Release Notes for Mapping Toolbox™

How to Contact MathWorks

www.mathworks.com	Web
comp.soft-sys.matlab	Newsgroup
www.mathworks.com/contact_TS.html	Technical Support

suggest@mathworks.com	Product enhancement suggestions
bugs@mathworks.com	Bug reports
doc@mathworks.com	Documentation error reports
service@mathworks.com	Order status, license renewals, passcodes
info@mathworks.com	Sales, pricing, and general information

508-647-7000 (Phone)

508-647-7001 (Fax)

The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

Release Notes for Mapping Toolbox™

Trademarks

Patents

Contents

R2012a

R2011b

R2011a

R2010b

R2010a

R2007a

R2006b

R2013a

Version: 3.7
New Features: Yes
Bug Fixes: Yes

KML export for line features

The new kmlwriteline function writes a geographic line specified by latitude and longitude coordinate vectors, and optionally an altitude vector, to a KML file. The existing kmlwrite function can now write geographic line features from a geoshape vector or line geostruct vector to a KML file.

Additional KML attributes for both points and lines: `AltitudeMode, Camera, Color,` **and** `LookAt`

The `kmlwrite` function can now include additional KML attributes when writing exporting geographic points or lines to a KML file: `AltitudeMode`, `Camera`, `Color`, `LookAt`, and `Width`.

3

Multiple track log and route import in `gpxread` function

The `gpxread` function can now read multiple track logs or routes from a GPX file

Dynamic vector input in `shapewrite`, `geoshow`, and `mapshow`

The `geoshow`, `mapshow`, `shapewrite` and `kmlwrite` functions now accept dynamic vectors as input. The following table lists the functions and identifies the specific dynamic vectors that they accept.

	geoshow	mapshow	shapewrite	kmlwrite
geopoint	Yes	—	Yes	Yes
mappoint	—	Yes	Yes	—
geoshape	Yes	—	Yes	Yes
mapshape	—	Yes	Yes	—

Auxiliary latitude converter objects

The toolbox includes several new classes, listed below, that provide methods for performing conversions between geodetic latitude and the four types of auxiliary latitude commonly used to implement map projections: authalic, conformal, isometric, and rectifying.

`map.geodesy.authaliclatitudeConverter`	Convert between geodetic and authalic latitudes
`map.geodesy.conformallatitudeConverter`	Convert between geodetic and conformal latitudes
`map.geodesy.isometriclatitudeConverter`	Convert between geodetic and isometric latitudes
`map.geodesy.rectifyinglatitudeConverter`	Convert between geodetic and rectifying latitudes

When using these classes, make sure there is no variable named `map` in the same workspace. The existence of a variable named `map` will cause MATLAB® to return an error with the message:

`"Attempt to reference field in non-structure array."`

Additional KML enhancements

The toolbox supports the following enhancements to KML support:

- The new `kmlwritepoint` function writes a geographic line specified by latitude and longitude coordinate vectors, and optionally an altitude vector, to a KML file

- The `kmlwrite` function can now export geographic point features from a geopoint vector or geoshape vector to a KML file.

- The `kmlwrite` function now accepts a vector of altitudes in addition to latitude and longitude coordinate vectors.

Geocentric and parametric latitude functions

Four new functions, `geocentricLatitude`, `geodeticLatitudeFromGeocentric`, `parametricLatitude`, and `geodeticLatitudeFromParametric` provide conversions between geodetic latitude and either geocentric latitude or parametric latitude.

Predicate for checking and validating angle unit strings

The new `map.geodesy.isDegree` function provides an easy consistent mechanism for checking and validating angle unit string inputs in which the alternatives `'degrees'` and `'radians'` are supported

When using the `map.geodesy.isDegree` function, make sure there is no variable named `map` in the same workspace. The existence of a variable named `map` will cause MATLAB to return an error with the message:

`"Attempt to reference field in non-structure array."`

Support for PolarStereographic (Variant B) Projection

The functions geotiffinfo, geotiff2mstruct, projfwd, and projinv now support the PolarStereographic (Variant B) projection, which is used in various coordinate systems such as WGS84/Antarctic Polar Stereographic.

Enhancements to geoshape and mapshape classes

The dynamic shape classes `geoshape` and `mapshape` have been made more robust with respect to possible data corruptions given invalid inputs in set operations.

Improved performance for gpxread function

Enhanced performance for reading GPX files containing waypoints using
the `gpxread` function.

WMS Database Modified

The WMS Database changes on a release-to-release basis, as some new servers are added and other servers are removed because they are no longer online or because their availability is too sporadic. A total of 279 servers (10.6% of the number of servers listed in R2012b) and 21,761 layers have been removed from the database.

A total of 17 new servers, with 1,896 layers, have been added. The new database contains a total of 2,378 servers and 76,283 layers.

If you want to find a server, use the server URL or a server URL search string with `wmsfind` to search for layers provided by the server or servers. Then use the `servers` method of the WMSLayer object to obtain the server or server URLs:

```
layers = wmsfind(urlSearchString, 'SearchField', 'serverurl')
servers = layers.servers
```

The following specific update has been made to the WMS Database since the last release:

- The USGS retired the Seamless Server on July 31, 2012. Services have been moved to the National Map at **'nationalmap.gov'**. Search for equivalent layers using the `urlSearchString` 'nationalmap.gov'.

Functions Being Removed
Compatibility Considerations: Yes

Function Name	What Happens When You Use the Function?	Use This Instead	Compatibility Considerations
combntns	Still runs	nchoosek	Replace all existing instances of combntns with nchoosek.
epsm	Still runs		If necessary, you can replace the expressions below with the constants to the right: epsm() 1.0E-6 epsm('deg') 1.0E-6 epsm('rad') deg2rad(1.0E-6)
cometm	Still runs		Replace instances of cometm with the following: [x,y] = mfwdtran(lat,lon); comet(x,y,p)
cometm3	Still runs		Replace instances of cometm3 with the following: [x,y,z] = mfwdtran(lat,lon,z); comet3(x,y,z,p)
restack	Still runs	uistack	Replace all existing instances of restack with uistack

Function Name	What Happens When You Use the Function?	Use This Instead	Compatibility Considerations
grepfields	Still runs	textscan	Replace all existing instances of grepfields with textscan
fipsname	Still runs	shaperead	Import the more recent TIGER/Line data set, available in shapefile format, using shaperead
colorui	Warning	uisetcolor	Replace all existing instances of colorui with uisetcolor
dcwdata	Still runs	vmap0data	The VMAP0 dataset has replaced DCW and can be accessed using vmap0data.
dcwgaz	Still runs	vmap0ui.	The VMAP0 dataset has replaced DCW and can be explored using vmap0ui.

15

Function Name	What Happens When You Use the Function?	Use This Instead	Compatibility Considerations
dcwread	Still runs	vmap0read	The VMAP0 dataset has replaced DCW and can be read using vmap0read.
dcwrhead	Still runs	vmap0rhead	The VMAP0 dataset has replaced DCW, the header data for which can be read using vmap0rhead.

R2012b

Version: 3.6
New Features: Yes
Bug Fixes: Yes

Dynamic representation of geographic line and polygon features with geoshape class

Geographic multi-point, line, and polygon features, in a geographic coordinate system, are represented by a geoshape vector.

Dynamic representation of point, line, and polygon map features with `mappoint` and `mapshape` classes

Geographic point feature, in a planar coordinate system, are represented by a `mappoint` vector. Geographic multi-point, line, and polygon features, in a planar coordinate system, are represented by a `mapshape` vector.

