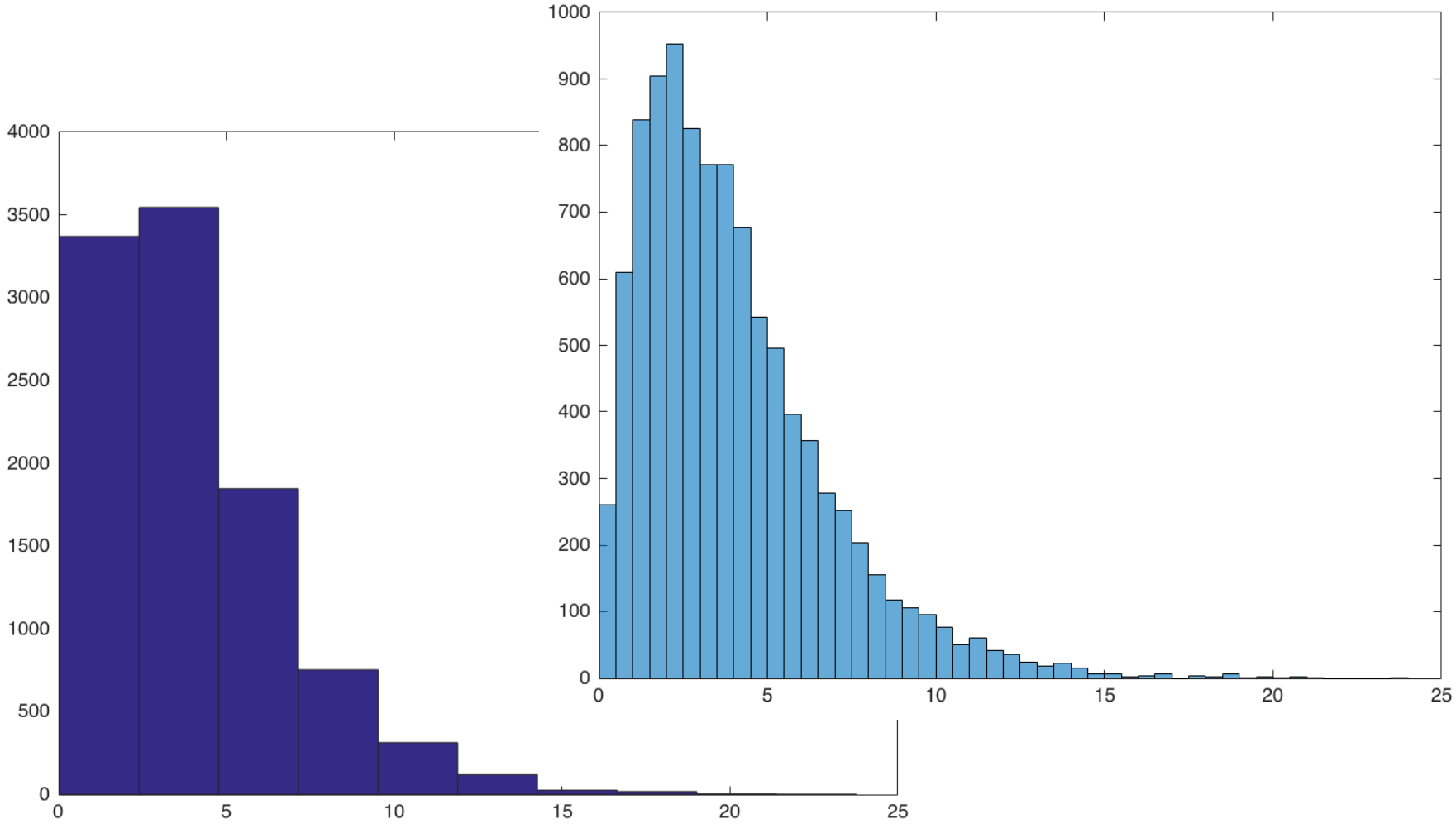
```
>> data = random('chi2',4,10000,1);
>> hist(data)
```



Command Window

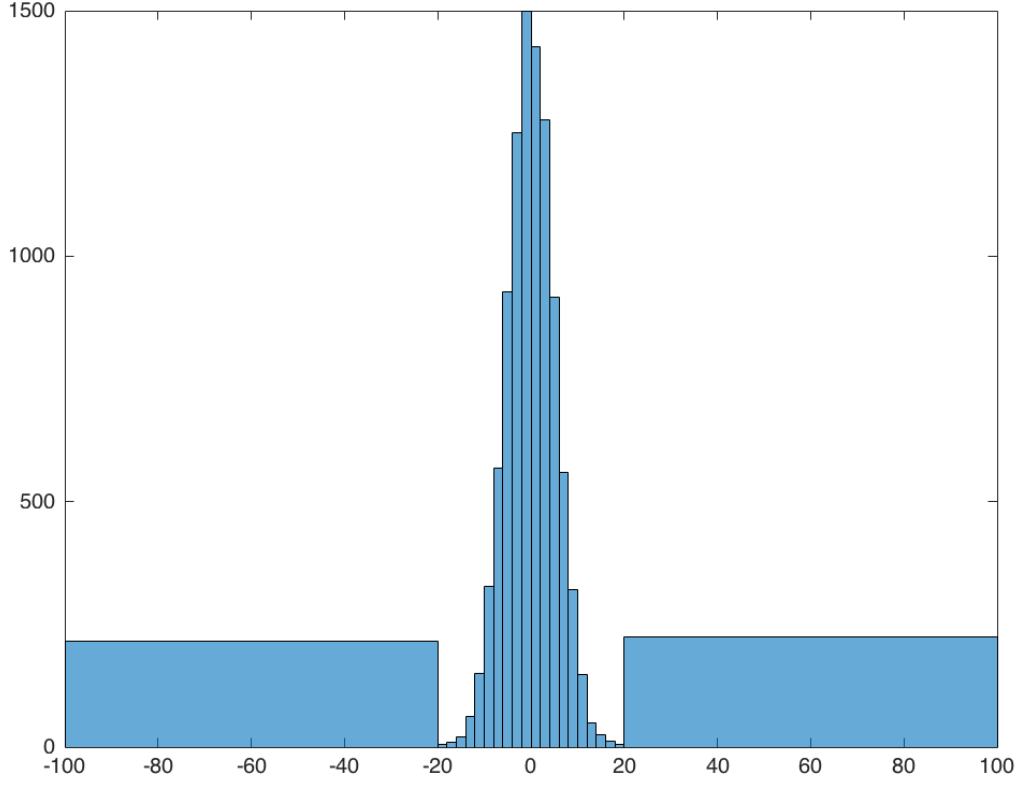
```
>> data = random('chi2',4,10000,1);
>> histogram(data)
```




4000
3500
3000
2500
2000
1500
1000
500
0
0
5
10
15
20
25
1000
900
800
700
600
500
400
300
200
100
0
0
5
10
15
20
25

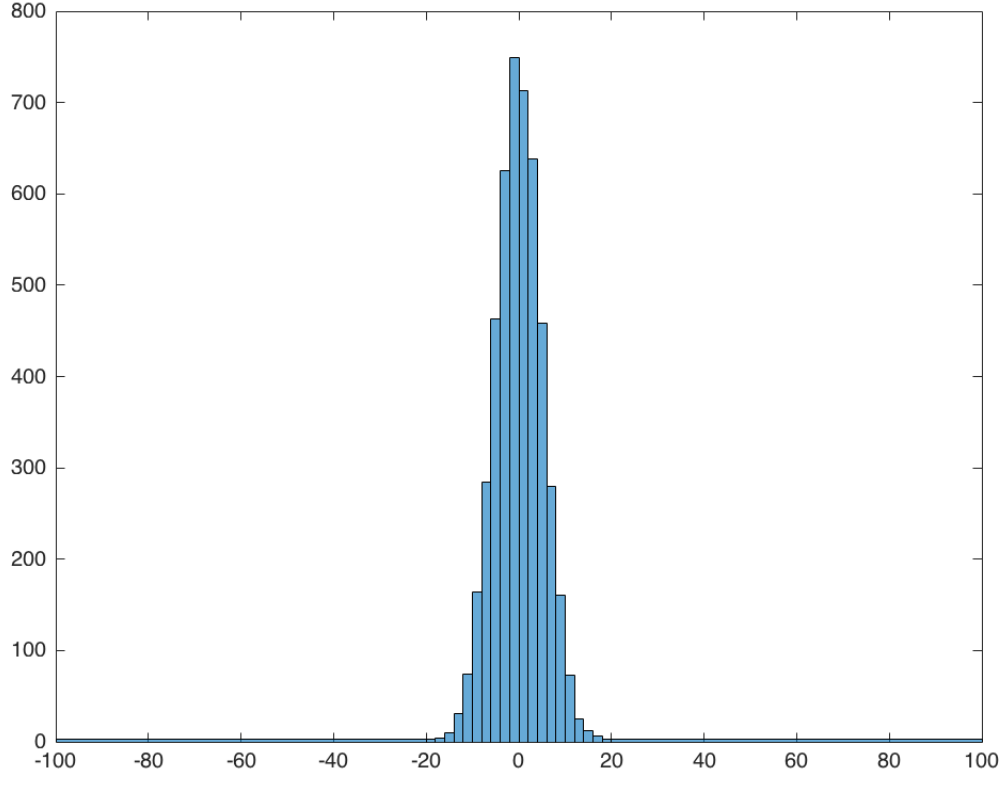
Command Window

```
>> bin_edges = [-100 (-20:2:20) 100];
>> histogram(data,bin_edges)
```



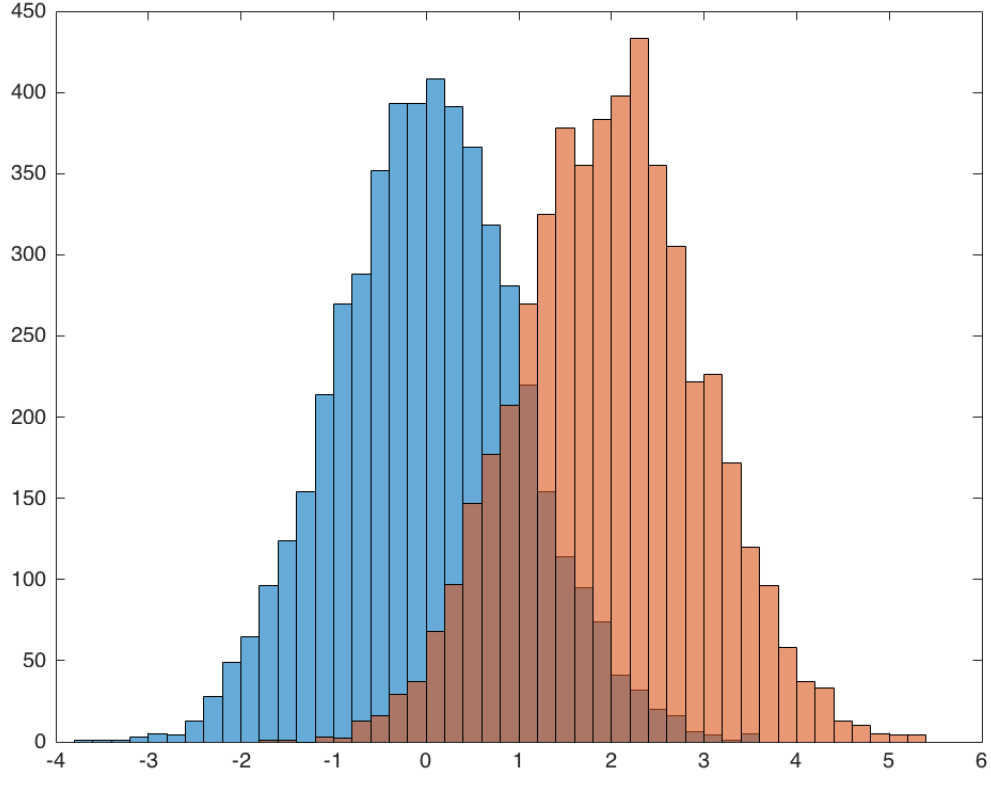
Command Window

```
>> bin_edges = [-100 (-20:2:20) 100];
>> histogram(data,bin_edges,'Normalization','countdensity')
```



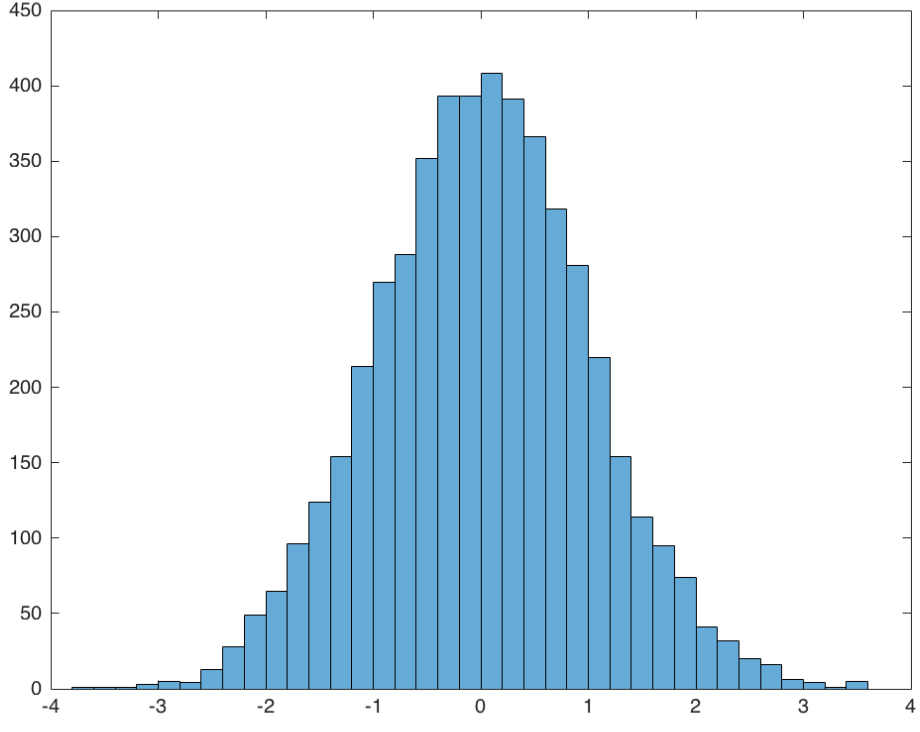
Command Window

```
>> hist([data1 data2])
>> histogram(data1)
>> hold on
>> histogram(data2)
>> hold off
```



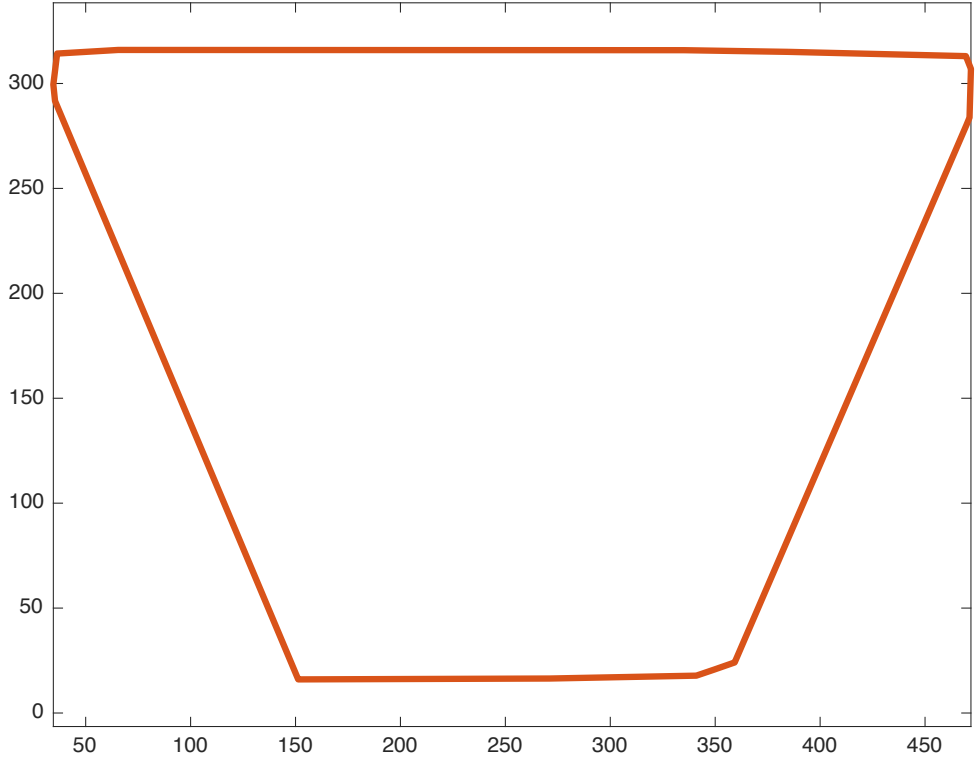
Command Window

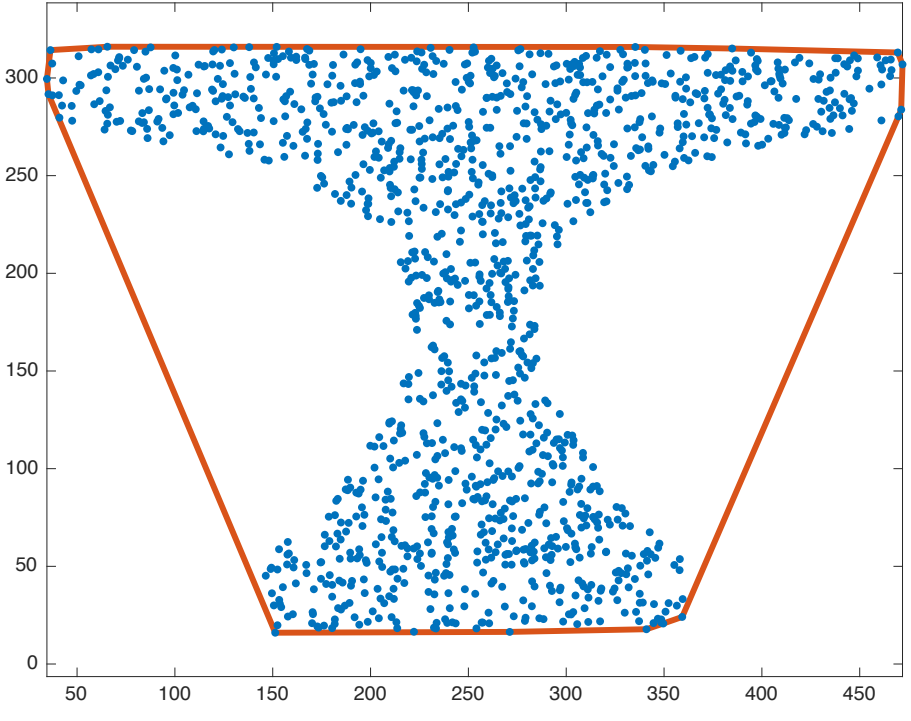
```
>> h = histogram(data1)

h =   Histogram with properties:

            Data: [5000x1 double]
          Values: [1x37 double]
         NumBins: 37
        BinEdges: [1x38 double]
        BinWidth: 0.2000
       BinLimits: [-3.8000 3.6000]
   Normalization: 'count'
       FaceColor: 'auto'
       EdgeColor: [0 0 0]

  Show all properties
```