Coordinate transformations to/from local east-north-up, north-east-down, and spherical systems

A new set of 20 functions for transforming between 3-D coordinate systems has been introduced. There are two global coordinate systems: the geodetic system and ECEF (Earth-Centered, Earth-Fixed) system. The three local coordinate systems are ENU (east-north-up), NED (north-east-down), and AER (azimuth-elevation-range).

- Geodetic to local coordinate transforms: `geodetic2enu`, `geodetic2ned`, `geodetic2aer`

- ECEF to local coordinate transforms: `ecef2enu`, `ecef2ned`, `ecef2aer`

- Local to geodetic coordinate transforms: `enu2geodetic`, `ned2geodetic`, `aer2geodetic`

- Local to ECEF coordinate transforms: `enu2ecef` , `ned2ecef`, `aer2ecef`

- Transformations between local systems: `aer2enu`, `aer2ned`, `enu2aer`, `ned2aer`

- 3-D vector transformations between the three Cartesian systems (ECEF, ENU and NED): `enu2ecefv`, `ned2ecefv`, `ecef2enuv`, `ecef2nedv`

Geographic quadrangles bounding points and lines with `geoquadpt` and `geoquadline` functions

`geoquadpt` computes a geographic quadrangle bounding scattered points. `geoquadline` computes a geographic quadrangle bounding a multi-part line. Both functions account for spherical topology.

Expanding latitude-longitude quadrangle with `bufgeoquad` **function**

The `bufgeoquad` function expands the latitude and longitude limits of geographic quadrangle, accounting for spherical topology.

Spheroid class methods for 3-D coordinate transformations
Compatibility Considerations: Yes

The three spheroid classes, `oblateSpheroid`, `referenceEllipsoid`, and `referenceSphere`, include 3-D transformation methods using geodetic and Earth-Centered Earth-Fixed (ECEF) Cartesian coordinates.

- `geodetic2ecef` - transforms geodetic to geocentric (ECEF) coordinates

- `ecef2geodetic` - transforms geocentric (ECEF) to geodetic coordinates

- `ecefOffset` - Computes Cartesian ECEF offset between geodetic positions

These methods can be used with either degrees or radians.

Compatibility Considerations

If you choose to replace calls to the existing `geodetic2ecef` and `ecef2geodetic` functions with calls to the new methods of the same names, be aware that the methods use latitude and longitude in units of degrees, but the functions assume units of radians.

Option to use in degrees in unwrapMultipart

The `unwrapMultipart` function now accepts an optional angle unit string, which can match either `'degrees'` or `'radians'`.

Changes in geopoint class
Compatibility Considerations: Yes

A new collection property, `Geometry`, has been added to the `geopoint` class. Its value is always `'point'`.

Compatibility Considerations

- The `lat` and `lon` inputs are restricted to either class type single or double. In R2012a, the `lat` and `lon` inputs may be any numeric type.

- If a dynamic property is set with a cell array of values, the class type of the values are restricted to strings. In R2012a, the class type of the values in the cell array input may be numeric, logical, or strings.

- When the input coordinate vectors are of different lengths, the lengths of the Latitude and Longitude property values are set to the longest length of the input vectors. In R2012a, the lengths of the property values is set to the length of the Longitude property.

Links to Internet geodata resources moved to Mapping Toolbox documentation
Compatibility Considerations: Yes

The information on finding geospatial data on the Internet, previously located in "Tech Note 2101" at URL http://www.mathworks.com/support/tech-notes/2100/2101.html has been moved into the Mapping Toolbox documentation. The same content and links to external data sources, with some updates and improvements, can be found at the URL:http://www.mathworks.com/help/map/finding-geospatial-data.html

Compatibility Considerations

The MathWorks web site provides a seamless redirect from the old tech note URL to the new one in the web-based documentation, but If you have any browser favorites or bookmarks to the old tech note URL, you could update them.

Smoother colormap interpolation in function `demcmap`
Compatibility Considerations: Yes

The `demcmap` function uses a smoother colormap interpolation scheme.

Compatibility Considerations

Differences in colors may occur from earlier versions, but the change should be barely perceptible.

Change in gshhs output structure
Compatibility Considerations: Yes

In the structure returned by the `gshhs` function, the field name
`CrossGreenwich` has changed to `CrossesGreenwich`.

Compatibility Considerations

In scripts, or other MATLAB files, that refer to it, the field name
`CrossGreenwich` needs to be changed to `CrossesGreenwich`.

New method option in `intrplat` and `intrplon`
Compatibility Considerations: Yes

The `intrplat` and `intrplon` functions now accept the method string `'pchip'`, which designates shape-preserving piecewise cubic interpolation (as in the MATLAB `interp1` function).

Compatibility Considerations

The string `'cubic'` is still accepted, but is now synonymous with `'pchip'`. Calls to `intrplat` and `intrplon` that use 'cubic' may interpolate slightly different latitude and longitude values.

Certain sample data files can be included in a deployed application

The geoid MAT-file and the following shapefiles can now be included, using the '-a' option, when using the MATLAB Compiler (TM) to build an application that uses Mapping Toolbox: landareas, usastatehi, usastatelo, worldcities, worldlakes, worldrivers.

WMS Database Modified

The WMS Database changes on a release-to-release basis, as some new servers are added and other servers are removed because they are no longer online or because their availability is too sporadic. A total of 7044 servers (75% of the number of servers listed in R2012a) and 15,488 layers have been removed from the database. The vast majority (6,982) of the servers no longer available are from the following server which has either changed its URL or is no longer in service.

```
http://nomads.ncdc.noaa.gov/thredds/wms/ncdcPaleoClimate
```

A total of 284 new servers, with 24,675 layers, have been added. The new database contains a total of 2,636 servers and 96,417 layers.

If you want to find a server, use the server URL or a server URL search string with wmsfind to search for layers provided by the server or servers. Then use the servers method of the WMSLayer object to obtain the server or server URLs:

```
layers = wmsfind(urlSearchString, 'SearchField', 'serverurl')
servers = layers.servers
```

The following specific updates have been made to the WMS Database since the last release:

- 19 new WMS servers from the USGS National Map Server. These servers provide ortho-imagery, land cover, scanned topo maps, and shaded relief layers. Search for the layers and servers using the urlSearchString **'isse.cr.usgs.gov'** or **'nationalmap.gov'**.

- 3 new WMS servers from the Intergovernmental Panel on Climate Change (IPCC). Search for the layers and servers using the urlSearchString **'ipcc-data.org'**.

- 29 new WMS servers from the University of San Diego focusing on natural disasters. Search for the layers and servers using the urlSearchString **'hyperquad.ucsd.edu'**.

- 33 new WMS servers from the USGS Energy Resources Program (http://energy.usgs.gov/). Search for the layers and servers using the urlSearchString **'certmapper.cr.usgs.gov'** .

The USGS is moving services from **imsortho.cr.usgs.gov** to **raster.nationalmap.gov**.

Some notable servers that have been removed are:

```
http://imsortho.cr.usgs.gov:80/wmsconnector/com.esri.wms.Esrimap/USGS_EDC_O
http://imsortho.cr.usgs.gov:80/wmsconnector/com.esri.wms.Esrimap/USGS_EDC_O
http://imsortho.cr.usgs.gov:80/wmsconnector/com.esri.wms.Esrimap/USGS_EDC_O
http://imsortho.cr.usgs.gov:80/wmsconnector/com.esri.wms.Esrimap/USGS_EDC_O
```

Search for equivalent layers using the `urlSearchString` **'isse.cr.usgs.gov'** or **'nationalmap.gov'**.

Microsoft has retired the TerraServer. The following servers have been removed.

```
http://terraserver-usa.com/ogccapabilities.ashx?
http://terraserver-usa.net/ogccapabilities.ashx?
http://terraservice.net/ogccapabilities.ashx?
http://columbo.nrlssc.navy.mil/ogcwms/servlet/WMSServlet/TerraServer.wms
```

Search for equivalent layers in the USGS National Map server by using the following urlSearchStrings:

- **'nationalmap.gov*Ortho'**
- **'nationalmap.gov*Scanned'**
- **'nationalmap.gov*DRG'**
- **'nationalmap.gov*Imagery'**
- **'isse*USGS_EDC_Ortho_HRO'**

R2012a

Version: 3.5
New Features: Yes
Bug Fixes: No

Data File Removal or Location Change
Compatibility Considerations: Yes

The following data files have been removed. (The data has been inlined in source code.)

```
toolbox/map/mapdisp/globedems.dat
toolbox/map/mapdisp/gtopo30s.dat
```

Also, the usgsdems.dat file has moved from mapdisp:

```
toolbox/map/mapdisp/usgsdems.dat
```

to mapformats:

```
toolbox/map/mapformats/usgsdems.dat
```

Compatibility Considerations

Before R2011b, you needed to include these three data files:

```
toolbox/map/mapdisp/globedems.dat
toolbox/map/mapdisp/gtopo30s.dat
toolbox/map/mapdisp/usgsdems.dat
```

and the −a flag when compiling code that used the functions, globedems, gtopo30s or usgsdems. You no longer need to do this.

geotiffinfo Now Handles Noncompliant GeoTIFF Files
Compatibility Considerations: Yes

Changes to the `geotiffinfo` function allow it to handle non-compliant GeoTIFF files better than in previous releases. If the `GTModelTypeGeoKey` is not set, `geotiffinfo` now issues a warning, assumes that the model type is `'ModelTypeProjected'`, and constructs a `spatialRef.MapRasterReference` object and a corresponding `RefMatrix`. If the `GTModelTypeGeoKey` is set to the value 3 (geocentric model), then `geotiffinfo` sets the `ModelType` field to `'ModelTypeGeocentric'` rather than empty (as in previous releases). When the `ModelTiepointTag` contains a nonsensical corner latitude, `geotiffinfo` issues a warning and clamps the corner latitude value to the interval `[-90 90]`.

Compatibility Considerations

Before R2012a, if a GeoTIFF file did not include either the `ModelTypeGeographic` or `ModelTypeProjected` tag, the `geotiffinfo` `RefMatrix` and `SpatialRef` fields were empty. The empty tags led to the creation of degenerative files.

geopoint Class to Hold Geographic Point Data

The new geopoint class provides a convenient, memory-efficient way to represent one or more geographic points. A geopoint vector can include a set of non-geographic attributes for each point. (When used for cities, for example, attributes might include name, country, population, and so on.) The geopoint class provides a rich set of properties and methods to describe, access and modify the geographic point data.

gpxread Function to Read GPX Files

Use the new `gpxread` function to read data from a GPX file. It enables import of GPS waypoints, routes, and track logs into MATLAB, via the GPS Exchange Format (GPX).

geotiffinfo Now Sets Filename to URL String
Compatibility Considerations: Yes

If the input to `geotiffinfo` is a URL, then the value in the Filename field of the output structure equals the URL.

Compatibility Considerations

Before R2012a, the value in the Filename field was a temporary file name.

Length Unit Validation and Conversion

The new `validateLengthUnit` function validates and standardizes a length
unit string. It accepts a wide variety of abbreviations, and both plural and
singular forms. The call `validateLengthUnit(`km')` returns `kilometer'.
The `unitsratio` function now handles several additional length units that
are used for geodetic applications in different parts of the world.

Improved Reference Spheroid Representations and Support
Compatibility Considerations: Yes

New `referenceEllipsoid` and `referenceSphere` classes provide intuitive, self-documenting representations of reference ellipsoids and spheres, with name and length unit properties as well as geometric properties, The reference ellipsoid class is based on the new `oblateSpheroid` class, which encapsulates the purely geometric aspects of a flattened ellipsoid of revolution.

You can easily construct a reference ellipsoid object for most commonly used coordinate systems, including all those supported by the `almanac` function and those included in the EPSG/OGP Geodetic Parametric Dataset (which is used in connection with the GeoTIFF Format). Similarly, you can construct reference sphere objects representing spherical models of the Earth, Sun, Moon and planets.

wgs84Ellipsoid Function

Many users today work exclusively in the World Geodetic System of 1984 (WGS 84). Along with many other roles, it serves as the native coordinate system the NAVSTAR Global Positioning System (GPS). All that may be needed in this case is the new `wgs84Ellipsoid` function, which returns a referenceEllipsoid object with property settings appropriate to the WGS `84 ellipsoid.

Compatibility Considerations

In addition to the earlier "ellipsoid vector" representation, the following functions have been extended to work with reference ellipsoid, oblate spheroid, and reference sphere objects:

`areaint`, `areamat`, `areaquad`, `axesm`, `azimuth`, `convertlat`, `defaultm`, `departure`, `distance`, `ecef2geodetic`, `ecef2lv`, `elevation`, `ellipse1`, `eqa2grn`, `geodetic2ecef`, `gradientm`, `grn2eqa`, `hista`, `lv2ecef`, `mapprofile`, `meanm`, `meridianarc`, `meridianfwd`, `mfwdtran`, `minvtran`, `rcurve`, `reckon`, `rsphere`, `scircle1`, `scircle2`, `setm`, `stdist`, `stdm`, `track`, `track1`, `track2`, `vfwdtran`, `vinvtran`

An "ellipsoid vector" is a 2–by-1 double having the form [semimajor_axis eccentricity]. Ellipsoid vectors are not self-identifying, they do not have a name property, and the length unit of the semimajor axis must be known and managed separately.) For backward compatibility, these functions continue to support ellipsoid vectors as well as the new representations.

`referenceSphere`, `referenceEllipsoid` and `wgs84Ellipsoid` provide superior alternatives to the `almanac` function and should be used in its place going forward.

The etopo function now supports reading additional ETOPO1 data sets

The complete set of ETOPO1 supported data sets is as follows:

```
etopo1_ice_c.flt
etopo1_bed_c.flt
etopo1_ice_c_f4.flt
etopo1_bed_c_f4.flt
etopo1_ice_c_i2.bin
etopo1_bed_c_i2.bin
```

Improvement to Functions usamap and worldmap
Compatibility Considerations: Yes

In the functions `usamap` and `worldmap` the axes are initialized with a spherical
Earth model having a radius of 6,371,000 meters rather than with a unit
sphere. 3D viewing has been made more robust. The options `'all'` and
`'allequal'` are now equivalent. In future releases `'allequal'` will be
removed.

Compatibility Considerations

This radius change affects the X and Y limits of the axes. If you are setting the
CameraPosition, CameraTarget, CameraUpVector, or CameraLightPosition
properties of the axes, with hardwired values determined in releases prior to
R2012a, then you need to multiply the first two elements (X and Y values) by
6,371,000. Likewise, if you are using the XLoc or YLoc properties to position a
scaleruler, you need to multiply their values by 6,371,000.