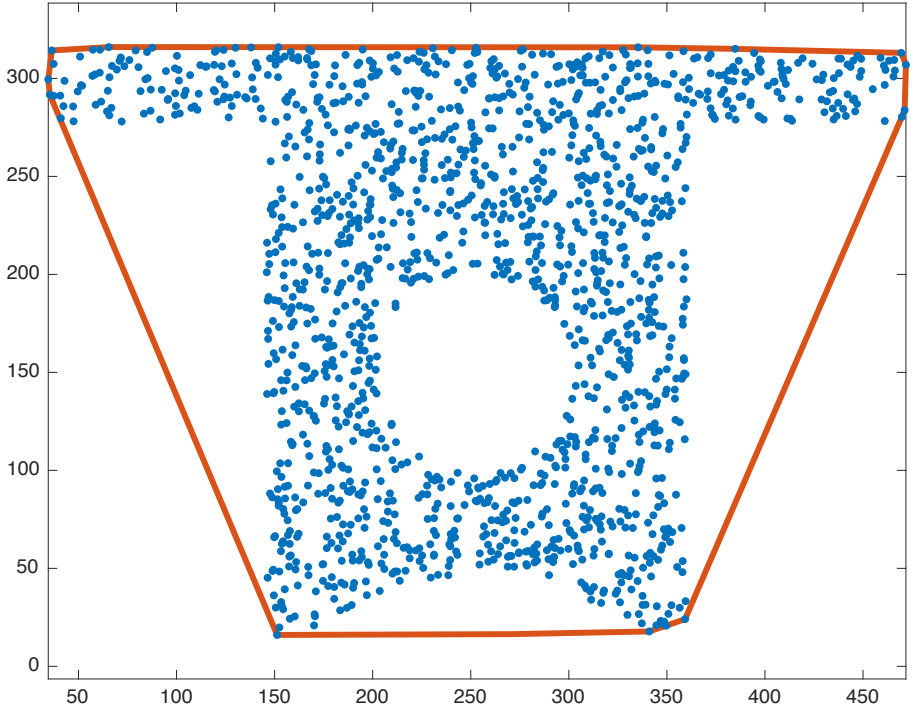


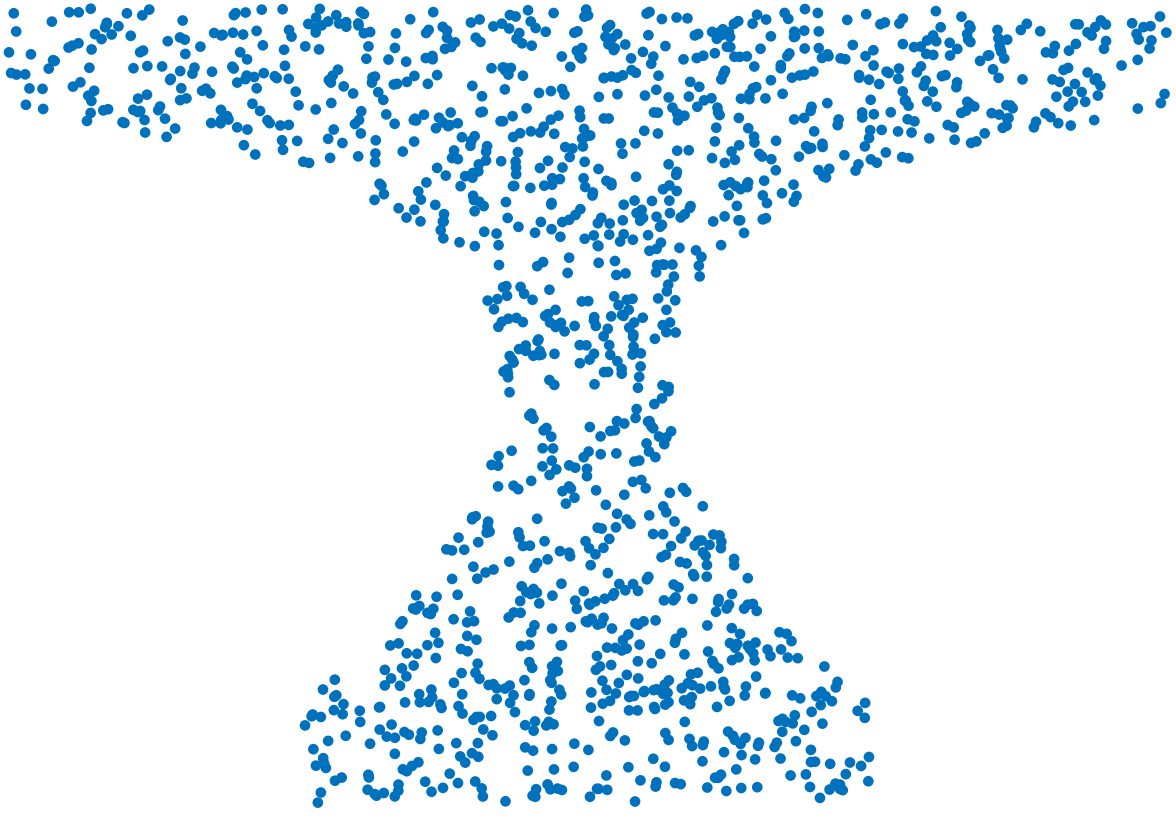
Alpha Shapes

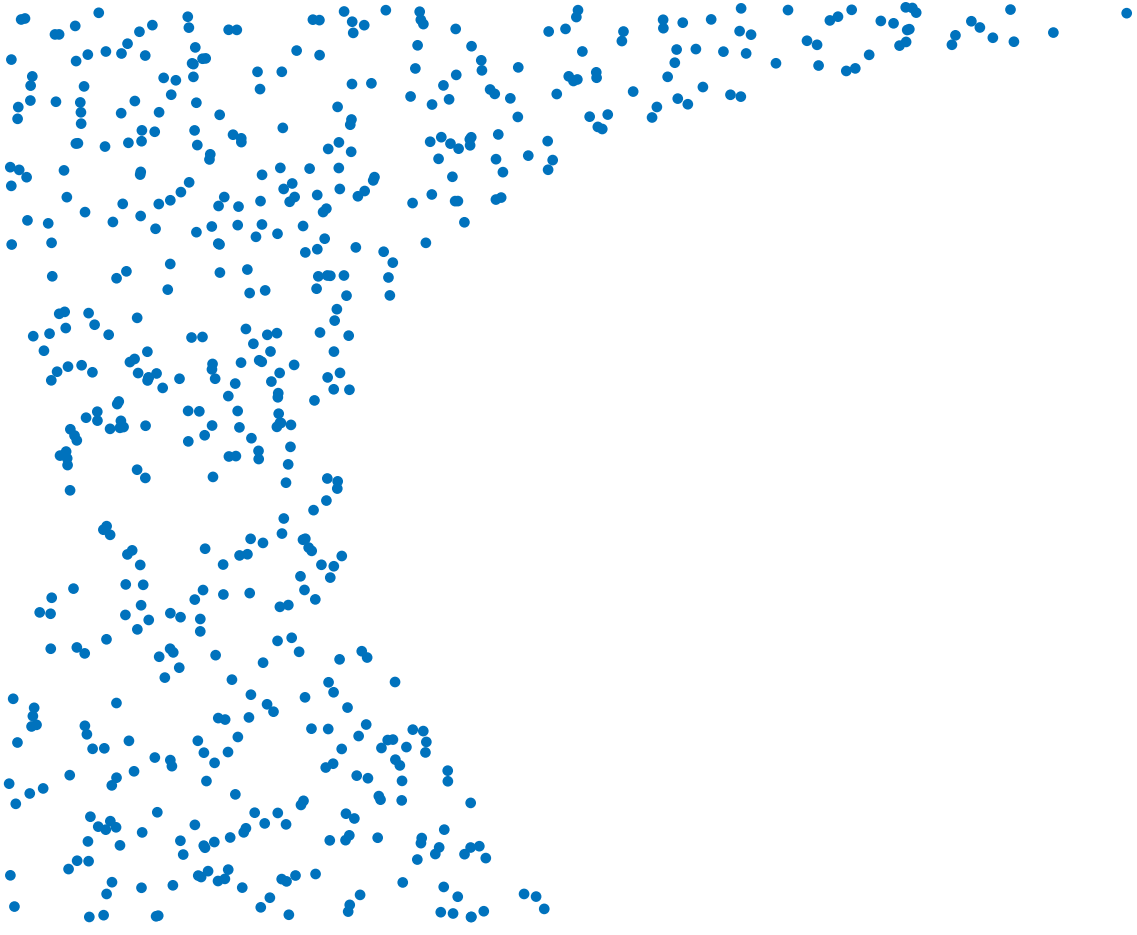
Here's a convex hull:

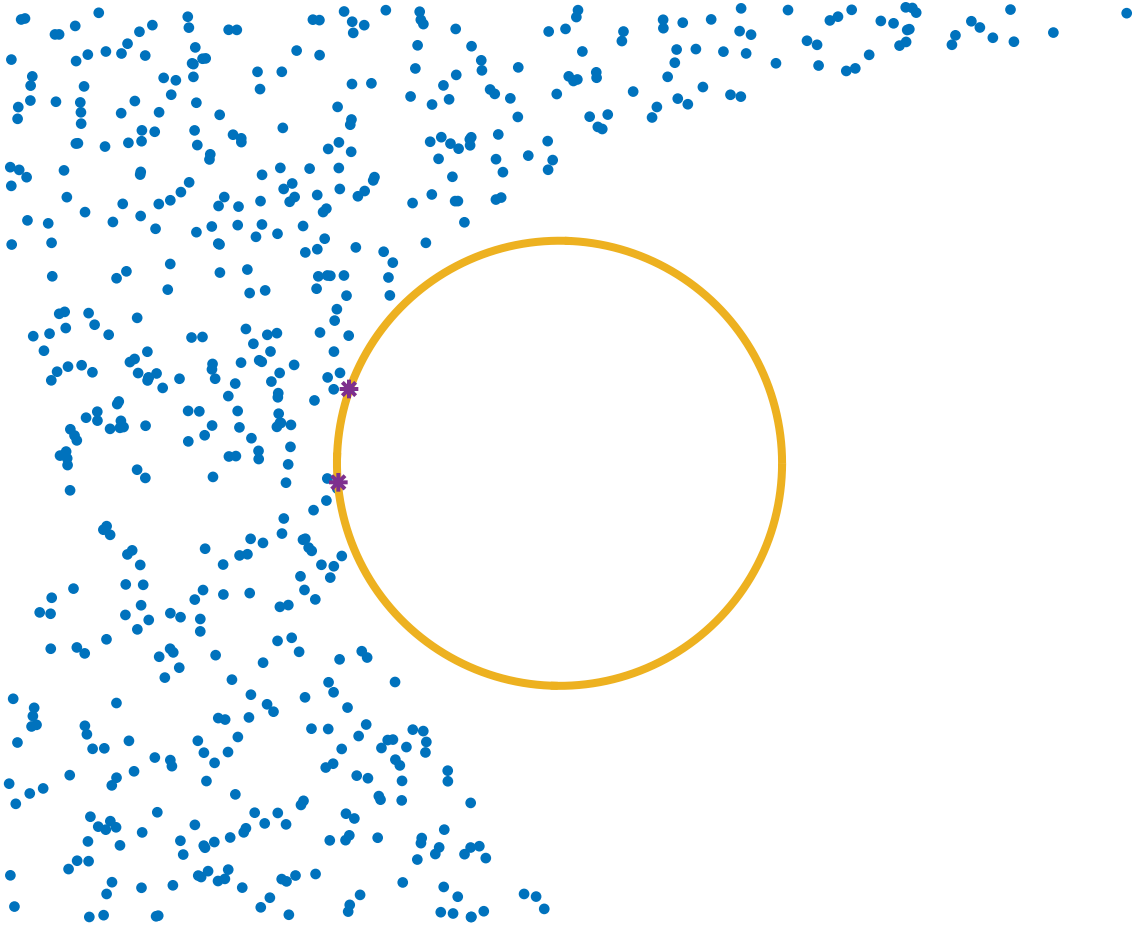


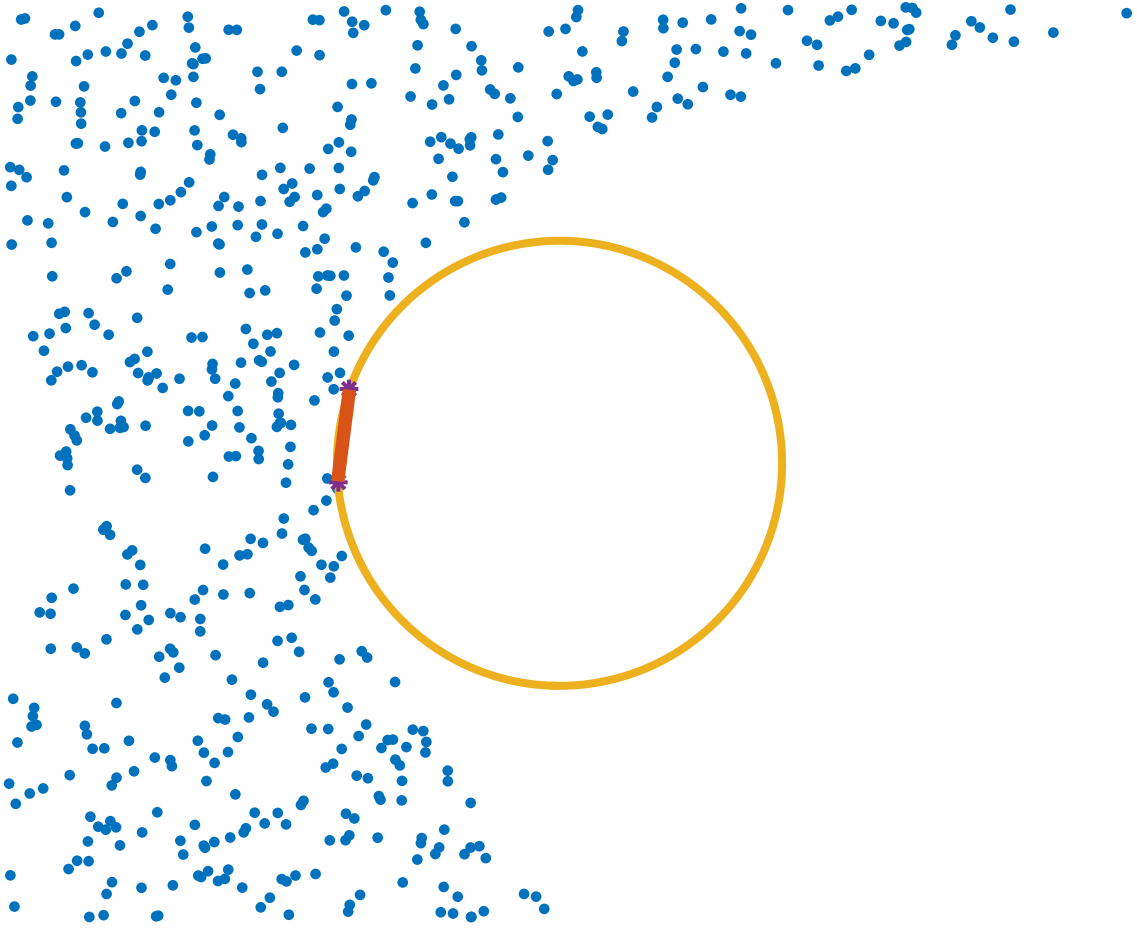

300
250
200
150
100
50
0
50
100
150
200
250
300
350
400
450

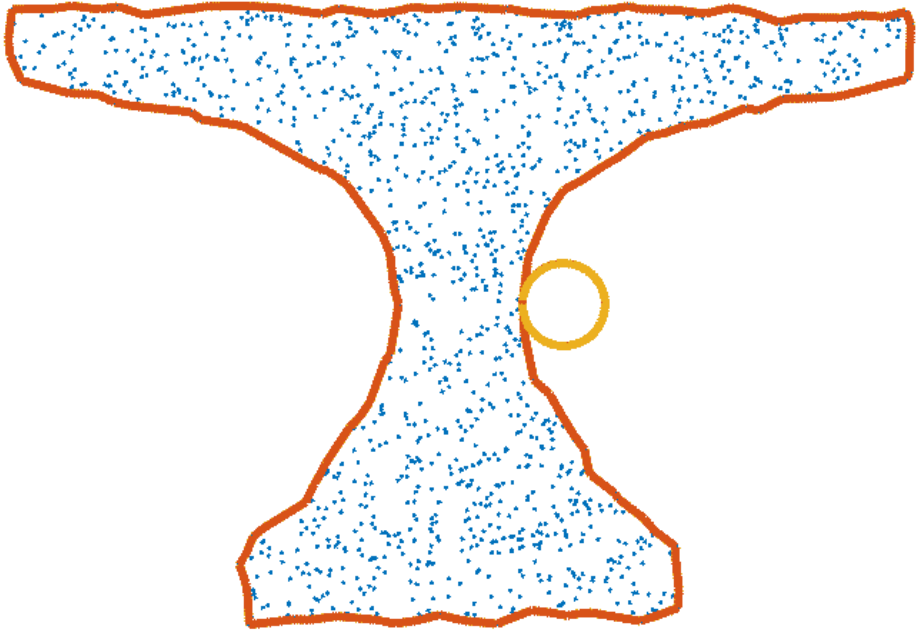

300
250
200
150
100
50
0
50
100
150
200
250
300
350
400
450

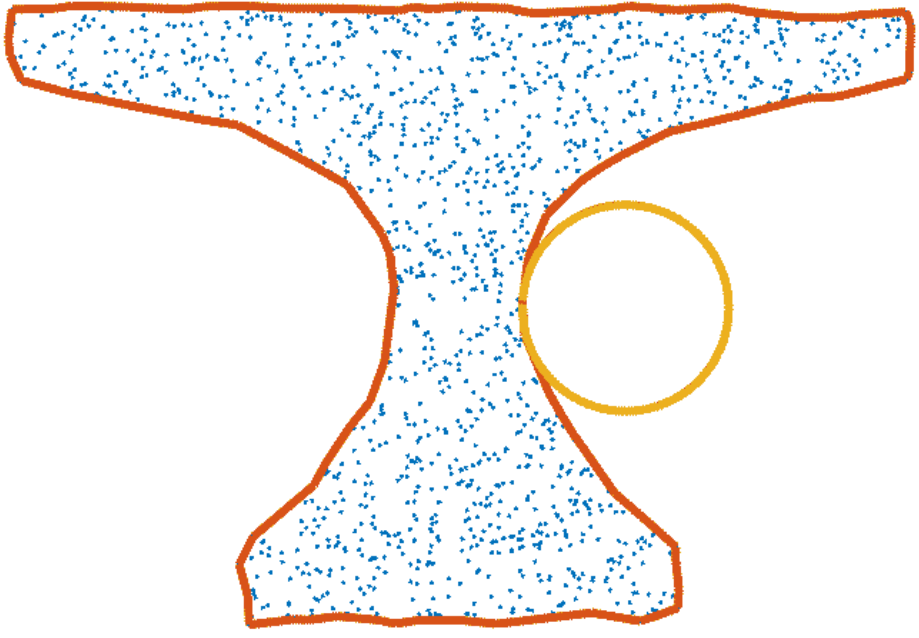


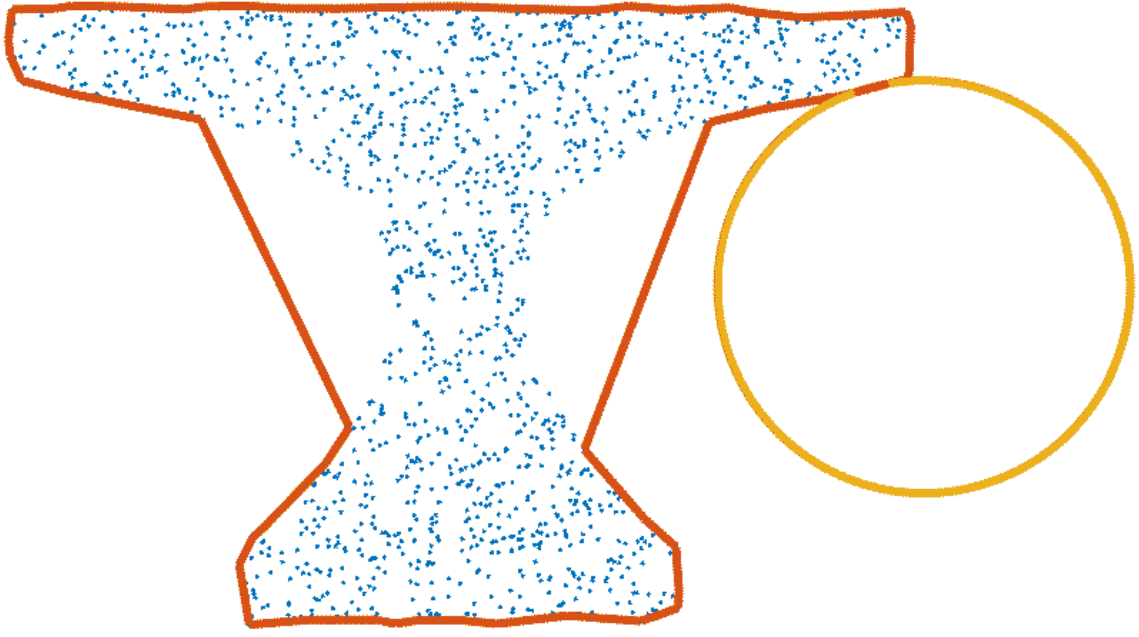


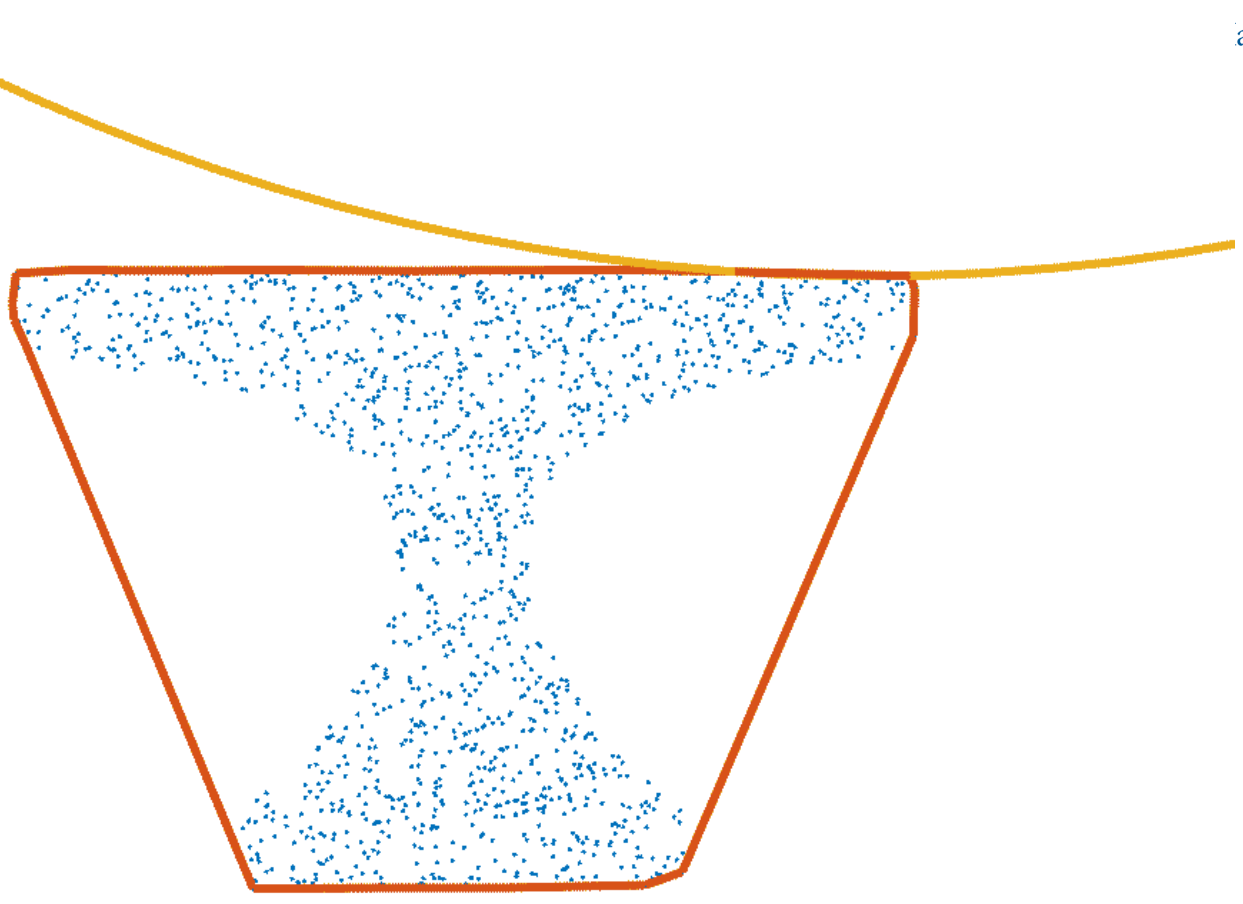






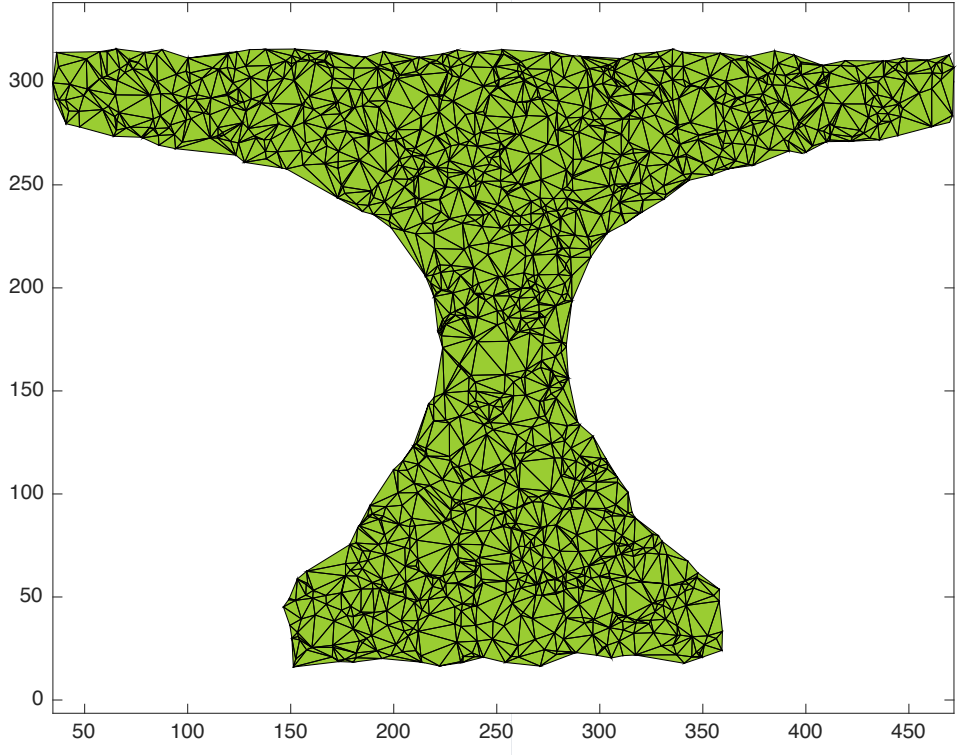






Command Window

```
>> a = alphaShape(x,y,20)

a =

   alphaShape with properties:

              Points: [1411x2 double]
               Alpha: 20
      HoleThreshold: 0
     RegionThreshold: 0

>> plot(a)
```



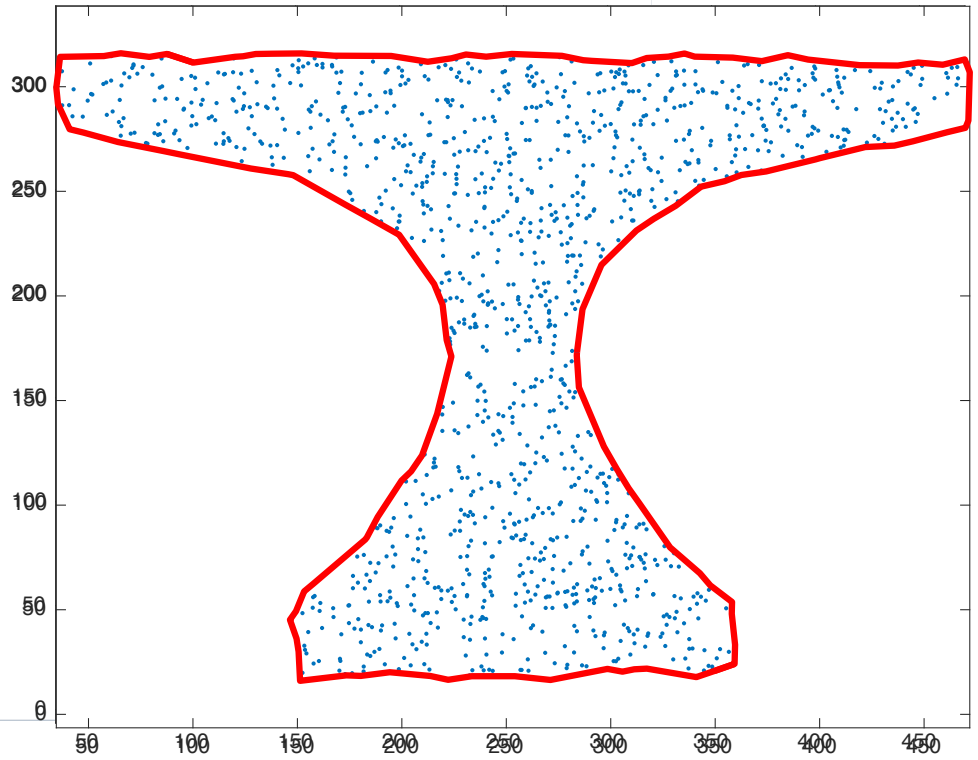
Alpha Shape-Based Geometrical Computations

`alphaSpectrum`	`perimeter`
`criticalAlpha`	`area`
`numRegions`	`surfaceArea`
`inShape`	`volume`
`alphaTriangulation`	`nearestNeighbor`
`boundaryFacets`	

Command Window

```
>> plot(x,y,'.')
>> axis equal
>> k = boundary(x,y);
>> hold on
>> plot(x(k),y(k),'r','LineWidth',3)
>> hold off
```

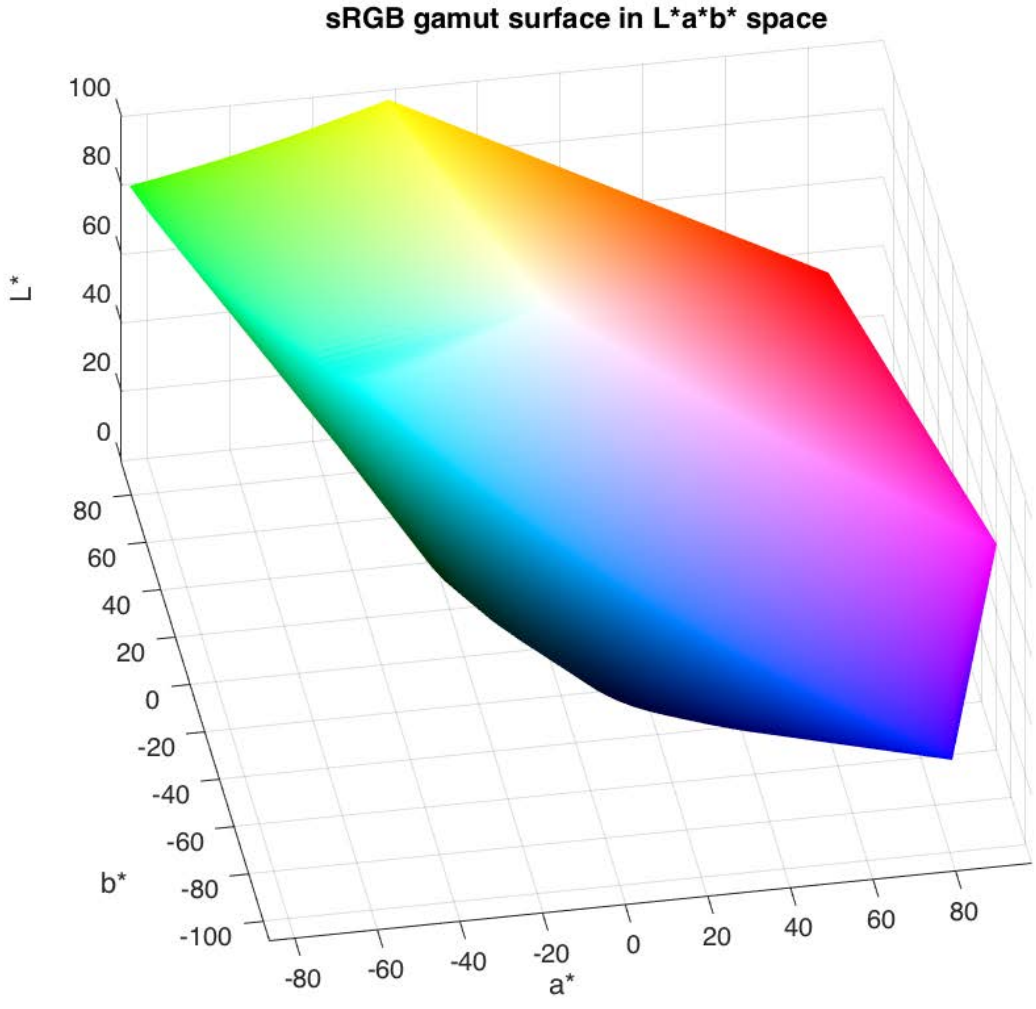
The boundary function is one application of alpha shapes.



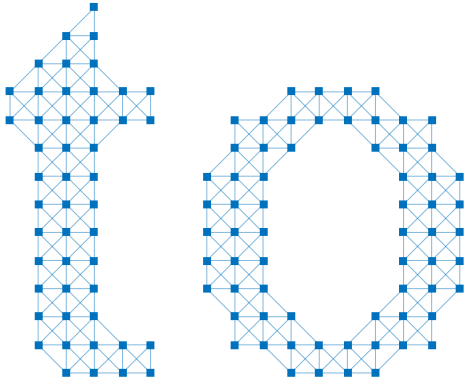
Command Window

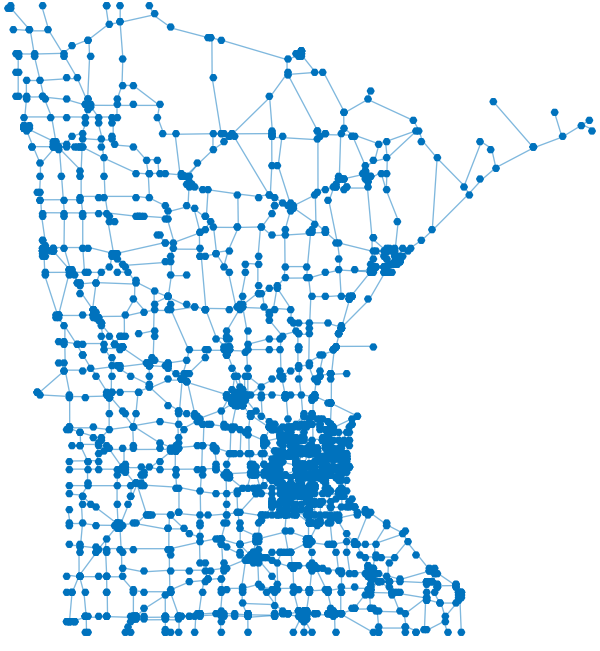
```
>> [r,g,b] = meshgrid(linspace(0,1,50));
>> rgb = [r(:), g(:), b(:)];
>> lab = rgb2lab(rgb);

>> a = lab(:,2);
>> b = lab(:,3);
>> L = lab(:,1);
>> k = boundary(a,b,L);

>> trisurf(k,a,b,L,'FaceColor','interp',...
       'FaceVertexCData',rgb,'EdgeColor','none')
>> xlabel('a*')
>> ylabel('b*')
>> zlabel('L*')
>> axis([-110 110 -110 110 0 100])
>> view(-10,35)
>> axis equal
```

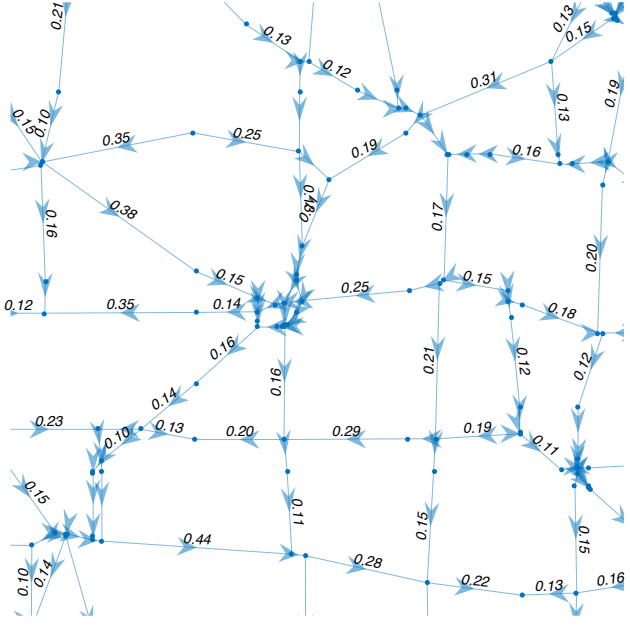
The boundary function works in 3-D, too.


sRGB gamut surface in L*a*b* space
L*
100
80
60
40
20
0
b*
80
60
40
20
0
-20
-40
-60
-80
-100
a*
-80
-60
-40
-20
0
20
40
60
80