43

WMS Database Modified

The WMS Database changes on a release-to-release basis, as some new servers are added and other servers are removed because they are no longer online or because their availability is too sporadic. A total of 265 servers (2.75% of the number of servers listed in R2011b) and 13,505 layers have been removed from the database. A total of 29 new servers, with 3,589 layers, have been added. The new database contains a total of 9,396 servers and 87,230 layers.

If you want to find a server, use the server URL or a server URL search string with `wmsfind` to search for layers provided by the server or servers. Then use the `WMSLayer.servers` method to obtain the server or server URLs:

```
layers = wmsfind(urlSearchString, 'SearchField', 'serverurl')
servers = layers.servers
```

A notable server that has been removed is:

```
http://aes.gsfc.nasa.gov/cgi-bin/wms?
```

Please use the string `"gsfc.nasa.gov"` rather than `"gsfc.nasa.gov` when finding layers from the NASA SVS Image Server.

R2011b

Version: 3.4
New Features: Yes
Bug Fixes: No

New contourcbar Function Creates Color Bar for Filled Contour Display

Use the `contourcbar` function to create a color bar associated with a filled contour display created with `contourfm`, `contourm`, `contour3m`, or `geoshow`.

Support for Web Map Service Version 1.3.0

Mapping Toolbox™ functions and classes now support Web Map Service (WMS) Version 1.3.0. See the tip in the `wmsread` reference page about how EPSG:4326 coordinates are encoded in WMS Version 1.3.0.

WMS Database Modified

The WMS Database changes on a release-to-release basis, as some new servers are added and other servers are removed because they are no longer online or because their availability is too sporadic. A total of 151 servers (7.52% of the number of servers listed in R2011a) and 16,466 layers have been removed from the database. A total of 7,768 new servers, with 30,525 layers, have been added. The new database contains a total of 9,632 servers and 97,146 layers.

If you want to find a new server, use the server URL or a server URL search string with `wmsfind` to search for layers provided by the server or servers. Then use the `WMSLayer.servers` method to obtain the server or server URLs:

```
layers = wmsfind(urlSearchString, 'SearchField', 'serverurl')
servers = layers.servers
```

The following updates have been made to the WMS Database since the last release:

- 7,708 new WMS servers from Unidata's Thematic Realtime Environmental Distributed Data Services (THREDDS) project. These servers are provided through several different institutions. Search for the layers and servers using the urlSearchString: `'thredds'`.

- 7 new WMS servers from various institutions that provide layers focused on Japan:

```
 http://cernunosat05.cern.ch/ArcGIS/services/Japan/...
    Japan_earthquake_Tsunami_area/MapServer/WMSServer?

 http://cernunosat05.cern.ch/arcgis/services/Japan/...
    SendaiMosaic/ImageServer/WMSServer?

 http://hazardmap.service-section.com/cgi-bin/...
    mapserv?map=/map/UserRaster/alav2a_O312_1.map

 http://hyperquad.telascience.org/cgi-bin/jp_earthquake?

 http://openls.geog.uni-heidelberg.de/geoserver/wms?
```

```
http://ows.geogrid.org/JapanBaseMap?

http://www.geographynetwork.ne.jp/ogc/wms?
```

- 21 new servers from the European Space Agency's ERDAS Apollo servers.
  Use the search string `'erdas.esrin'` to search for them.

Two of the servers from the European Space Agency (ESA) are no longer
available:

```
http://mapdev.eo.esa.int/mapServer/mapServer

http://mapdev.esrin.esa.int/mapServer/mapServer
```

You can find many of the layers from these servers on the new ERDAS
Apollo servers. These changes have resulted in slight modifications
to the documentation examples for the `WebMapServer.getMap` and
`WebMapServer.updateLayers` methods since the global MODIS layer has
moved.

Some notable servers that have been removed are:

```
http://ims.cr.usgs.gov/wmsconnector/...
    com.esri.wms.Esrimap/USGS_EDC_Ortho_StateLocal?

http://ims.cr.usgs.gov/wmsconnector/...
    com.esri.wms.Esrimap/USGS_EDC_Ortho_Urban?

http://nhdgeo.usgs.gov/wmsconnector/...
    com.esri.wms.Esrimap/nhdgeowms?

http://nmcatalog.er.usgs.gov/catalogwms/base
```

```
http://nmcatalog.usgs.gov/catalogwms/base

http://columbo.nrlssc.navy.mil/ogcwms/servlet/...
   WMSServlet/OpenGIS_Web_Mapping_Services_(WMS).wms?
```

Changes to Error and Warning Identifiers
Compatibility Considerations: Yes

In R2011b, some error and warning message identifiers in Mapping Toolbox have changed.

Compatibility Considerations

If you have scripts or functions that use specific identifiers, you must update the code to use the new identifiers. Typically identifiers are used to turn off specific warnings, or in code that uses a `try/catch` statement and performs an action based on a specific error identifier.

For example, the `map:eastof:obsolete` identifier has changed to `map:removing:eastof`. If your code checks for `map:eastof:obsolete`, you must update it to check for `map:removing:eastof` instead.

To determine the identifier for a warning, run the following command just after you see the warning:

```
[msg,msgid] = lastwarn;
```

This command saves the message identifier to the variable `msgid`.

To determine the identifier for an error, run the following commands just after you see the error:

```
exception = MException.last;
msgid = exception.identifier;
```

For a mapping of the new warning identifiers to the original identifiers, see the solution Why is my code that includes Mapping Toolbox message identifiers not working?

New Location for Sample Data

The Mapping Toolbox sample data sets, such as `coast.mat` and `boston.tif`, moved from `toolbox/map/mapdemos` to `toolbox/map/mapdata`. All these data sets are still on the MATLAB path, but it's helpful to know their specific location if you want to peruse them (or their attributions), or use them to try out the Map Viewer. Also note that the sample SDTS DEM data has moved into its own subfolder, `sdts`, within the `mapdata` folder.

almanac Function Now Returns More Precise Eccentricity Value
Compatibility Considerations: Yes

When used with parameter `'airy'`, the `almanac` function now returns an eccentricity value derived from a full-precision value of inverse flattening.

Compatibility Considerations

In previous releases, the inverse flattening value used to calculate eccentricity was truncated and therefore less precise.

Note When `'airy'` is used, almanac returns an ellipsoid vector for the Airy 1830 reference ellipsoid. This has always been the case and has not changed in R2011b, but through R2011a the documentation incorrectly indicated that `'airy'` was the designation for the Airy 1849 reference ellipsoid. In R2011b this documentation error has been corrected. There is no Airy 1849 option in almanac, but if you need to you can construct an ellipsoid vector for the 1849 ellipsoid as follows:

```
[6377.340189   flat2ecc(1/299.3249646)]
```

In this case, the semi-major axis length is given in kilometers, which is consistent with the length unit default of almanac.

R2011a

Version: 3.3
New Features: Yes
Bug Fixes: No

Spatial Referencing Improvements
Compatibility Considerations: Yes

Raster Referencing Classes

The new `spatialref.GeoRasterReference` and `spatialref.MapRasterReference` classes relate georeferenced images or data to geographic or planar coordinates. Most Mapping Toolbox functions that work with referencing vectors and matrices now work with referencing objects, as well. Unlike the older referencing matrix and vector representations, a referencing object is self-documenting, providing a rich set of properties to describe both the intrinsic and extrinsic geometry.