Graphs

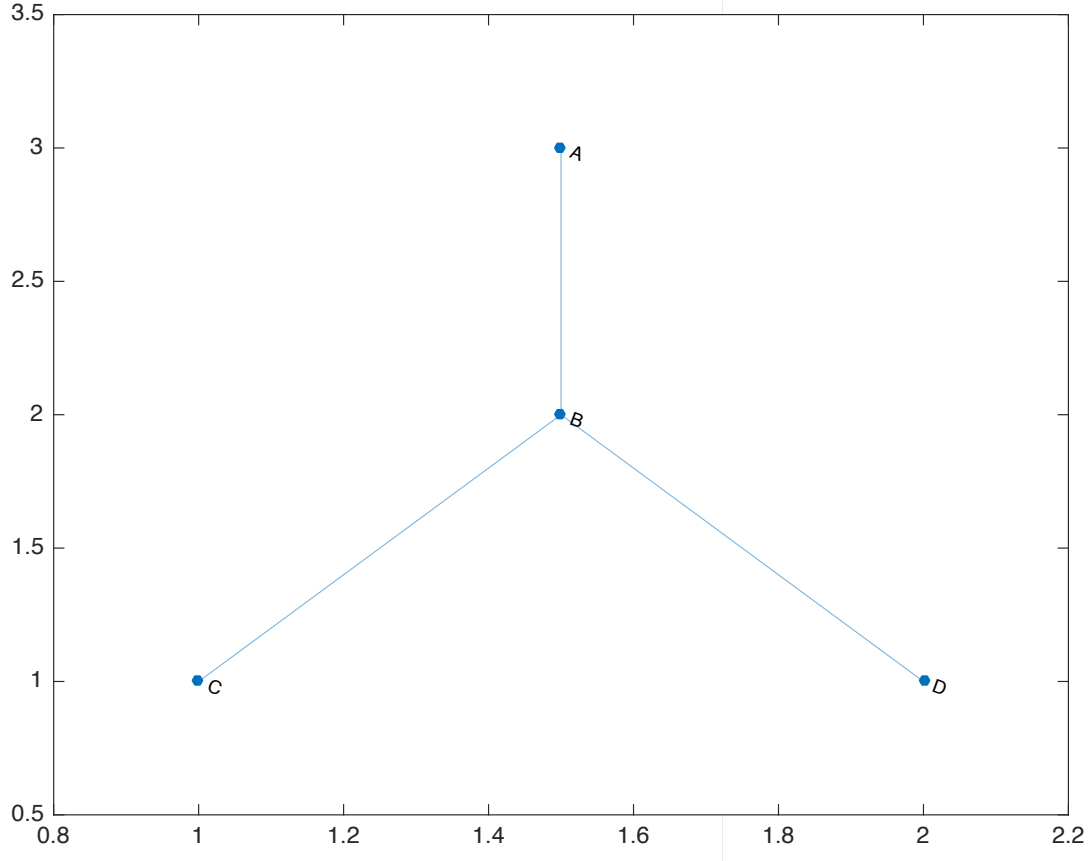





0.21
0.13
0.12
0.31
0.13
0.15
0.19
0.13
0.10
0.35
0.25
0.19
0.16
0.38
0.48
0.17
0.16
0.20
0.15
0.25
0.15
0.12
0.35
0.14
0.18
0.16
0.21
0.12
0.12
0.16
0.14
0.23
0.10
0.13
0.20
0.29
0.19
0.11
0.15
0.11
0.15
0.44
0.15
0.28
0.10
0.14
0.22
0.13
0.16

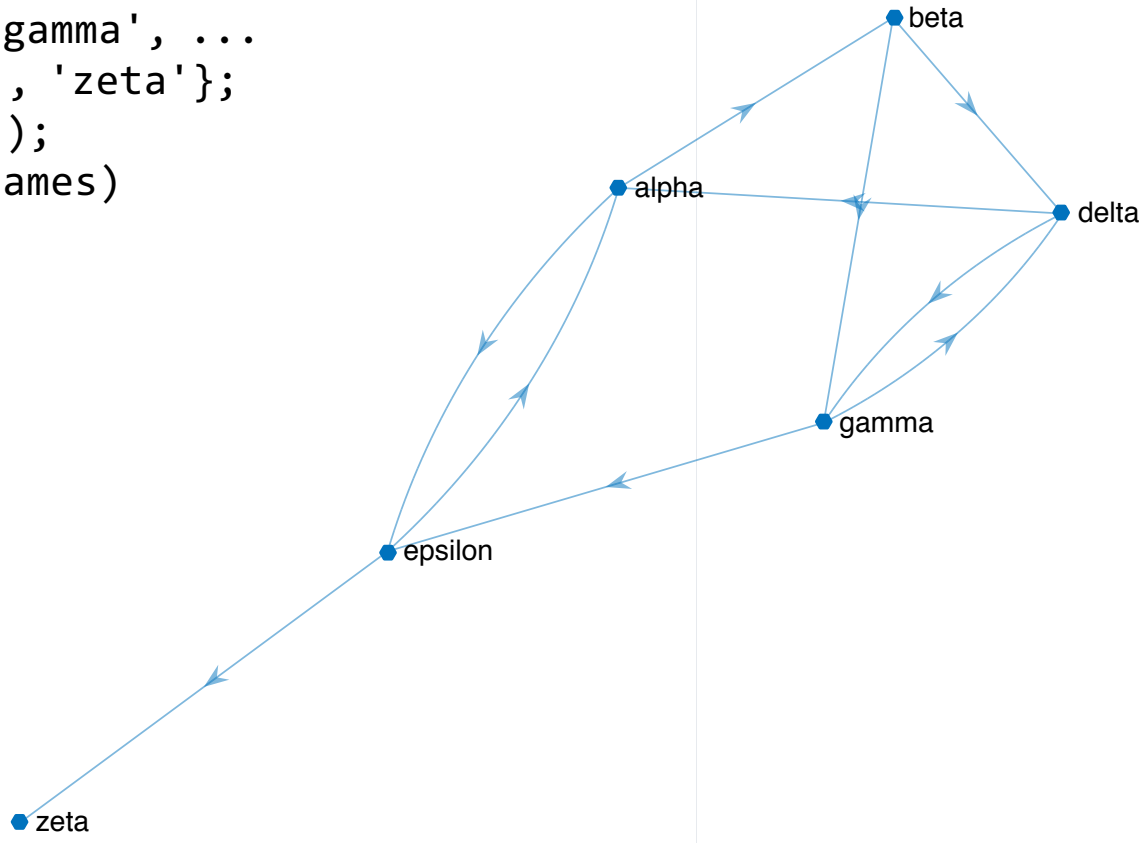
Command Window

```
>> G = graph({'A', 'B', 'B'}, {'B', 'C', 'D'})

G =   graph with properties:

    Edges: [3x1 table]
    Nodes: [4x1 table]

>> plot(G)
```



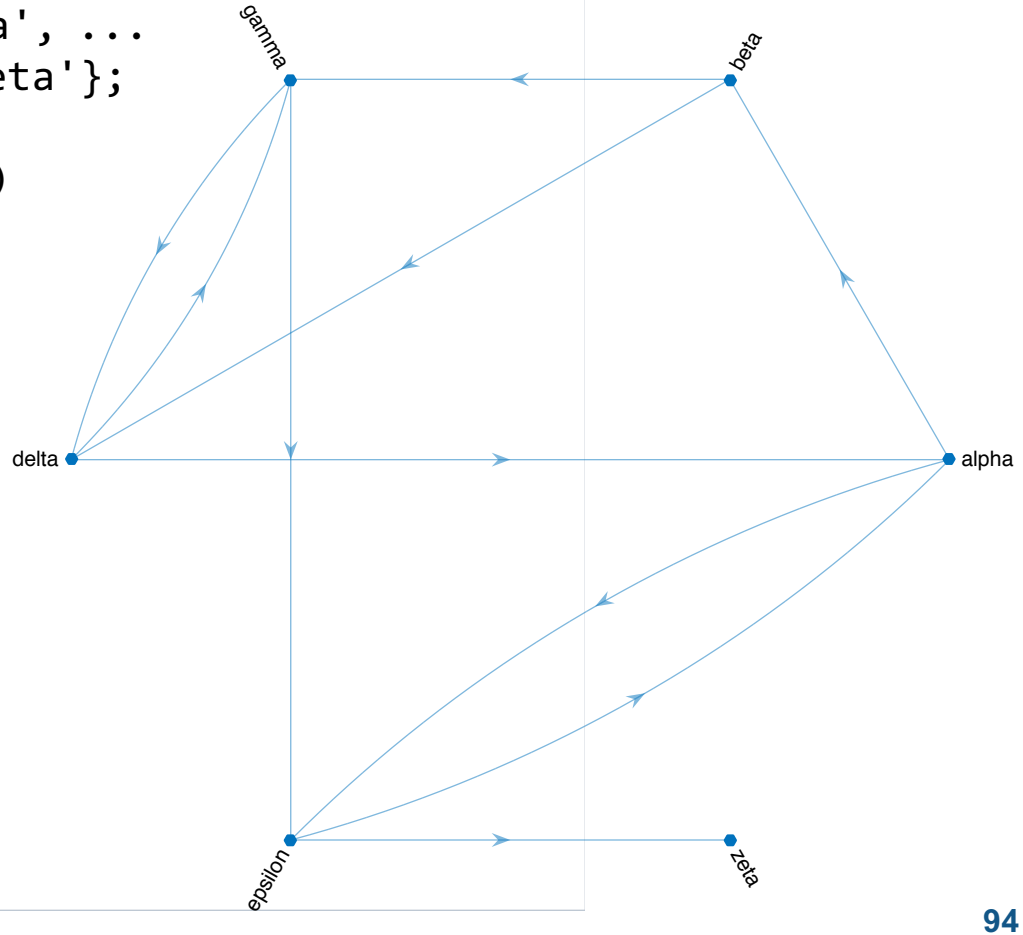
Command Window

```
>> s = [1 1 2 2 3 3 4 4 5 5];
>> t = [2 5 3 4 4 5 3 1 1 6];
>> names = {'alpha', 'beta', 'gamma', ...
            'delta', 'epsilon', 'zeta'};
>> weight = randi(100, size(s));
>> D = digraph(s, t, weight, names)

D =

   digraph with properties:

    Edges: [10x2 table]
    Nodes: [6x1 table]

>> plot(D)
>> axis equal
```



Command Window

```
>> s = [1 1 2 2 3 3 4 4 5 5];
>> t = [2 5 3 4 4 5 3 1 1 6];
>> names = {'alpha', 'beta', 'gamma', ...
            'delta', 'epsilon', 'zeta'};
>> weight = randi(100, size(s));
>> D = digraph(s, t, weight, names)

D =

   digraph with properties:

    Edges: [10x2 table]
    Nodes: [6x1 table]

>> plot(D,'Layout','circle')
>> axis equal
```

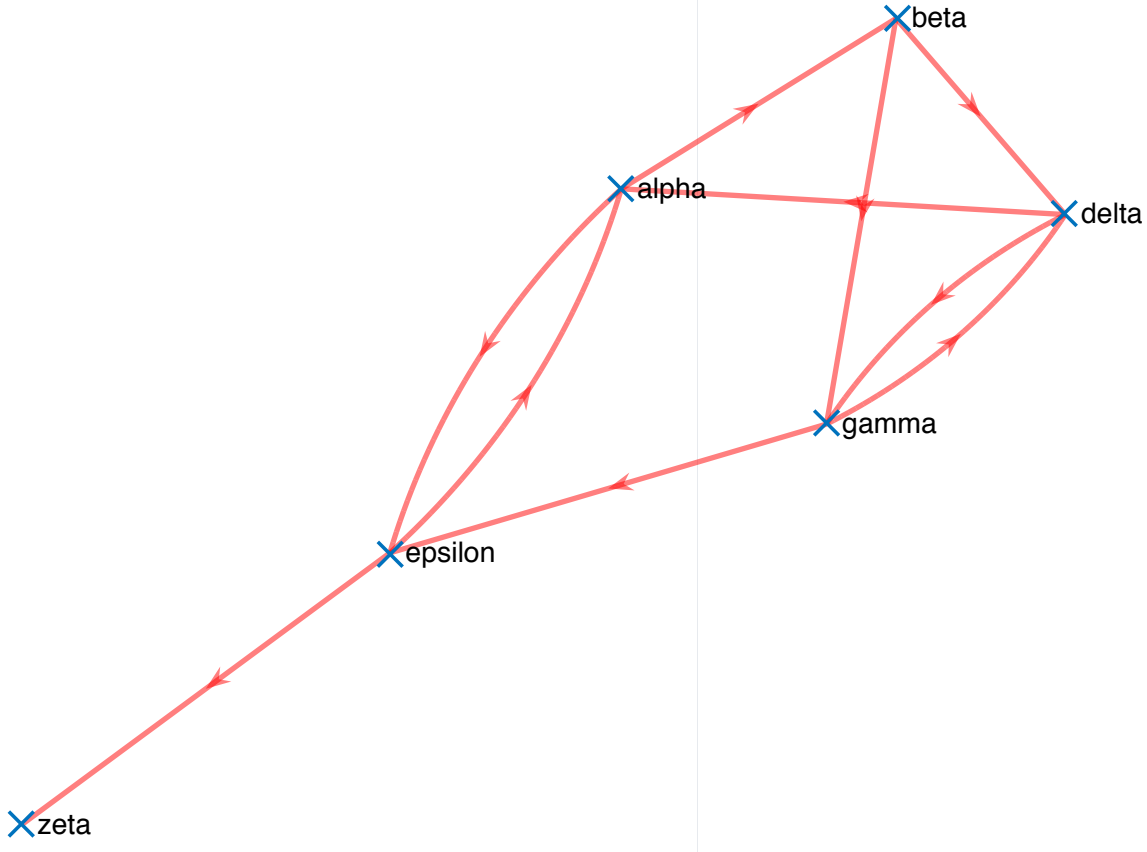


Command Window

```
>> p = plot(D)

p =

   GraphPlot with properties:

     NodeColor: [0 0.4470 0.7410]
    MarkerSize: 4
        Marker: 'o'
     EdgeColor: [0 0.4470 0.7410]
     LineWidth: 0.5000
     LineStyle: '-'
     NodeLabel: {'alpha'  'beta'  'gamma'  'delta'  'epsilon'  'zeta'}
     EdgeLabel: {}
         XData: [-0.0350 0.5891 0.4312 0.9742 -0.5613 -1.3981]
         YData: [0.4486 0.8689 -0.1321 0.3850 -0.4499 -1.1206]

  Show all properties
```

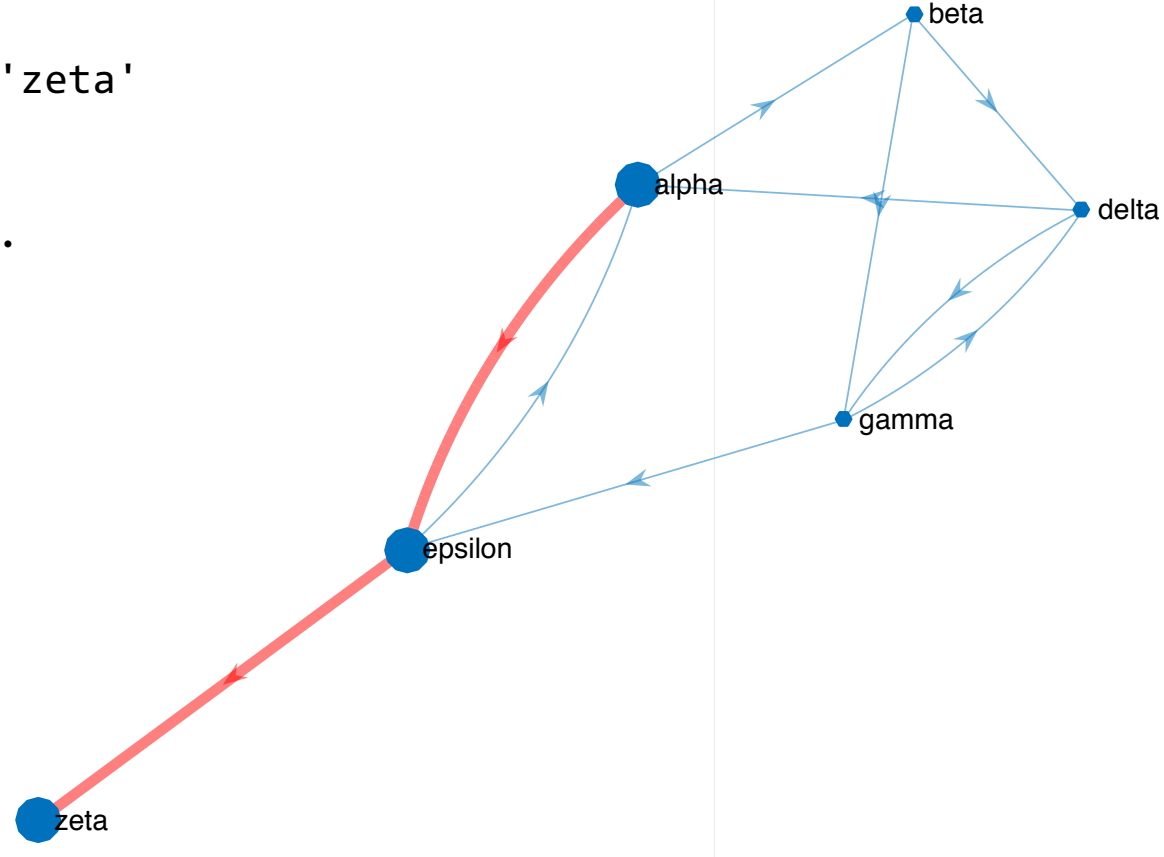
Command Window

```
>> p.MarkerSize = 8;
>> p.Marker = 'x';
>> p.EdgeColor = 'red';
>> p.LineWidth = 1.5;
```



Command Window

```
>> path = shortestpath(D, 'alpha', 'zeta')

shortPath =

     'alpha'    'epsilon'    'zeta'

>> p = plot(D);
>> highlight(p, shortPath, ...
   'MarkerSize',12,...
   'EdgeColor','r',...
   'LineWidth',3)
```



Command Window

```
>> distances(D)

ans =

     0     9    28    81    42   139
   118     0    19    72   108   205
    99   108     0    53    89   186
    47    56    65     0    89   186
    10    19    38    91     0    97
   Inf   Inf   Inf   Inf   Inf     0

>> outdegree(D)

ans =

     2
     2
     2
     2
     2
     0
```