These functions now work with `GeoRasterReference` objects:

areamat	contour3m	contourfm	contourm
filterm	findm	geoshow	geotiffinfo
geotiffread	gradientm	grid2image	imbedm
latlon2pix	limitm	los2	ltln2val
mapprofile	maptrims	meshgrat	meshlsrm
meshm	neworig	pix2latlon	resizem
setltln	setpostn	usamap	vec2mtx
viewshed	worldfileread	worldfilewrite	worldmap

These functions now work with `MapRasterReference` objects:

geotiffinfo	geotiffread	map2pix	mapbbox
mapoutline	mapshow	mapview	pix2map
pixcenters	worldfileread	worldfilewrite	

Use the new `georasterref` and `maprasterref` functions to construct `GeoRasterReference` and `MapRasterReference` objects.

Compatibility Considerations

Use the new referencing classes instead of referencing matrices and referencing vectors. To convert referencing matrices or referencing vectors to the `GeoRasterReference` class, use the conversion functions `refvecToGeoRasterReference` and `refmatToGeoRasterReference` as shown:

```
R = refvecToGeoRasterReference(refvec, rasterSize)
R = refmatToGeoRasterReference(refmat, rasterSize)
```

To convert a referencing matrix to the `MapRasterReference` class, use the conversion function `refmatToMapRasterReference` as shown:

```
R = refmatToMapRasterReference(refmat, rasterSize)
```

New geotiffwrite Function to Write GeoTIFF Files

The `geotiffwrite` function exports georeferenced images or data. Now, in addition to reading georeferenced images with `geotiffread`, you can also write them, with `geotiffwrite`.

WMS Database Modified

The WMS Database changes on a release-to-release basis, as some new servers are added and some unavailable servers are removed. A total of 666 servers (26.8% of the number of servers listed in R2010b) and 234,156 layers have been removed from the database. A total of 190 new servers, with 16,395 layers, have been added. The new database contains a total of 2,023 servers and 83,087 layers. Since the number of layers stored in the database is significantly reduced from earlier versions, the access time is quicker.

If you want to find one of the new servers, use `wmsfind` to search for the URL:

```
wmsfind(URL, 'SearchField', 'serverurl')
```

The following updates have been made to the WMS Database since the last release:

- 57 new WMS servers from NOAA's Environmental Research Division Data Access Program (ERDDAP):

  ```
  http://coastwatch.pfeg.noaa.gov/erddap/wms
  ```

  These servers provide oceanographic data. To find out more about them, visit the ERDDAP web site.

- Two new WMS servers from the U.S. Geological Survey Coastal and Marine Program ncWMS program:

  ```
  http://coast-enviro.er.usgs.gov/ncWMS/wms)
  ```

- A new Mars Space Flight Facility MapServer:

  ```
  http://ms.mars.asu.edu/TES_TI_Putzig?
  ```

- Two new servers from the Bureau of Land Management in partnership with the U.S. Forest Service:

  ```
  http://www.geocommunicator.gov
  ```

- A new Metacarta WMS server providing images from VMP0 tiles:

  ```
  http://vmap0.tiles.osgeo.org/wms/vmap0
  ```

- A new server from the NASA Goddard Space Flight Institute providing data for the Tropical Rainfall Measuring Mission (TRMM):

```
http://gdata2.sci.gsfc.nasa.gov/daac-bin/wms_trmm?
```

- This version of the database is significantly reduced from earlier versions primarily due to the reduction of servers (servelets) hosted by the NRL GIDB Portal server:

```
http://columbo.nrlssc.navy.mil
```

  In R2010b, 425 servers with 228,227 layers were listed in the database. At the time of qualification, the portal server is hosting data from only 51 servers, with a total of 10,715 layers.

- The JPL Global Imagery Service server:

```
http://onearth.jpl.nasa.gov/wms.cgi?
```

  is no longer providing full WMS services for any of the datasets. Any server (e.g., `http://webapps.datafed.net/OnEarth_JPL.ogc?`) that cascades data from this server is also affected by the change. The server is still included in the database. Examples in the help and reference pages that referred to this server have been updated to use a different server.

  A small subset of the data can be accessed using a non-standard `TiledWMS` request. The available tiled patterns can be found at:

```
http://pat.jpl.nasa.gov/wms.cgi?request=GetTileService
```

  The WMS parameters must be in the exact order. If you wish to obtain a tile, you can use the prefix:

```
 'http://onearth.jpl.nasa.gov/wms.cgi?/SERVICE=WMS&'
```

  in front of the request (found in the CDATA section of the `GetTileService` request). For example:

```
url = ['http://onearth.jpl.nasa.gov/wms.cgi?/SERVICE=WMS&' ...
   'request=GetMap&layers=global_mosaic&srs=EPSG:4326&'  ...
   'format=image/jpeg&styles=visual&width=512&height=512&' ...
   'bbox=-180,58,-148,90'];
[A, R] = wmsread(url);
```

Enhancements to geotiffinfo, geotiffread, and worldfileread

The `geotiffinfo`, `geotiffread`, and `worldfileread` functions now have additional syntax options. Also, the `geotiffinfo` function now returns information about GeoTIFF tags.

Improved Performance for gtopo30

Enhanced performance for reading GTOPO30 tiles using the `gtopo30` function.

Improved Performance for gshhs

Enhanced performance for reading GSHHS data sets using the gshhs function. The gshhs function has been qualified on GSHHS releases 1.1 through 2.1 (version 8). Also, it can now read even newer versions, if they adhere to the same header format as releases 2.0 and 2.1.

The improved gshhs can now read the files below:

```
wdb_borders_x.b
wdb_rivers_x.b
```

where x is one of the letters c, l, i, h, and f, corresponding to increasing resolution.

Second Input Argument of roundn No Longer Optional
Compatibility Considerations: Yes

The second input argument to roundn, a real, integer-valued exponent n, is no longer optional.

Compatibility Considerations

If you omitted n in previous releases, a warning was issued and a default value of −2 was used. Now, if you omit n, you will receive an error. Change any code that calls roundn with one input argument like this: roundn(x) to this: roundn(x,-2).

Comet Menu Item Removed from maptool

The **Comet** menu item is no longer available in maptool. You can still call cometm directly from the command line.

R2010b

Version: 3.2
New Features: Yes
Bug Fixes: No

MATLAB Plot Selector Now Includes mapshow and geoshow

The Plot Selector workspace tool creates graphs of workspace variables. The mapshow and geoshow functions have been added to the list of possible plotting functions available in the Plot Selector. For more information about the Plot Selector, see Enhanced Plot Selector Simplifies Data Display.

Support for Retrieving Web Map Service Data in Image/BIL Format

Some servers render layers in the `'image/bil'` format as a single band with a class type of `int16` or `int32`. You can now use the `wmsread` function to retrieve this data.

Expanded Data Type Support for mapshow and geoshow

The mapshow and geoshow functions now have expanded class support for raster data display.

WMS Database Modified

The WMS Database changes on a release-to-release basis, as some new servers are added and some unavailable servers are removed. A total of 244 servers (10.4% of the number of servers listed in R2010a) and 111,514 layers have been removed from the database. A total of 380 new servers, with 65,834 layers, have been added. The new database contains a total of 2,502 servers and 300,848 layers. Some notable new servers in the database are:

- 242 new WMS servers from NOAA's Environmental Research Division Data Access Program (ERDDAP) (`http://coastwatch.pfeg.noaa.gov/erddap/wms`). These servers provide oceanographic data, and additional information about them may be found at `http://coastwatch.pfeg.noaa.gov/erddap/info/index.html`.

- A server from the European Space Agency, removed in R2010a but now back in the Database (`http://ssems1.esrin.esa.int/mapServer/mapServer?`).

- 115 new servers from the DataFed Web Map Server (`http://webapps.datafed.net`).

- Two new servers from NASA WorldWind WMS (`http://www.nasa.network.com/elev?`   and `http://www.nasa.network.com/wms?`). The `'elev'` server provides data in the `'image/bil'` format.

- Two new servers from the USGS dedicated to emergency operations. These servers provide imagery of the 2010 oil spill in the Gulf of Mexico:

```
http://hdds.usgs.gov/arcgis/services/...
   201004_OilSpill_GulfOfMexico/MapServer/...
   WMSServer
```

and the 2010 earthquake in Haiti:

```
http://hdds.usgs.gov/ArcGIS/services/...
   201001_Earthquake_Haiti/MapServer/...
   WMSServer?
```

KML Schema Updated to Version 2.2

The KML schema has been updated to Version 2.2.

Population Density Data Added to usastatelo.shp

The `usastatelo` shapefile now contains average population density data by state for the year 2000 from the U.S. Census Bureau Web site.

Elements in korea.mat File Rounded
Compatibility Considerations: Yes

In the korea.mat file, the numbers in the referencing vector (the refvec) were very nearly integer valued. Elements of these variables have been rounded slightly to become exact integers, as follows:

```
[12 45 115]
```

Compatibility Considerations

If you use the new version of the korea.mat file, your results will be slightly different than those obtained with the older version of the file.

Changes in Behavior for Contouring Functions
Compatibility Considerations: Yes

Due to a recent bug fix, the contouring functions now exhibit many
improvements. The bug fix in question, Bug 192285, addressed problems with
contours displayed by contourm, contour3m, and contourfm. The following
figure illustrates filled contours produced by the contourfm function in
R2010b, as compared to R2010a and earlier releases.

```
figure('Color','white');
worldmap world;
load geoid;
contourfm(geoid, geoidrefvec)
```
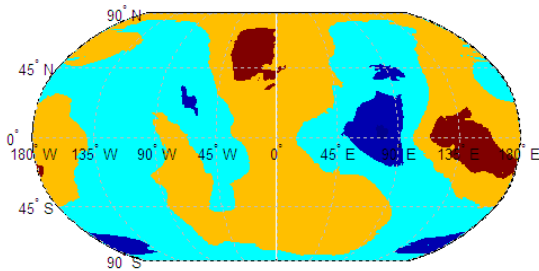

R2010a
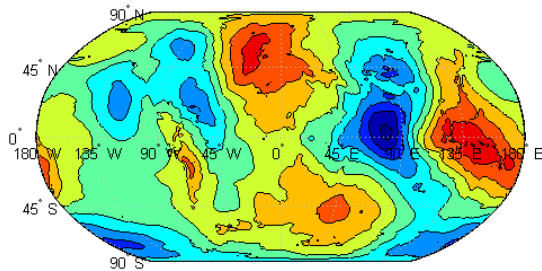
R2010b

Compatibility Considerations

If you call any of the contouring functions, expect the behavior changes
described in the following table.

Summary	New Behavior	Compatibility Considerations
Handles	Each of the three contouring functions now returns a handle to an `hggroup` object as its second output.	In previous releases, `contourm` returned a handle to a MATLAB contourgroup object as its second output, while `contourfm` and `contour3m` each returned an array of patch handles.
Contour lines and levels	The contouring functions now produce one line per contour level. The hggroup returned by `contourm` has exactly one line child per contour level.	The contouring functions now construct an equivalent (or better) display using fewer graphics objects than in previous releases.
Default contour levels	In `contourfm` and `contour3m`, the default contour levels are now consistent with `contourm`, as well as the MATLAB `contour` and `contourf` functions.	The default contour levels have changed from those in previous releases.
Non-positive contour levels	If you supply 0 or a negative number for V in the syntaxes `contourm(Z,R,V)` or `contourm(lat,lon,Z,V)`, `contourm` creates a plot with a single contour at that level.	In previous releases, if you supplied 0 or a negative number for V, the `contourm` function drew no contour lines, returned an empty contour matrix, and constructed a contourgroup with no children.
Parameter settings	The parameter settings for `contourm` and `contourfm` are both more selective and more fully documented. You can set only the parameters described in the `contourm` reference page.	In previous releases, you could set any valid contourgroup property (possibly with unexpected results).

Summary	New Behavior	Compatibility Considerations
Contour level tag	In R2010b, each contour line has its `Tag` property set to a string representation of its contour level, preceded by `'contour line:'`. These tags display in the lower left of the axes when you click on a contour line.	In previous releases, the tags contained only the contour level strings.
Filled area tag	In R2010b, each fill polygon (patch) has its `Tag` property set to a string beginning with `'contour interval:'` and followed by its minimum and maximum levels (as strings) in square brackets. These tags display in the lower left of the axes when you click within a fill polygon.	In previous releases, the tags in patches created by `contourfm` contained the string `'Cpatches'` and did not display when you clicked.
`AppData` and `UserData`	Each contour line object now has a contour `Level` field in its `AppData` property, and the patches representing fill polygons have `'MinLevel'` and `'MaxLevel'` `AppData` fields. `contourm` and related functions no longer set the `UserData` property of any graphics object.	Previously, `UserData` was set to the contour level value for each line and to a minimum contour level value for each patch. If you have an application or GUI that checks `UserData` values for individual lines or patches, work with the `getappdata` function instead.
Contours separating filled areas	By default, the `contourfm` function now draws black contour lines to separate filled areas of different colors.	In previous releases, the function did not draw lines by default. To suppress the lines, specify `'LineColor','none'`.

Summary	New Behavior	Compatibility Considerations
Border of data	When `'LineColor'` is set to a value other than `'none'`, the contourfm function no longer draws lines around the boundary of the data. The data boundary is not a contour, so it is not treated as such. The only lines drawn are true contours.	In previous releases, the function drew lines around the boundary of the data, in addition to the contour lines themselves.
Line colors	If you set `'LineColor'` for contourm or contour3m to `'auto'` or `'flat'`, the line colors come from the figure's colormap, as always. But, as of R2010b, if you change the figure's colormap after creating the contours, the line colors do not change.	In previous releases, if you changed the figure's colormap after creating the contours, the line colors changed. To change the line colors, use contourcmap.
Fill colors	When you call contourfm, the fill colors are derived from the figure's colormap, as in previous releases. As of R2010b, if you change the figure's colormap after calling contourfm, the fill colors are not affected.	To change the fill colors after plotting the filled contours, call contourcmap.
Globe map display	You can now use the globe map display with the contouring functions. The contour3m function warns, but if you are careful to scale your input data correctly relative to the radius of your reference sphere, you can still use it.	In previous releases, you could not use the globe map display with the contouring functions.

Now that `contourfm` produces correct results more consistently, it also takes somewhat longer to run.

clabelm No Longer Breaks Contour Lines
Compatibility Considerations: Yes

In previous releases, the clabelm function broke contour lines to display the contour level tag. The breaks failed to scale appropriately during zooming or when the figure size changed. Now, instead of breaking the contour line, the clabelm function sets the color of the background where the tag is inserted to the color of the ancestor axes or line.

Compatibility Considerations

If you want to display the contour labels without a background color, as in previous releases, use set to specify 'BackgroundColor','none' on the text object handle array returned by clabelm.

Changes in geoshow Behavior with 'DisplayType','contour'
Compatibility Considerations: Yes

Many aspects of the fix to Bug 192285 apply to geoshow contouring options as well as to contourm. When applied to a data grid with the DisplayType parameter set to 'contour', geoshow now contours the grid in the same way that contourm would, constructs a handle to the same sort of hggroup, and accepts the same set of optional parameters (as documented on the contourm reference page).