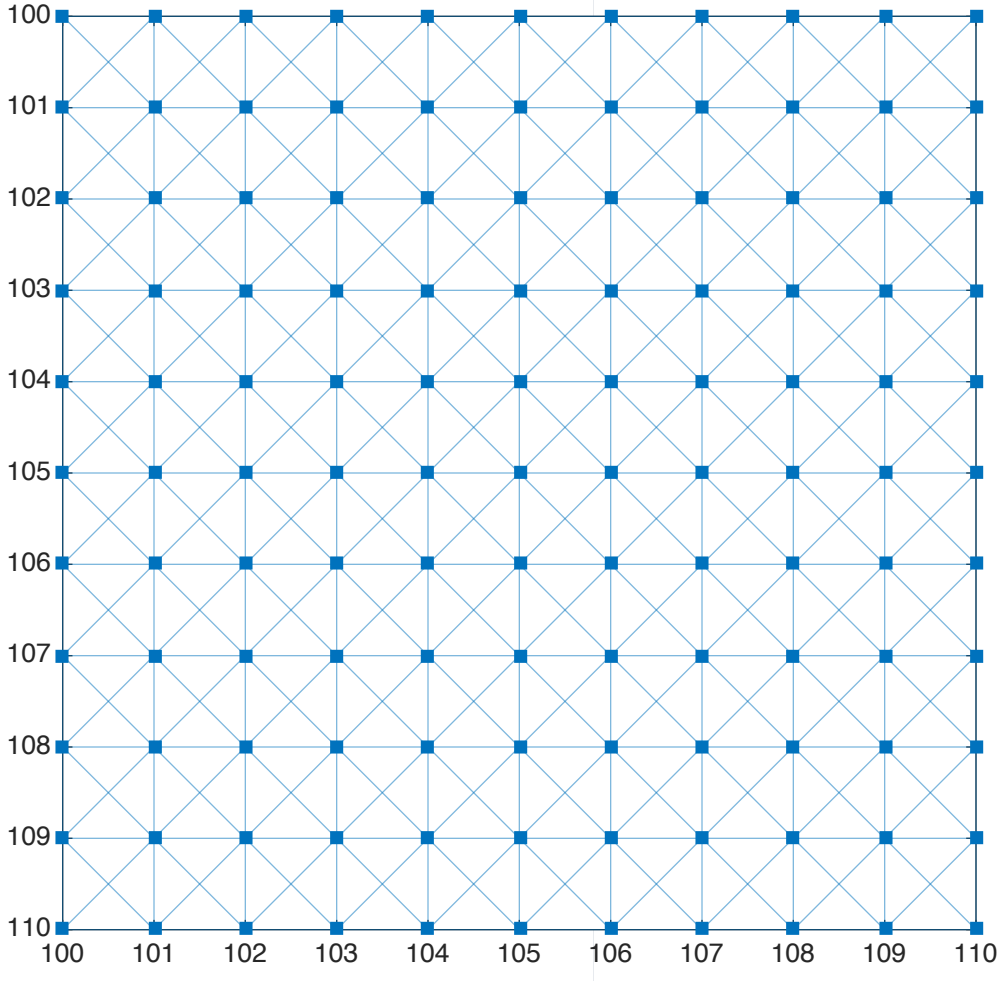
Command Window

```
>> g8 = imageGraph([480 640],8)

g8 =

   graph with properties:

    Edges: [1225442x2 table]
    Nodes: [307200x2 table]

>> g8.Nodes(1:4,:)

ans =

     x    y
     _    _
     1    1
     1    2
     1    3
     1    4
```

imageGraph and plotImageGraph are Steve's prototype functions. Look for them on the File Exchange soon.

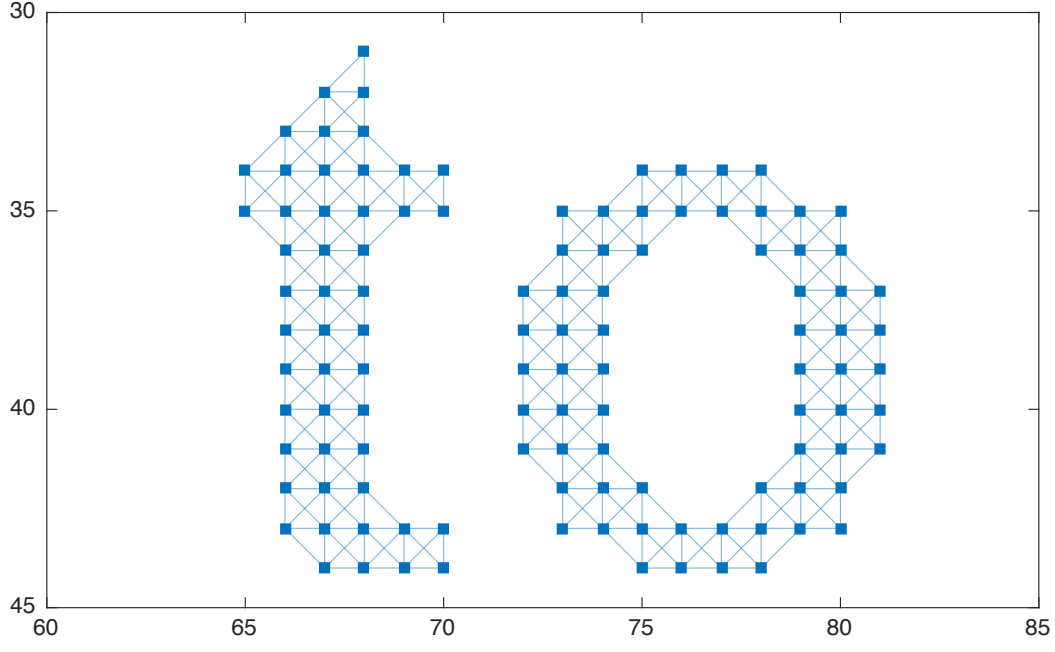
Command Window

```
>> g8.Edges(1:4,:)

ans =

     EndNodes     Weight
    _________    _______
    1       2          1
    1     481          1
    1     482     1.4142
    2       3          1

>> plotImageGraph(g8)
>> axis([100 110 100 110])
```



Command Window

```
>> bw = imread('text.png');
>> g = imageGraph(size(bw),8);
>> foreground_nodes = find(bw);
>> g = subgraph(g,foreground_nodes);
>> plotImageGraph(g)
>> axis([60 85 30 45])
```

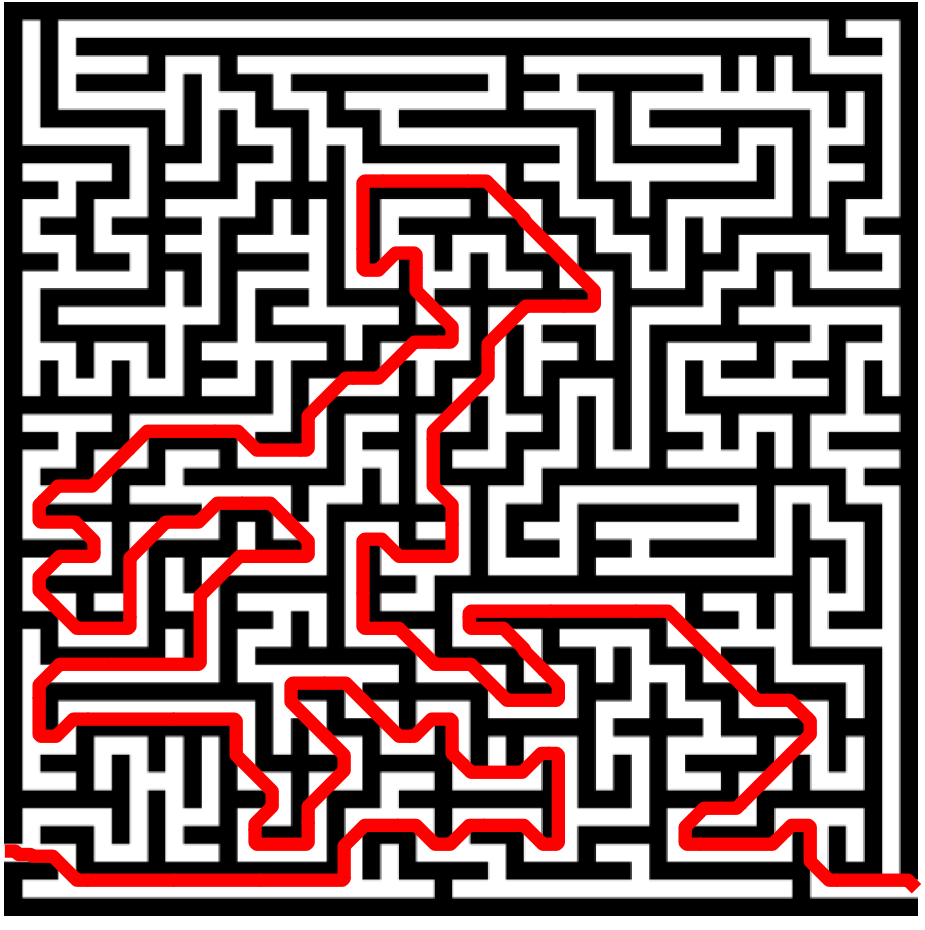


A binary image mask can be used to compute a subgraph.

The term watershed refers to a ridge that ...

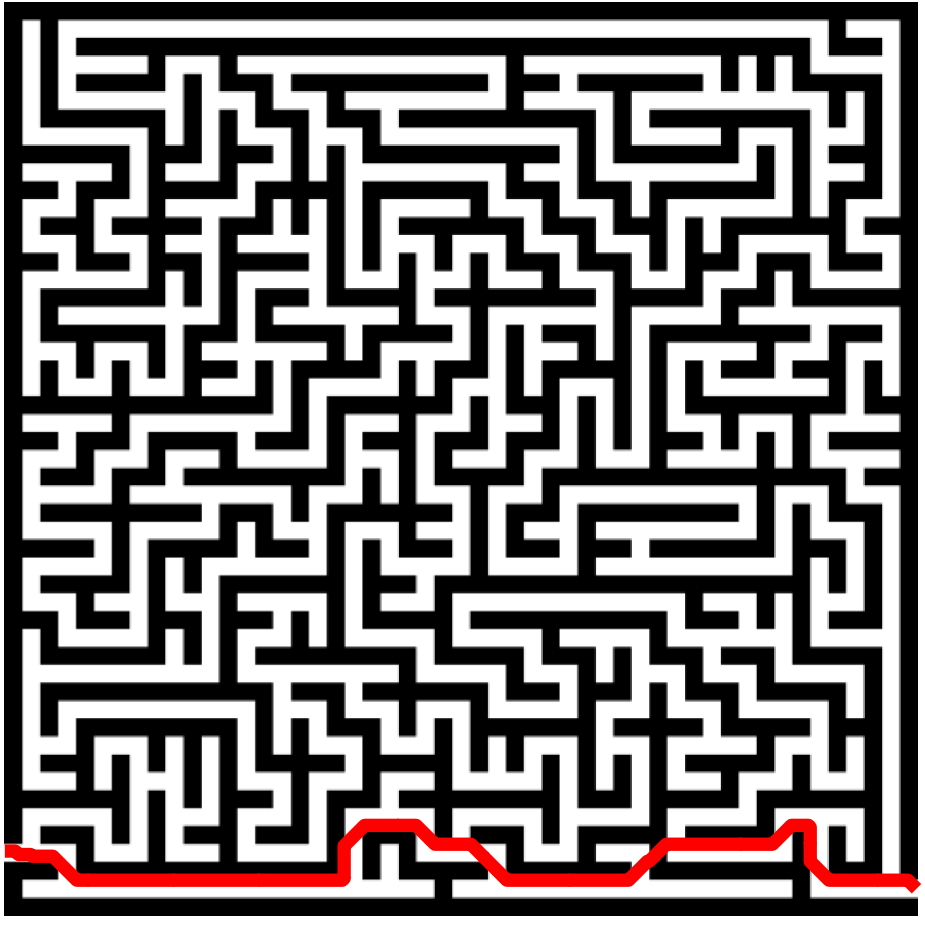
... divides areas drained by different river systems.

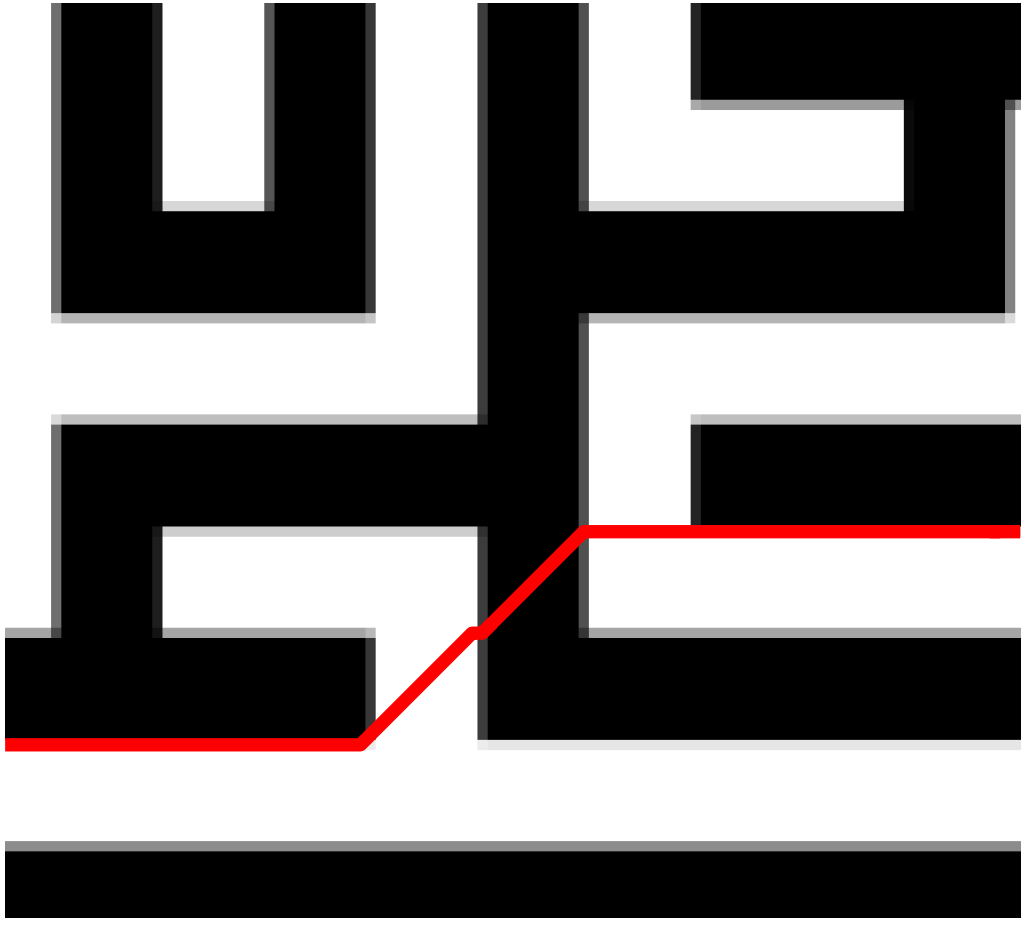
A Cheating Maze Solver



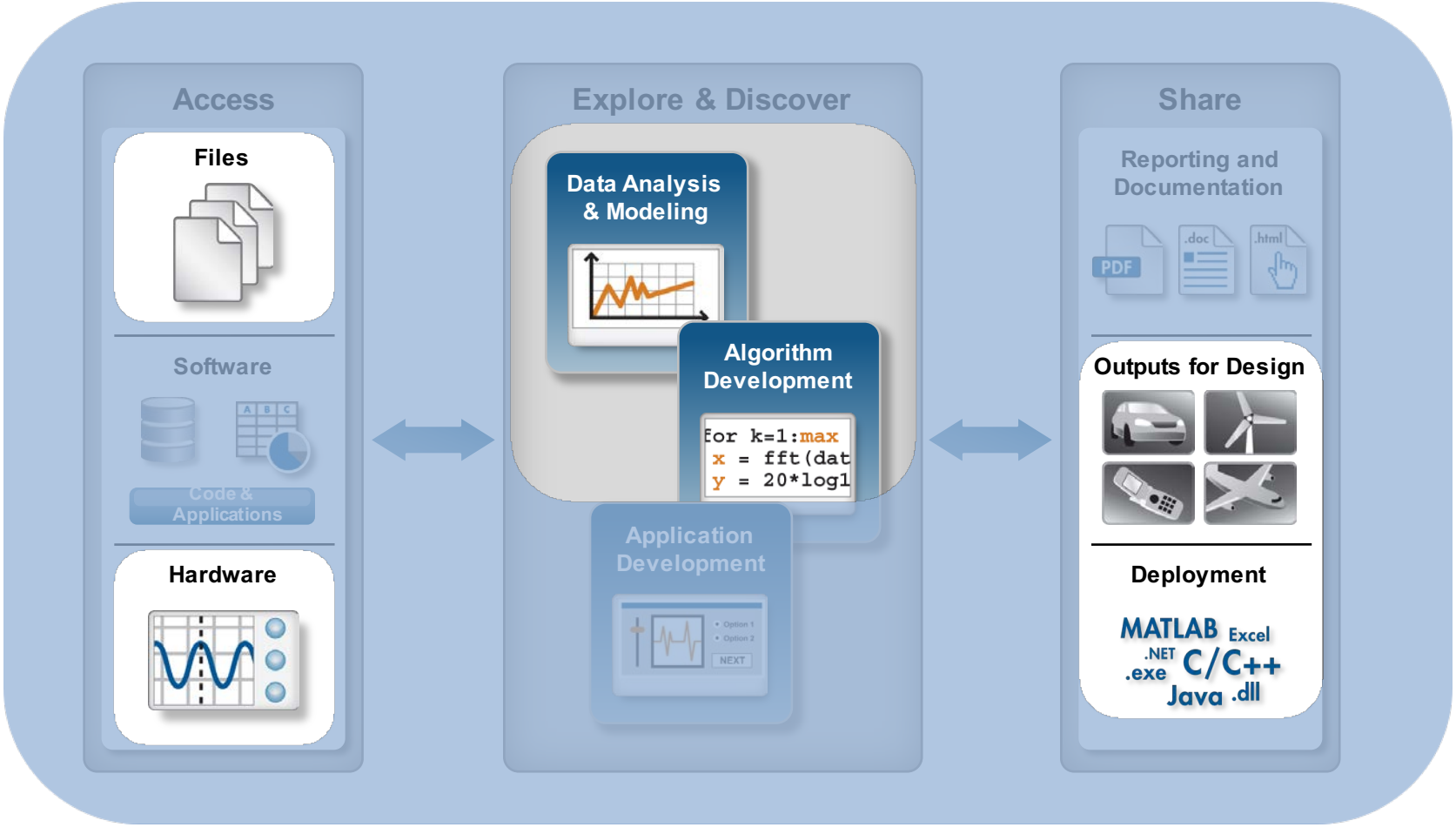
Command Window

```
>> maze = imread('maze-51x51.png');
>> g = imageGraph(size(maze),8);
>> tunneling_penalty = 100;

>> touches_wall = any(~maze(g.Edges.EndNodes),2);
>> g.Edges.Weight(touches_wall) = tunneling_penalty;

>> p = shortestpath(g,start_node,finish_node);

>> imshow(maze)
>> hold on
>> plot(g.Nodes.x(p),g.Nodes.y(p),'r','LineWidth',5)
>> hold off
```





MATLAB Performance

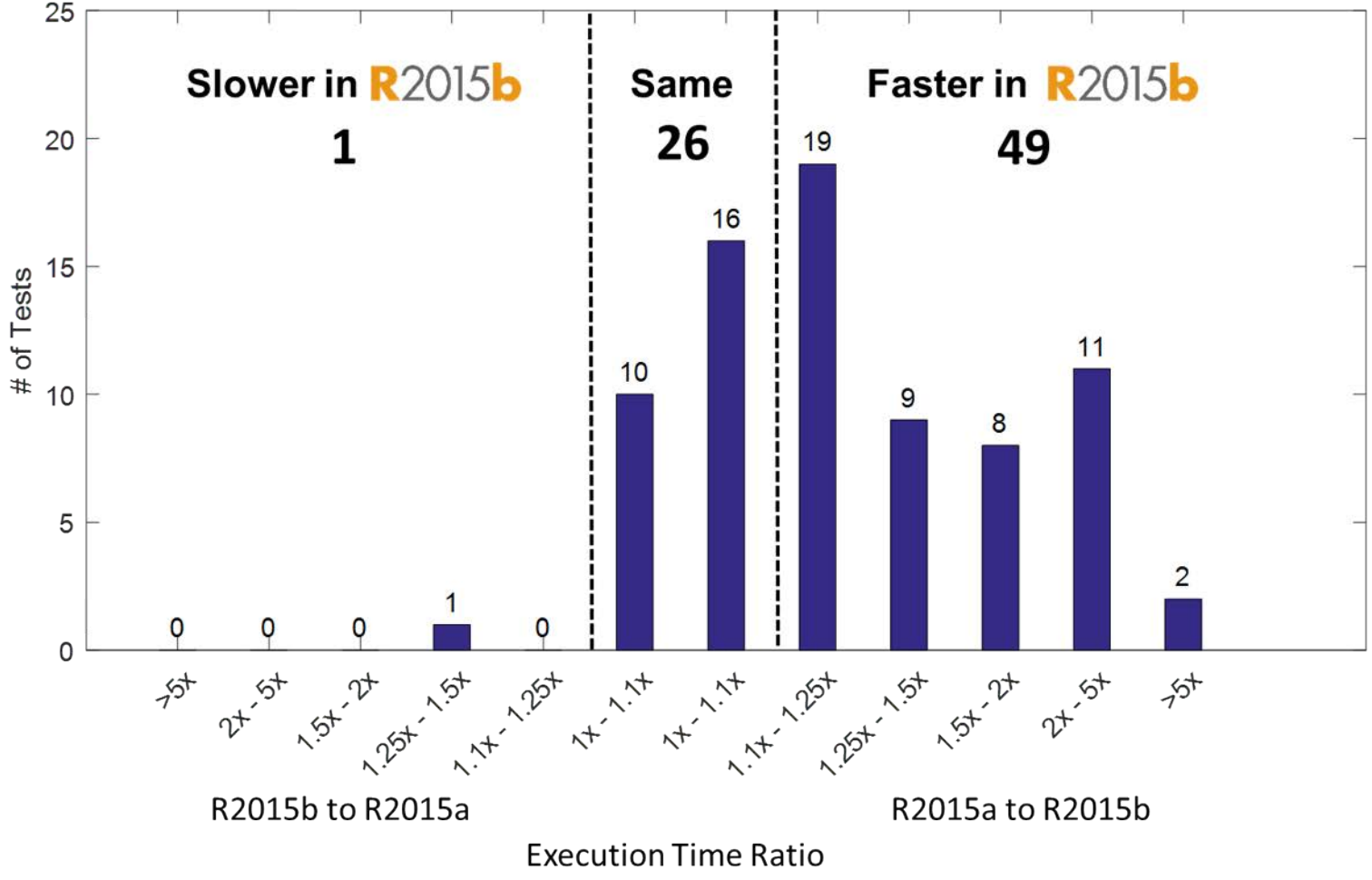


MATLAB Execution Engine

R2015b

- The old MATLAB interpreter is no more.
- All MATLAB code is compiled to machine code just-in-time (JIT)

User Application Benchmarks



Notable Speed-ups

- Function calls
- Some element-wise operations
- Some object-oriented code

Command Window

```
>> f = @() fibonacci(25);
>> timeit(f)

ans =

    0.7781
```

R2015a 0.7781

```
function x = fibonacci(n)
    if n < 2
        x = n;
        return
    else
        x = fibonacci(n-1) + fibonacci(n-2);
    end
end
```

Command Window

```
>> f = @() fibonacci(25);
>> timeit(f)

ans =

    0.0113
```

R2015a 0.7781

R2015b 0.0113

Ratio: 68.9

On Steve's MacBook Pro

```
function x = fibonacci(n)
    if n < 2
        x = n;
        return
    else
        x = fibonacci(n-1) + fibonacci(n-2);
    end
end
```

Faster Functions

- Dozens of MATLAB functions
- Dozens of Image Processing Toolbox functions
- More are optimized during every six-month release cycle via:
  - Algorithm changes
  - Multithreading
  - Advanced processor instructions
  - Improved memory cache efficiency

Some Faster MATLAB Functions

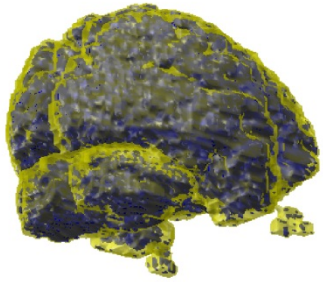
conv2	integer conversion functions	integer arithmetic functions
fft	sortrows	convn
histc	svd	trig functions
decimal trig functions	chol	textscan
qr	interpolation functions	rand
Bessel functions	airy	psi
permute	flip	circshift
repmat	linear algebra functions (AMD)	filter
readtable	sort	typecast
Excel I/O functions	table I/O functions	VideoReader

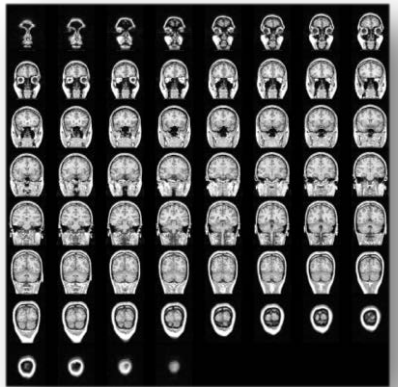
Some Faster Image Processing Toolbox Functions

edge	imdilate	imerode
imfilter	imresize	iradon
medfilt2	bwmorph	imabsdiff
imclose	imopen	dicomread
std2	blockproc (in parallel clusters)	imhist
rgb2gray	imlincomb	adapthisteq
histeq	imrotate	intlut
DICOM functions	Gaussian filtering	

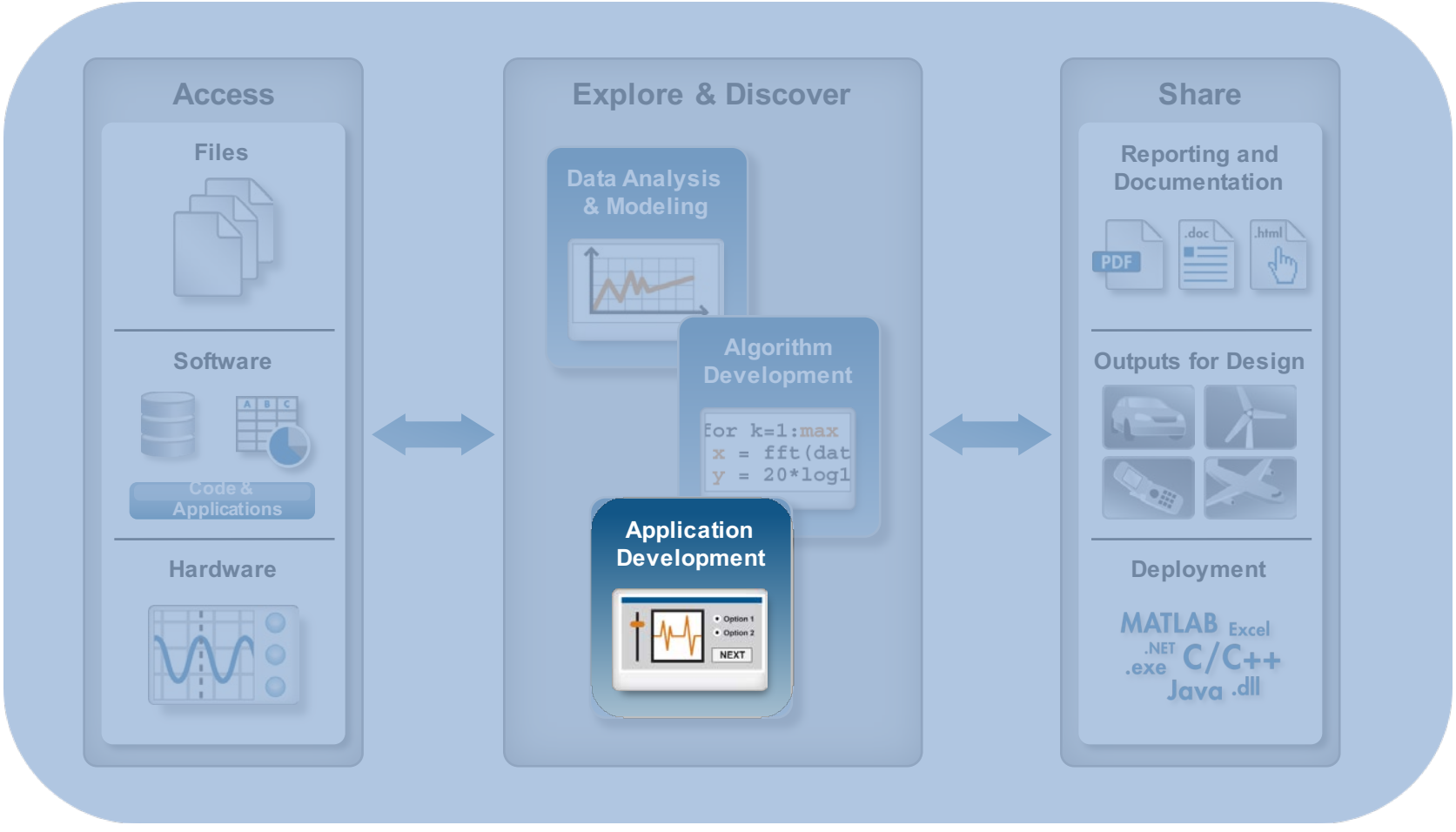
Computing on the GPU

- Requires Parallel Computing Toolbox
- And a supported GPU
  - *CUDA-enabled NVIDIA GPU with compute capability 2.0 or higher*
- 311 MATLAB functions supported
- 48 Image Processing Toolbox functions supported





Software Development

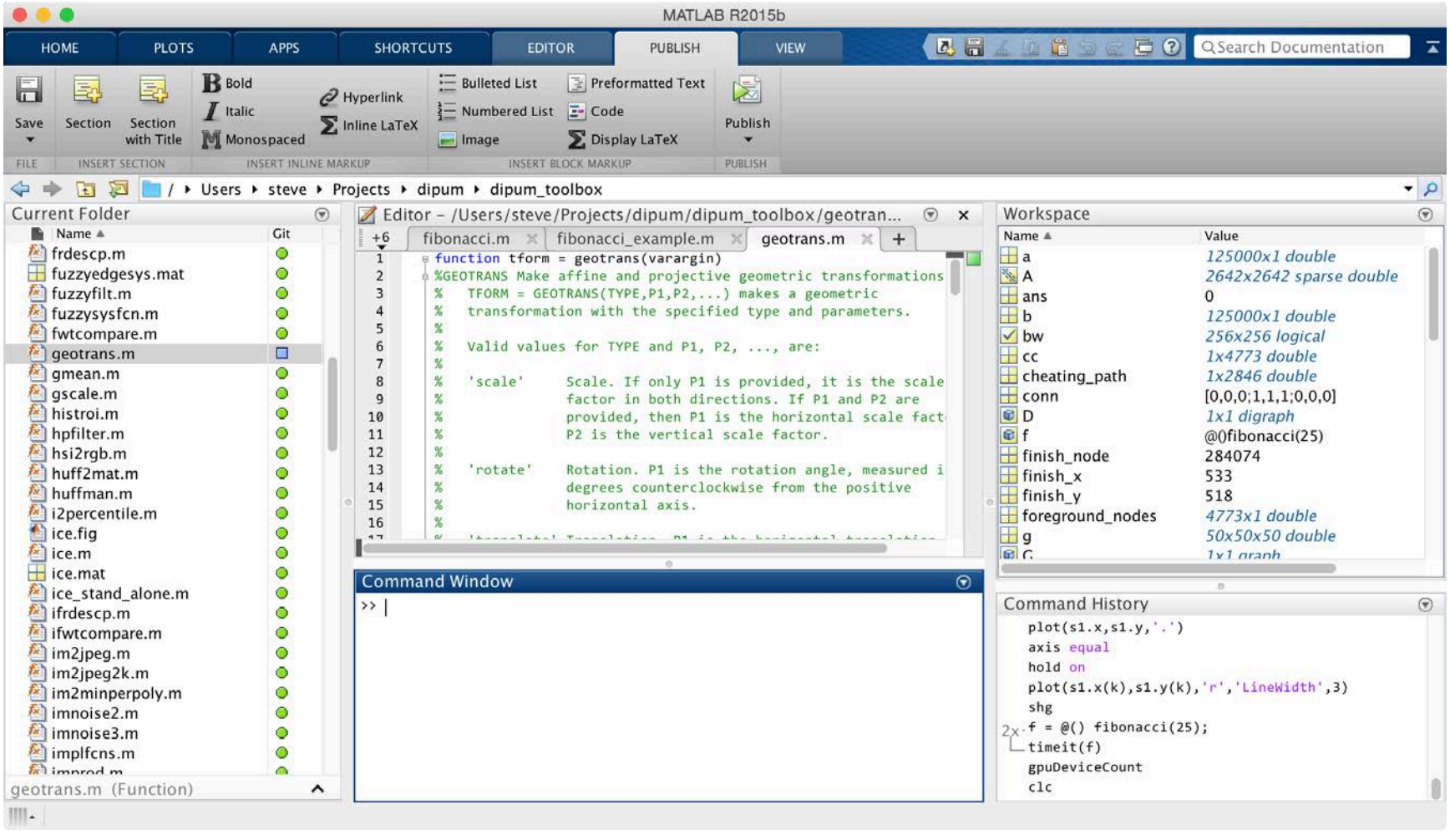


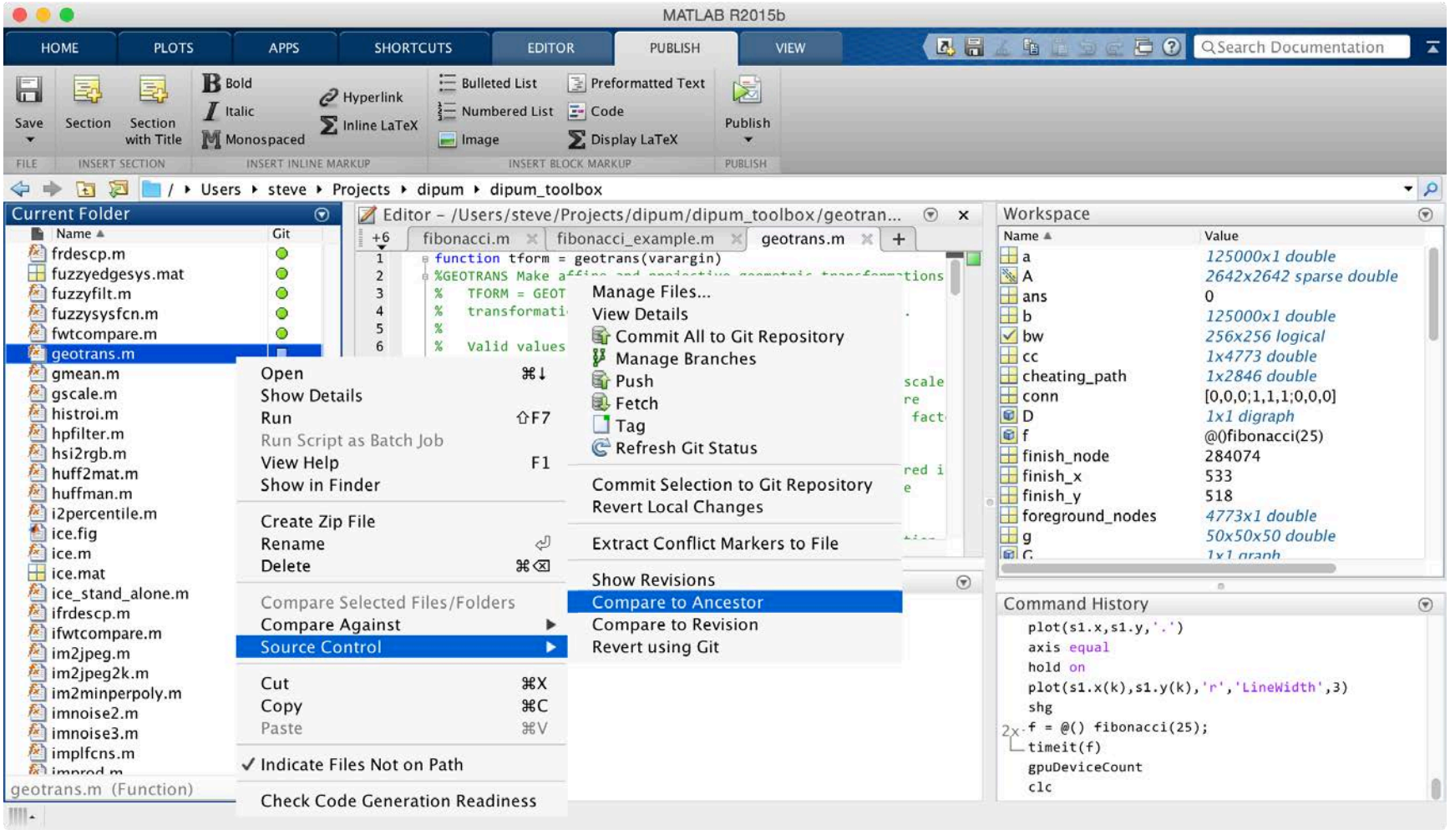
Do You Distribute Code Collections to Others?

Then you have fallen into the world of the software developer.