Compatibility Considerations

When used with 'DisplayType','contour' in previous releases, geoshow constructed a MATLAB contourgroup and returned its handle. In previous releases, you could set any contourgroup property via geoshow (possibly with unexpected results); you can now set only the parameters described in the contourm reference page—a useful, relevant, and validated subset.

Changes in geoshow Behavior with `'DisplayType','surface'`
Compatibility Considerations: Yes

When applied to a data grid with the `'DisplayType'` parameter set to `'surface'`, geoshow now sets the `'FaceColor'` property to `'interp'`, unless the `'CData'` property is also passed into the function. In that case, the `'FaceColor'` is set to `'texturemap'`.

Compatibility Considerations

When used with `'DisplayType'`, `'surface'`, in previous releases, geoshow set the `'FaceColor'` property to `'texturemap'`.

Changes in Behavior for the handlem Function
Compatibility Considerations: Yes

Changes in Finding Filled Contour Handles
The `'Cpatches'` option has been removed in R2010b and replaced by `'fillcontour'`.

Compatibility Considerations

In earlier versions of MATLAB, you could do the following:

```
load geoid
worldmap world
contourfm(geoid, geoidrefvec, 10)
h = handlem('Cpatches');
```

(The output h is an array of patch object handles.)

In R2010b, to achieve a comparable result, you can use either:

```
h = handlem('fillcontour');
```

or

```
h = handlem('contour');
```

(The output h is a handle to an hggroup.)

In cases where hggroups with both filled and unfilled contours exist, use the `'fillcontour'` syntax to return only handles to the hggroups with filled contours. If you use the handlem(`'contour'`) syntax, you will return handles to all hggroups containing contours generated by Mapping Toolbox functions.

Changes in Finding 3-D Contour Handles
The `'contour3d'` syntax has been removed in R2010b and replaced by `'contour'`.

Compatibility Considerations

In earlier versions of MATLAB, you could do the following:

```
load geoid
worldmap world
contour3m(geoid, geoidrefvec, 10)
h = handlem('contour3d');
```

(The output h is an array of patch object handles.)

In R2010b, to achieve a comparable result, you can use:

```
h = handlem('contour');
```

(The output h is a handle to an hggroup.)

Changes in Finding Contour Label Handles

The documentation in R2010a noted 'clabels' as the string to use in finding contour labels. This is incorrect and should be 'clabel'. The documentation has been changed to 'clabel' in R2010b.

Syntax Changes for contourcmap
Compatibility Considerations: Yes

In previous releases, the `contourcmap` function had the following syntax:

`contourcmap(cdelta,cmapstr)`

Now, `contourcmap` accepts the colormap string by itself:

`contourcmap(cmapstr)`

Or the colormap string with `cdelta`:

`contourcmap(cmapstr,cdelta)`

Notice that the position of the two input arguments has changed. The `cdelta` argument now appears after the colormap string.

In addition to this change in syntax, the `contourcmap` function exhibits some changes in behavior:

- When the axes contains Mapping Toolbox contour objects, `cdelta` is ignored and the resultant colormap contains the same number of colors as the original colormap. The `ColorAlignment` is set to `'center'` for contour lines and `'ends'` for filled contours and cannot be modified.

- After you have created a `contourcmap`, if you change the figure's colormap, the colorbar will change colors. The contourlines and the fill will not change colors. To work around this problem, do not set the figure colormap directly; set `contourcmap`.

- You can now set the `'Colorbar'` parameter to `'off'` to remove it from your map.

Compatibility Considerations

The original syntax, with the colormap string and `cdelta` in the reverse order, still works. In a future release, this undocumented syntax will be removed.

Change in Behavior for bufferm
Compatibility Considerations: Yes

If you specify `'out'` for the `direction` string when calling the `bufferm` function, the returned buffer zone will include all points outside the polygon within a specified distance of its edge. In previous releases, the returned buffer zone also included the points within the polygon.

Compatibility Considerations

If you have code that specifies `'out'` for the `direction` string, you now receive only the region outside the polygon but not the polygon itself. If you want to receive the union of the polygon and the buffer zone, as you did in previous releases, use the `polybool` function.

maptrims No Longer Trims Edge Rows and Columns

In previous releases, you could call `maptrims` with an input value for `latlim` or `lonlim` that corresponded to a parallel or meridian that ran precisely along a cell boundary. However, when you did so, the cells adjacent to that boundary would be trimmed off even if they fell completely within the requested limits. Now, if `latlim` or `lonlim` corresponds to a cell boundary, the output grid extends all the way to that limit. If a limiting parallel or meridian cuts through a column or row of input cells, the limit is truncated to avoid partial cells.

Change in Longitude Limits for WMSMapRequest and WMSLayer

The `WMSMapRequest` and `WMSLayer` objects now accept longitude limits from `[0 360]` or from `[-180 180]`. In previous releases, longitude limits had to be from `[-180 180]`.

polyxpoly Now Issues Warning when 'unique' Option Combined with Segment Indices

If you attempt to use the following syntax:

```
[xi,yi,ii] = polyxpoly(x1,y1,x2,y2,'unique')
```

polyxpoly issues a warning and ignores the 'unique' flag.

R2010a

Version: 3.1
New Features: Yes
Bug Fixes: No

WMS Database Modified
Compatibility Considerations: Yes

The WMS Database changes on a release-to-release basis, as some new servers are added and some unavailable servers are removed. A total of 199 servers (10.2% of the original number of servers) and 66,270 layers have been removed from the database. A total of 602 servers, with 207,269 layers, have been added.

Some notable new servers in the database are:

- The OnMars (`http://onmars.jpl.nasa.gov/wms.cgi`) and OnMoon (`http://onmoon.jpl.nasa.gov/wms.cgi`) servers from the Jet Propulsion Laboratory

- The MassGIS server (`http://giswebservices.massgis.state.ma.us/geoserver/wms`) from the Massachusetts Office of Geographic and Environmental Information

- The National Map Seamless servers (`http://ims.cr.usgs.gov`, `http://imsortho.cr.usgs.gov`, and `http://imselev.cr.usgs.gov`) from the U.S. Geological Survey

Compatibility Considerations

Some servers are no longer accessible and have been removed from the database. If you have code based on these servers, it will no longer run. To fix this problem, search the WMS Database for another comparable server.

Some examples from the R2009b documentation have been modified due to server inaccessibility. Some notable servers that have been removed are:

- Several of the servers from CubeWerx® (`http://demo.cubewerx.com/dem/cubeserver/cubeserv.cgi`)

- Two of the servers from the European Space Agency (ESA) (`http://mapdev.eo.esa.int/mapServer/mapServer` and `http://mapdev.esrin.esa.int/mapServer/mapServer`)

The etopo Function Now Supports the ETOPO1 and ETOPO2v2 Data Sets
Compatibility Considerations: Yes

Before R2010a, the `etopo` function supported ETOPO2–2001 (2-minute) and ETOPO5 (5-minute) data. Support has been added for ETOPO2v2c (2-minute) and ETOPO1c (1-minute) data. The ETOPO1 model, released in March 2009, is the most recent and contains the highest resolution data. For information on downloading the ETOPO data sets, see Technical Note 2101: Accessing Geospatial Data on the Internet for the Mapping Toolbox on the Mathworks Web site.

Compatibility Considerations

The `etopo` function still works with ETOPO2 and ETOPO5 data.