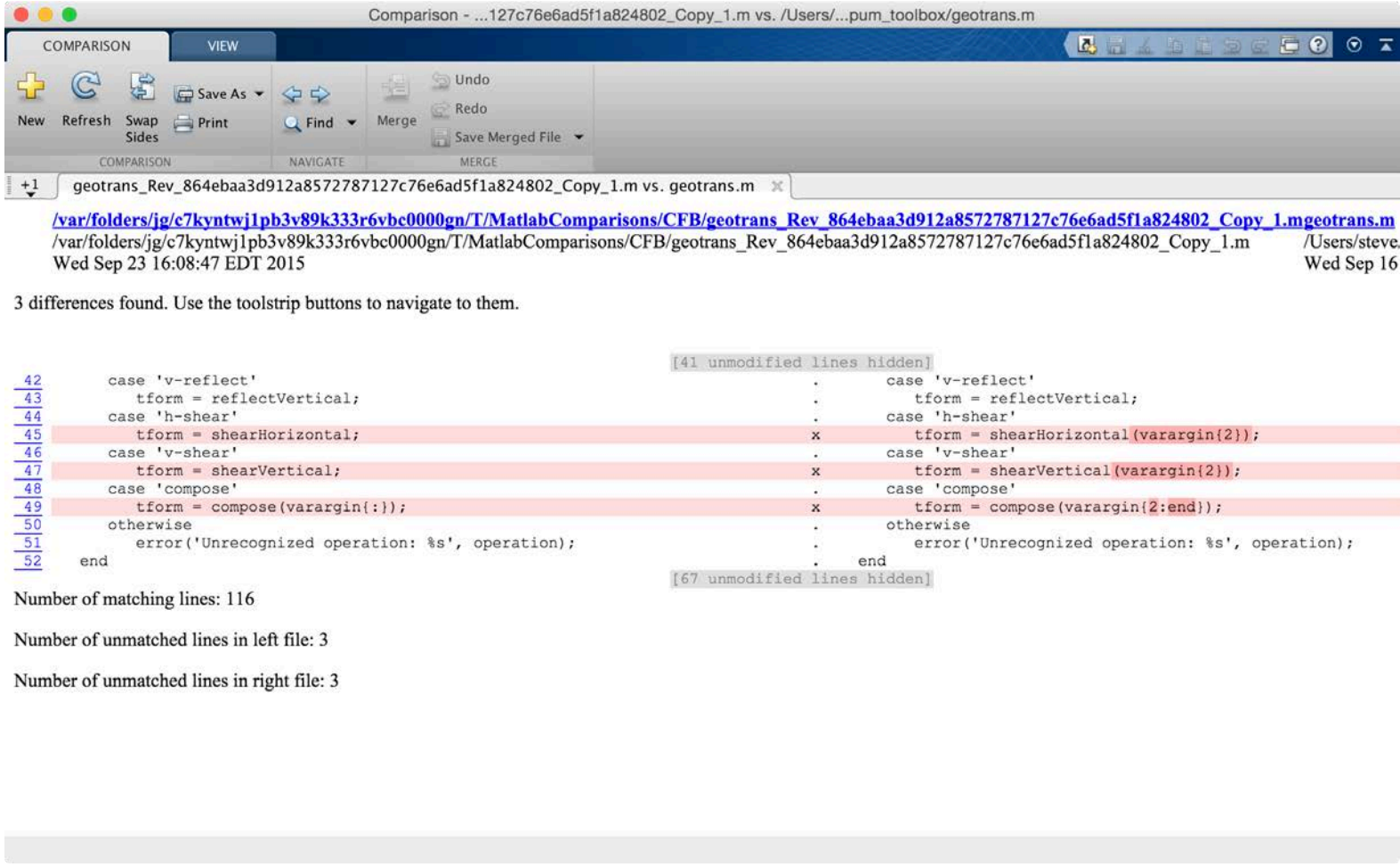
There are some things you need:

- Version control
- File and folder comparison tools
- Unit testing tools
- Packaging

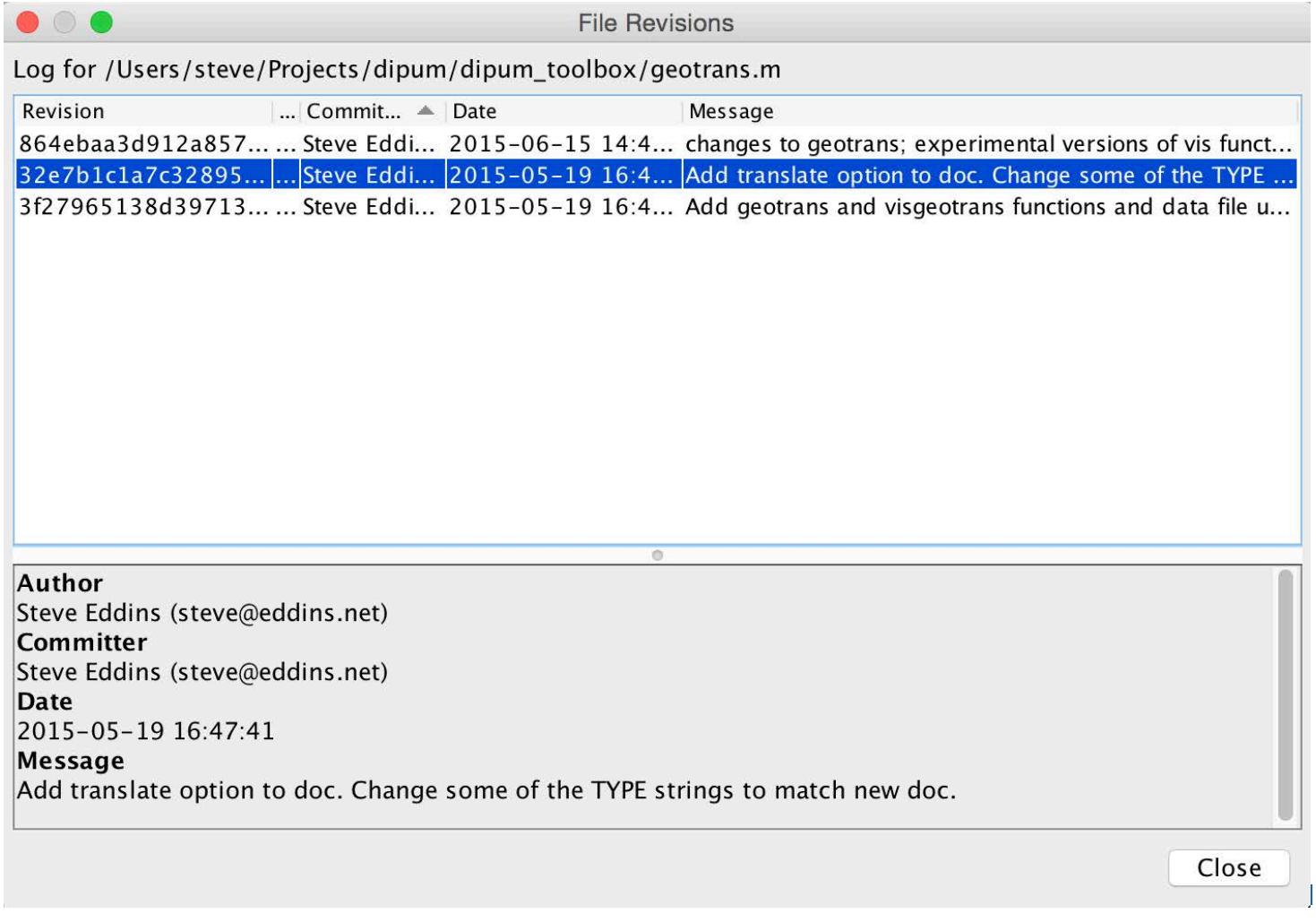

MATLAB R2015b
HOME
PLOTS
APPS
SHORTCUTS
EDITOR
PUBLISH
VIEW
Search Documentation
Save
Section
Section with Title
Bold
Italic
Monospaced
Hyperlink
Inline LaTeX
Bulleted List
Numbered List
Image
Preformatted Text
Code
Display LaTeX
Publish
FILE
INSERT SECTION
INSERT INLINE MARKUP
INSERT BLOCK MARKUP
PUBLISH
/ ▸ Users ▸ steve ▸ Projects ▸ dipum ▸ dipum_toolbox
Current Folder
Name
Git
frdescp.m
fuzzyedgesys.mat
fuzzyfilt.m
fuzzysysfcn.m
fwtcompare.m
geotrans.m
gmean.m
gscale.m
histroi.m
hpfilter.m
hsi2rgb.m
huff2mat.m
huffman.m
i2percentile.m
ice.fig
ice.m
ice.mat
ice_stand_alone.m
ifrdescp.m
ifwtcompare.m
im2jpeg.m
im2jpeg2k.m
im2minperpoly.m
imnoise2.m
imnoise3.m
implfcns.m
geotrans.m (Function)
Editor - /Users/steve/Projects/dipum/dipum_toolbox/geotran...
fibonacci.m
fibonacci_example.m
geotrans.m
function tform = geotrans(varargin)
%GEOTRANS Make affine and projective geometric transformations
% TFORM = GEOTRANS(TYPE,P1,P2,...) makes a geometric
% transformation with the specified type and parameters.
%
% Valid values for TYPE and P1, P2, ..., are:
%
% 'scale' Scale. If only P1 is provided, it is the scale
% factor in both directions. If P1 and P2 are
% provided, then P1 is the horizontal scale fact
% P2 is the vertical scale factor.
%
% 'rotate' Rotation. P1 is the rotation angle, measured i
% degrees counterclockwise from the positive
% horizontal axis.
%
Command Window
>>
Workspace
Name
Value
a 125000x1 double
A 2642x2642 sparse double
ans 0
b 125000x1 double
bw 256x256 logical
cc 1x4773 double
cheating_path 1x2846 double
conn [0,0,0;1,1,1;0,0,0]
D 1x1 digraph
f @()fibonacci(25)
finish_node 284074
finish_x 533
finish_y 518
foreground_nodes 4773x1 double
g 50x50x50 double
Command History
plot(s1.x,s1.y,'.')
axis equal
hold on
plot(s1.x(k),s1.y(k),'r','LineWidth',3)
shg
2x f = @() fibonacci(25);
timeit(f)
gpuDeviceCount
clc


MATLAB R2015b
HOME
PLOTS
APPS
SHORTCUTS
EDITOR
PUBLISH
VIEW
Search Documentation
Save
Section
Section with Title
Bold
Italic
Monospaced
Hyperlink
Inline LaTeX
Bulleted List
Numbered List
Image
Preformatted Text
Code
Display LaTeX
Publish
FILE
INSERT SECTION
INSERT INLINE MARKUP
INSERT BLOCK MARKUP
PUBLISH
/ ▸ Users ▸ steve ▸ Projects ▸ dipum ▸ dipum_toolbox
Current Folder
Name
Git
frdescp.m
fuzzyedgesys.mat
fuzzyfilt.m
fuzzysysfcn.m
fwtcompare.m
geotrans.m
gmean.m
gscale.m
histroi.m
hpfilter.m
hsi2rgb.m
huff2mat.m
huffman.m
i2percentile.m
ice.fig
ice.m
ice.mat
ice_stand_alone.m
ifrdescp.m
ifwtcompare.m
im2jpeg.m
im2jpeg2k.m
im2minperpoly.m
imnoise2.m
imnoise3.m
implfcns.m
geotrans.m (Function)
Editor - /Users/steve/Projects/dipum/dipum_toolbox/geotran...
fibonacci.m
fibonacci_example.m
geotrans.m
function tform = geotrans(varargin)
%GEOTRANS Make a
% TFORM = GEOT
% transformati
% Valid values
Open ⌘↓
Show Details
Run ⇧F7
Run Script as Batch Job
View Help F1
Show in Finder
Create Zip File
Rename
Delete ⌘⌫
Compare Selected Files/Folders
Compare Against
Source Control
Cut ⌘X
Copy ⌘C
Paste ⌘V
Indicate Files Not on Path
Check Code Generation Readiness
Manage Files...
View Details
Commit All to Git Repository
Manage Branches
Push
Fetch
Tag
Refresh Git Status
Commit Selection to Git Repository
Revert Local Changes
Extract Conflict Markers to File
Show Revisions
Compare to Ancestor
Compare to Revision
Revert using Git
Workspace
Name
Value
a 125000x1 double
A 2642x2642 sparse double
ans 0
b 125000x1 double
bw 256x256 logical
cc 1x4773 double
cheating_path 1x2846 double
conn [0,0,0;1,1,1;0,0,0]
D 1x1 digraph
f @()fibonacci(25)
finish_node 284074
finish_x 533
finish_y 518
foreground_nodes 4773x1 double
g 50x50x50 double
Command History
plot(s1.x,s1.y,'.')
axis equal
hold on
plot(s1.x(k),s1.y(k),'r','LineWidth',3)
shg
f = @() fibonacci(25);
timeit(f)
gpuDeviceCount
clc

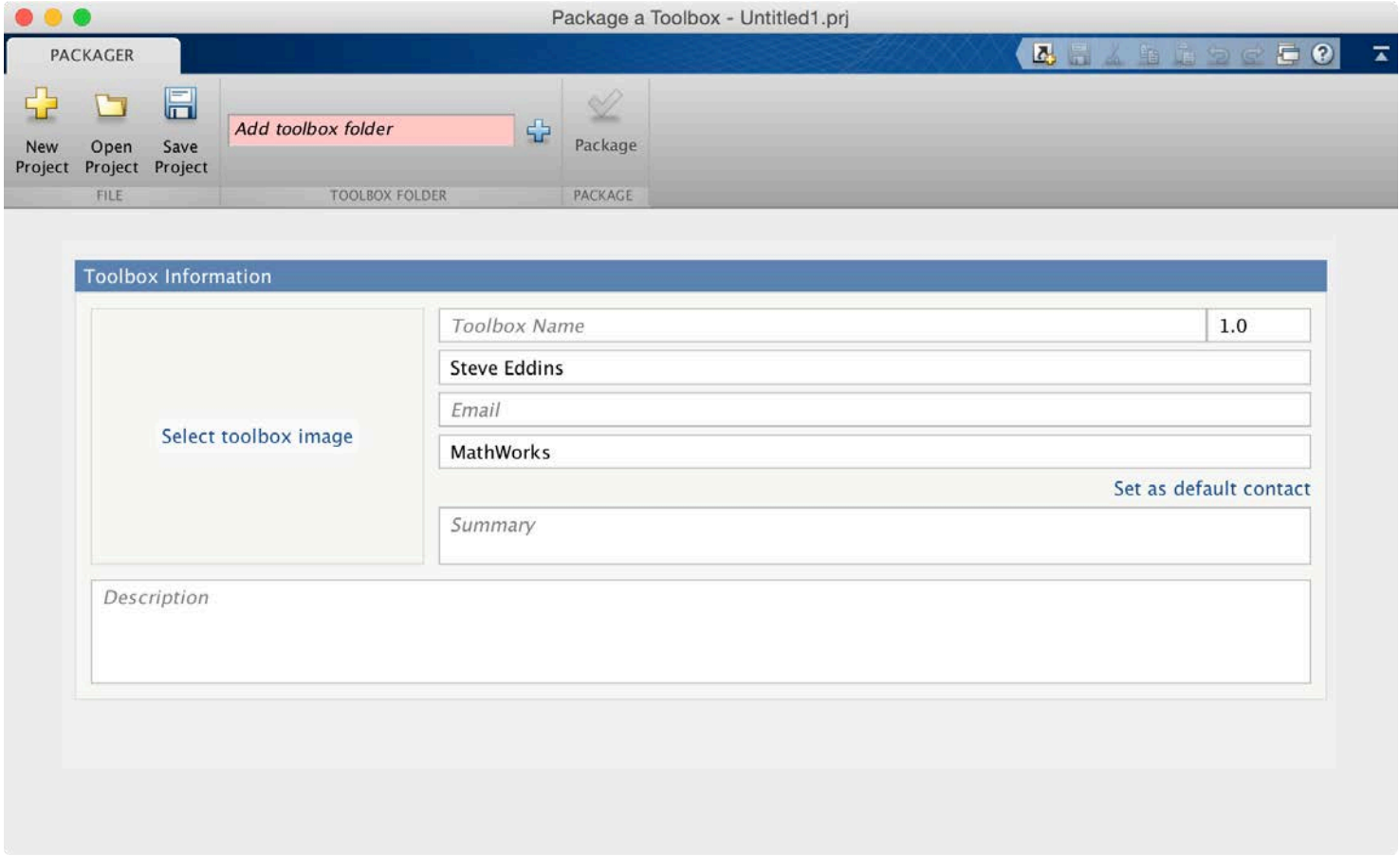
File and Folder Comparison Tool



File Revision History



Toolbox Packager (produces self-installing file)



Unit Testing

- Verify the expected behavior of the smallest components of your code
- Automate the verification process that it can be easily (and frequently!) repeated
- Demo: writing test cases by writing functions

MATLAB Unit Testing Framework

- Supports a variety of test writing styles:
  - Scripts
  - Functions
  - Custom TestCase objects
- Large collection of validation checks
- Extensively customizable
  - test fixtures
  - parameterized tests
  - diagnostics
  - plugins

Image Processing Algorithms and Apps

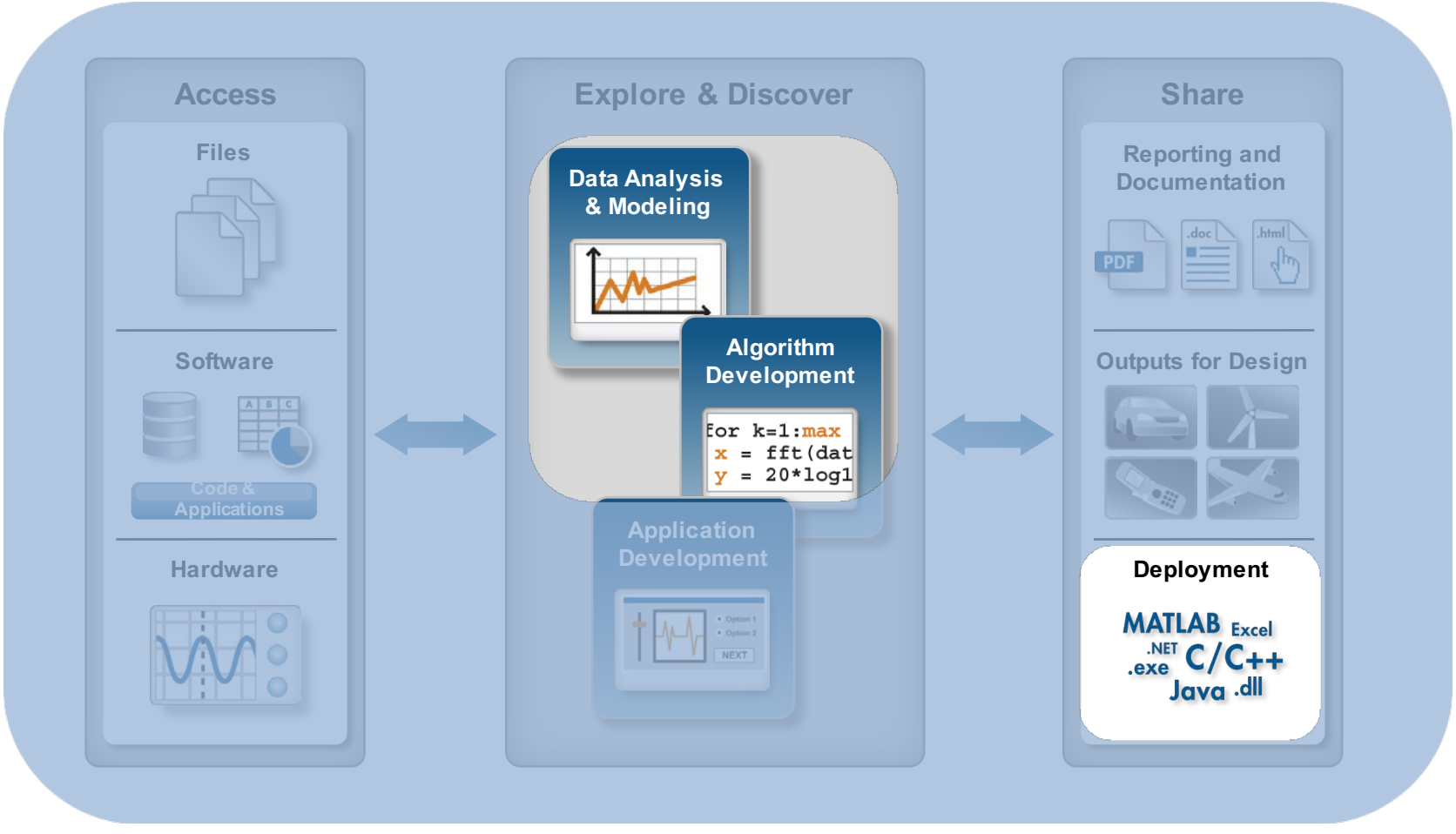


Image Registration

- Intensity-based
  - Unimodal
  - Multimodal
- Phase correlation
- Translation-only
- Nonrigid
- Visualization
  - imshowpair
  - imfuse

Image Analysis and Segmentation

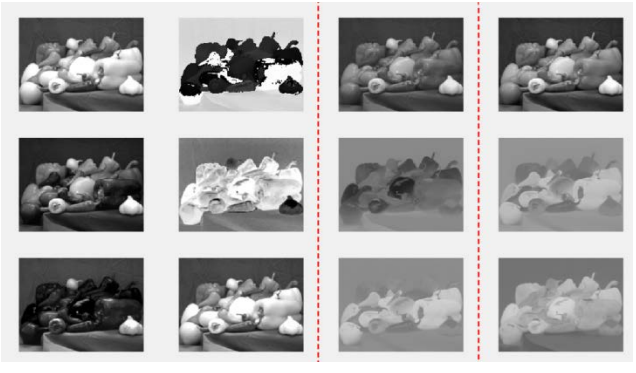
- Circle finding
- Active contours
- Fast marching segmentation
- Fast geodesic interactive segmentation

General Image Processing Operators

- Binary image convex hulls
- Geodesic distance transform
- Gradient magnitude, phase, x- and y-components
- Multilevel thresholding
- Quality metrics
- Gabor filtering
- Box filtering
- Gray-level connected components

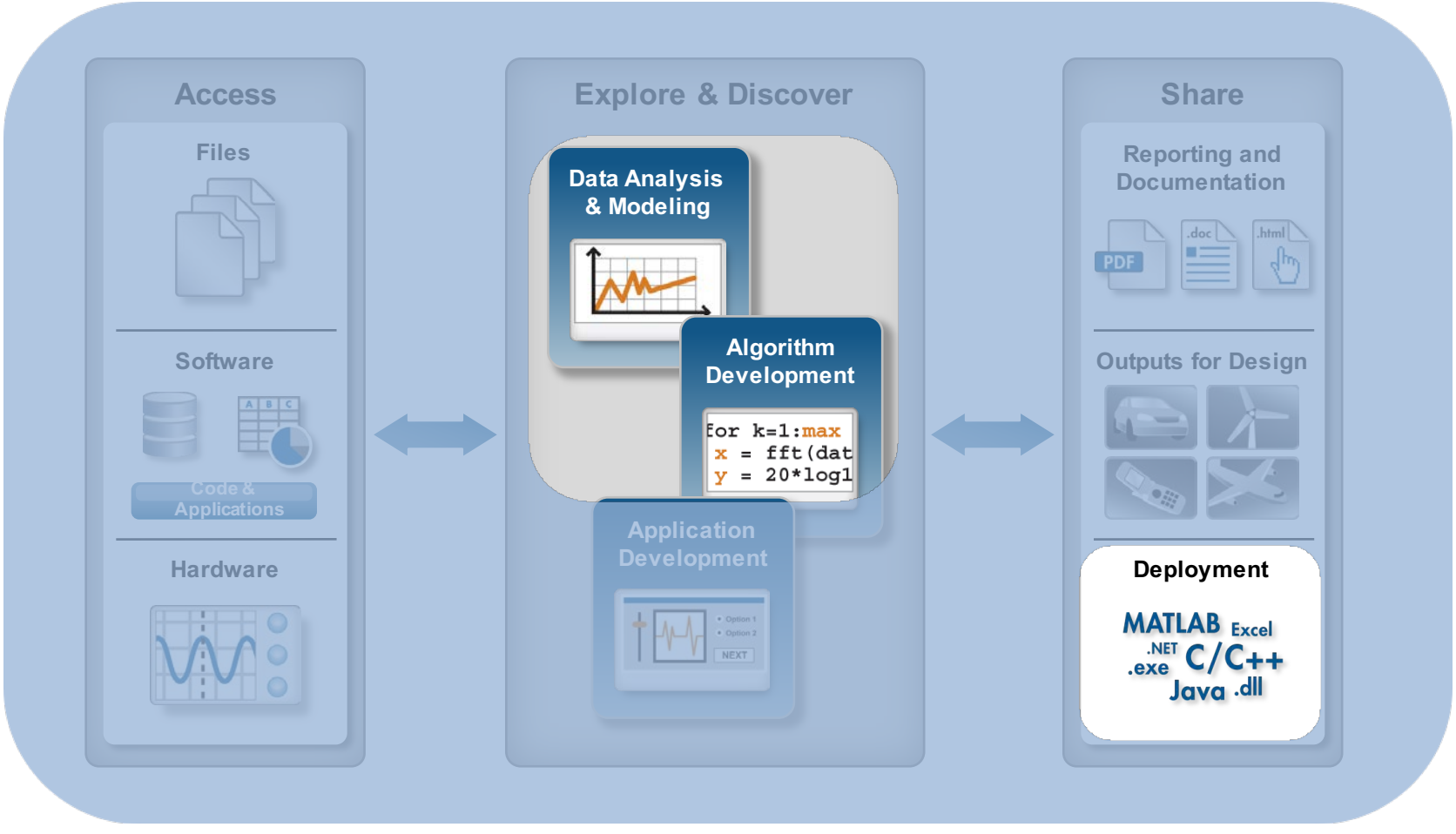
Image Processing Apps

- Color Image Thresholder
- Image Segmentation
- Image Region Analyzer
- Batch Image Processor



Color Thresholder Demo

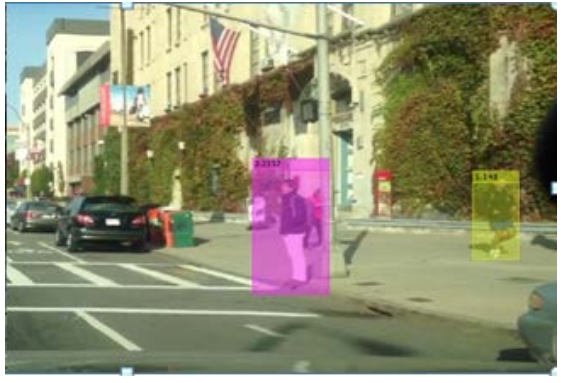
Computer Vision Algorithms and Apps



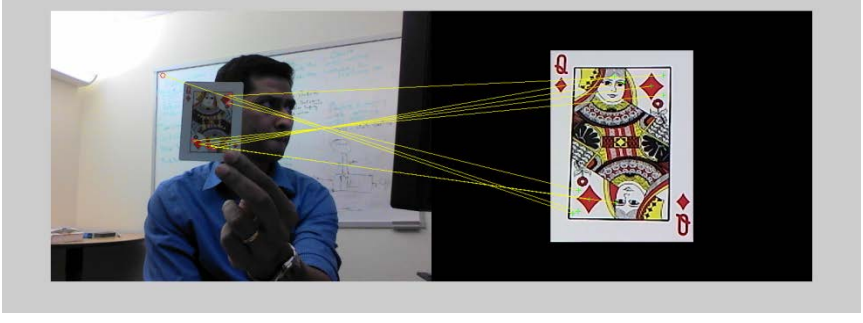
Feature Detection and Extraction

- SURF
- MSER
- BRISK
- Harris Corner
- FREAK
- HOG
- LBP
- Feature matching
- Visualizations for features, feature matching

Object Detection and Recognition



- Cascade detector (Viola-Jones)
  - Face detector
  - Upper body detector
- Cascade detector training
- HOG-based pedestrian detector
- Bag-of-features (BoF)
- Image category classification
- Image search and retrieval
- Training image labeler app
- Nonmaximal bounding box suppression

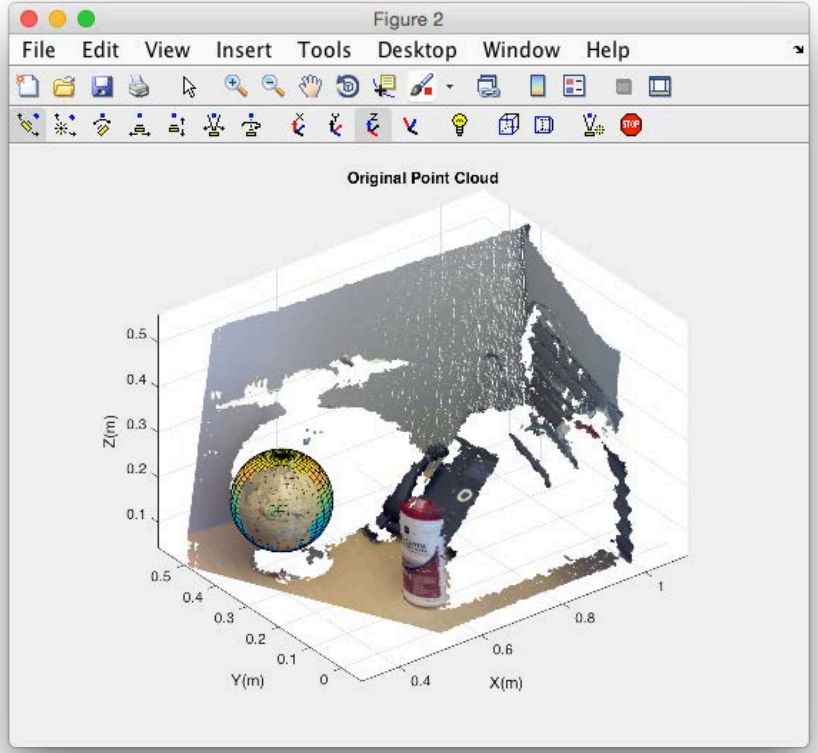


Object Tracking

- Multiobject tracking framework
  - Kalman filter
  - Hungarian algorithm for detection with tracks
- CAMShift tracker
- KLT-based point tracker
- Tracked object annotation

3-D Point Cloud Processing

- K-D tree-based neighborhood search
- Point cloud file I/O
- Registration
- Rigid transformation
- Point cloud stitching
- Downsampling
- Import point clouds from Microsoft Kinnect
- Geometric shape fitting
- Streaming point cloud viewer

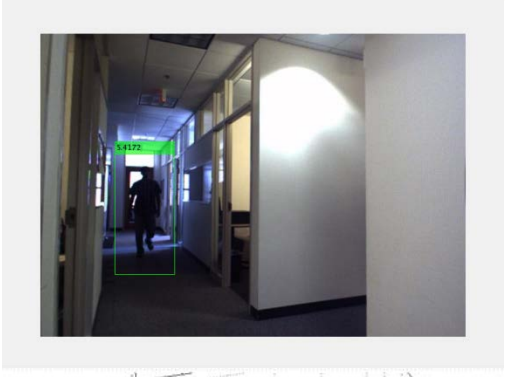


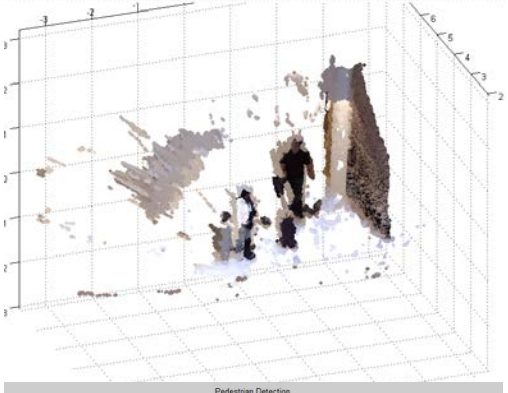
Camera Calibration

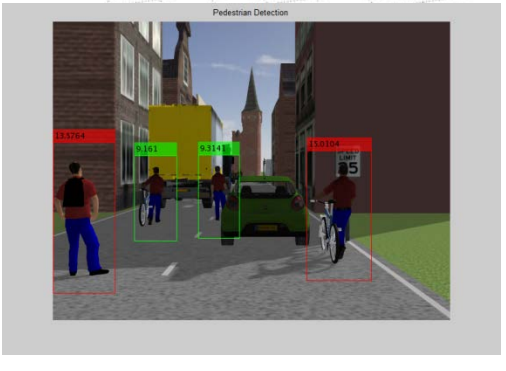
- Single-camera calibration functions and interactive app
- Stereo-camera calibration functions an interactive app
- Estimation of camera calibration errors

Multiple View Geometry

- Stereo vision
  - Stereo rectification
  - Disparity calculation
  - 3-D scene reconstruction
- Fundamental matrix estimation
- Camera pose estimation
- Camera location visualization
- Triangulation

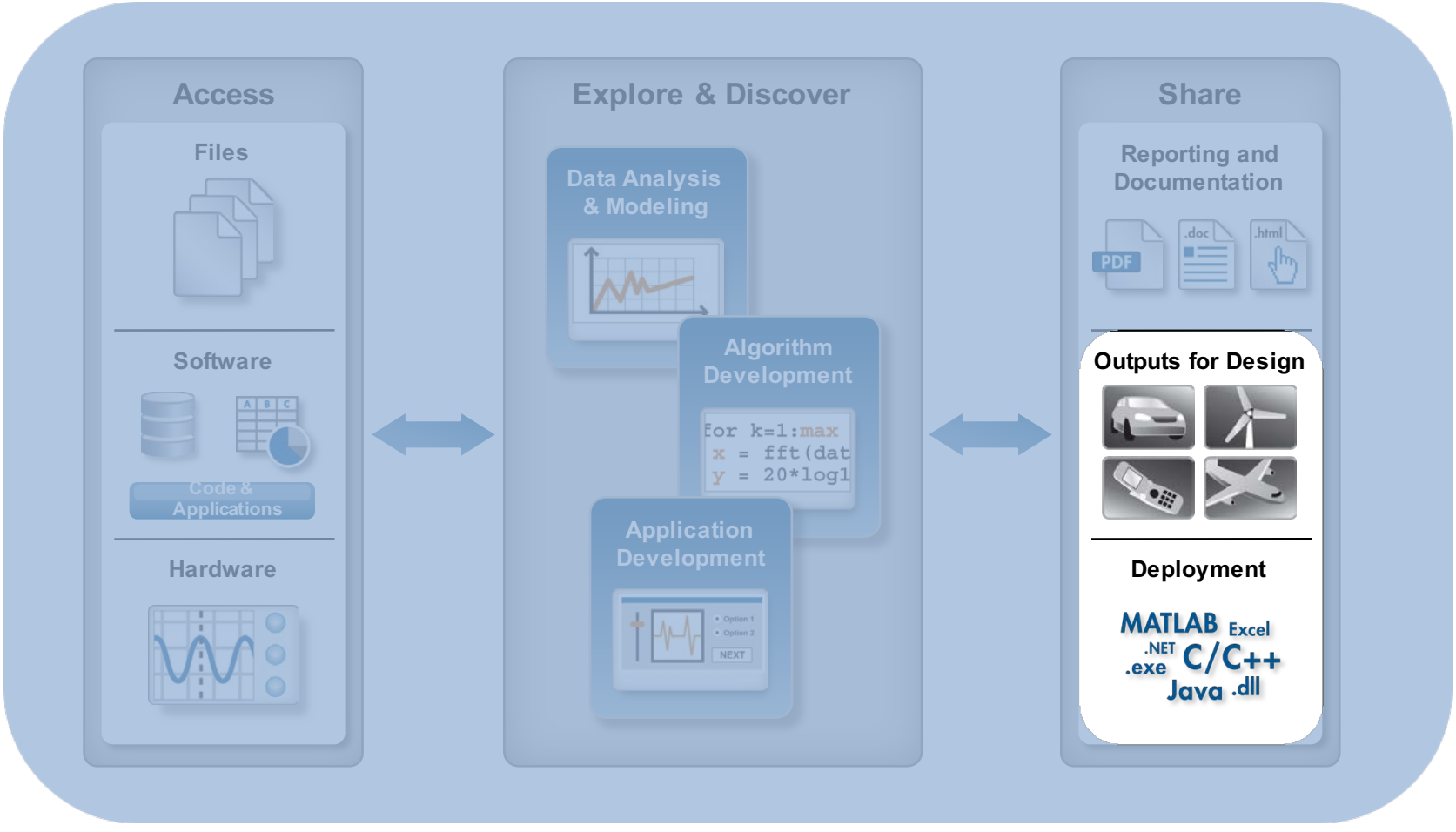






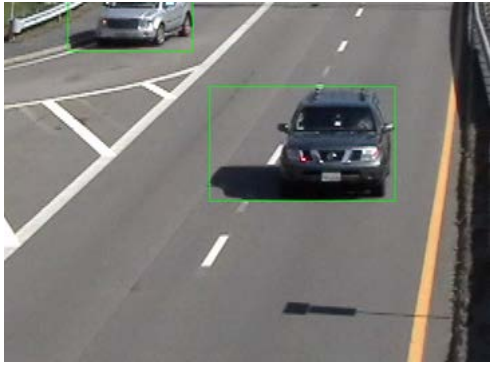
Camera Calibrator Demo

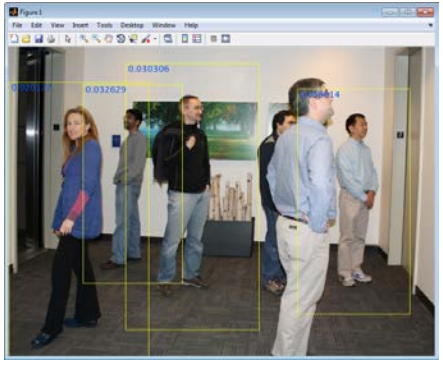
Code Generation

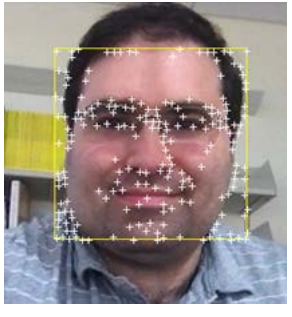


Generate C Code from your Algorithms

- Requires MATLAB Coder
- 80 Image Processing Toolbox functions
- Most Computer Vision System Toolbox functions







A Lot Has Happened in Five Years!

- Access
  - Files
  - Software
  - Hardware
- Graphics
- Math
- Software Development
- Image Processing
- Computer Vision

A Lot Has Happened in Five Years!