Now Possible to Retrieve Legend for WMS Map

The `Details` property of the `WMSLayer` class contains a `Style` field. A `LegendURL` structure has been added to this field. The information in the `LegendURL` structure, if provided by the server, enables you to retrieve a legend image for a specific WMS map.

Clipping Property Default Now Set to 'off'
Compatibility Considerations: Yes

Clipping is now 'off' by default for both map frames and graticule ("grid") lines. This is advantageous in certain display situations, but it also means that the frame and graticule may extend outside the axes limits (unlike the data plotted on the map), if you zoom in on a figure containing a map. You can use set to turn clipping back on, like this, for example:

```
set(handlem('frame'),'Clipping','on')
set(handlem('grid'),'Clipping','on')
```

Compatibility Considerations

In releases before R2010a, the default for the Clipping property was 'on'.

95

The shaperead and shapewrite Functions Now Support Non-ASCII Characters

You can now use the shaperead and shapewrite functions to import and export attributes with non-ASCII characters. To use this feature, you must set your character encoding scheme to match that used by the shapefile. For example, if your session is configured to support US_ASCII character encoding and you want to import a shapefile with Japanese characters, you must first change your configuration to support Shift_JIS.

Display Range Increased for eqdazim and eqaazim Projections
Compatibility Considerations: Yes

In previous releases, the Equidistant Azimuthal (`eqdazim`) and Lambert Azimuthal Equal-Area (`eqaazim`) projections did not allow projection or display of data points farther than 160 degrees from the projection origin. Now you can set the projection radius for either of these projections to up to 179.5 degrees.

Use the `FLatLimit` property to control this setting. For example, to choose the largest possible value, pass the following property name-value pair to `axesm` or `setm`:

```
'FLatLimit',[-Inf 179.5]
```

Compatibility Considerations

The default value of the projection radius remains 160 degrees, so you do not need to update any code that relies on the default value.

Before R2010a, you could use the following line of code to initialize the equatorial aspect of an azimuthal projection:

```
axesm(projectionName,'MapLonLimit',westernAndEasternLimits)
```

Now, if you enter this line of code with `eqdazim` or `eqaazim` in place of *projectionName*, you will receive a warning message, and your `'MapLonLimit'` input will be ignored. To use the `'MapLonLimit'` property to initialize the equatorial aspect of an azimuthal projection, you should enter the following:

```
axesm(projectionName,'FLatLimit',[], ...
   'MapLonLimit',westernAndEasternLimits)
```

See Example 7: Equatorial Azimuthal Projection in the Axes for Drawing Maps section in the User's Guide for an illustration of this usage.

The GUIs mlayers and mobjects No Longer Support EraseMode
Compatibility Considerations: Yes

The EraseMode property, represented by the **Emode** button, has been removed from the mlayers and mobjects GUIs.

Compatibility Considerations

Before R2010a, the mlayers and mobjects GUIs had an EraseMode option, which made it possible to set the erase mode on a particular map layer or object. This property controlled the technique MATLAB used to draw and erase hggroup child objects.

scatterm Now Returns a Handle to an hggroup Object
Compatibility Considerations: Yes

The function scatterm in the syntax

```
h = scatterm(...)
```

now returns a handle to an hggroup.

Compatibility Considerations

In previous releases, scatterm returned a vector of patch handles.

mdistort Now Returns a Handle to a contourgroup Object
Compatibility Considerations: Yes

The output of the syntax

```
h = mdistort(...)
```

is now a scalar handle to a contourgroup object containing the contours and text.

Compatibility Considerations

In previous releases, mdistort returned handles to the line and text objects and used the syntax

```
[h,ht] = mdistort(...)
```

with two output arguments. The second output of mdistort is now redundant because these handles will be available as children of h.

polybool No Longer Errors when Given Empty Input Vertex Arrays
Compatibility Considerations: Yes

When one or both pairs of input vertex arrays is empty, the `polybool` function now returns either empty values or the values of the non-empty input pair, depending on the requested operation.

For example, consider the following case:

```
[x,y] = polybool('union', [0 0 1 1], [0 1 1 0], [], [])
```

The `polybool` function returns the first input pair: [0 0 1 1], [0 1 1 0]. Now consider the `'intersection'` operation:

```
[x,y] = polybool('intersection', [0 0 1 1], [0 1 1 0], [], [])
```

The `polybool` function returns [], [].

Compatibility Considerations

In previous releases, if one or both pairs of input vertex arrays were empty, the `polybool` function would issue an error.

Functions Being Removed
Compatibility Considerations: Yes

Function Name	What Happens When You Use the Function?	Use This Instead	Compatibility Considerations
etopo5	Errors	etopo	Replace all existing instances of etopo5 with etopo.

R2009b

Version: 3.0
New Features: Yes
Bug Fixes: No

New Features for Creating Web Map Service Maps

New functions and classes now make it possible to interact with Web Map Service (WMS) servers and render WMS maps. Use the new features to search a built-in database of pre-qualified WMS servers and layers. Retrieve customized geographic data sets and related metadata from WMS servers. The new classes encapsulate WMS servers, data layers, metadata, and map requests. See the Creating Web Map Service Maps chapter in the User's Guide and related Class Reference for more information.

New makerefmat Syntax for Constructing Referencing Matrices

A new parameter name-value pair syntax makes it easier to construct referencing matrices with `makerefmat`. You can use the new syntax for an image or raster grid that is referenced to and aligned with a geographic coordinate system but not for one that is referenced to a 2-D map coordinate system. Use parameters to set the number of rows (M) and columns (N) of the raster or image to be used with the referencing matrix; the latitude and longitude limits of the geographic quadrangle bounding the georeferenced raster; and the edges from which row and column indexing start, designating, for example, columns that run either south-to-north or north-to-south.

Some Functions Now Accept Referencing Matrices as Input

The functions below now accept referencing matrices as input, and some of them (maptrims, resizem, and vec2mtx) also generate referencing matrices as output. The functions that generate referencing matrices as output do so only in cases where referencing matrices are used as input. If referencing vectors are used as input, referencing vectors are also generated as output. Note that the functions in this table work exclusively with data grids or images that are referenced to geographic (latitude or longitude) coordinates.

areamat	imbedm	maptrims	setltln	usamap
filterm	limitm	meshgrat	setpostn	worldmap
findm	los2	meshlsrm	resizem	
getseeds	ltln2val	meshm	vec2mtx	
gradientm	mapprofile	neworig	viewshed	

New Angle Conversion Functions
Compatibility Considerations: Yes

Two new functions, radtodeg and degtorad, replace older functions rad2deg and deg2rad. They are functionally identical.

Compatibility Considerations

The older functions, rad2deg and deg2rad, will continue to work, but when writing new code the newer versions are recommended.

Expanded Support for GSHHS Global Coastline Data

The gshhs function now supports Versions 1.4 and later of the Global
Self-Consistent, Hierarchical, High-Resolution Shoreline Database (GSHHS).

New Behavior for polymerge when Three or More Line Segments Have Common End Point
Compatibility Considerations: Yes

The behavior of the `polymerge` function has changed in cases of three or more distinct parts with a common end point. In such cases, the choice of which parts to merge is ambiguous; therefore, none of the corresponding parts are connected at that common point.

Compatibility Considerations

In previous releases, if three or more parts shared a common end point, the `polymerge` function attempted to merge them. The result, however, was unspecified and sometimes obviously wrong.

Automatic Conversion of Latitude Limits to Ascending Order
Compatibility Considerations: Yes

The functions axesm and setm require that the latitude limits in the 'MapLatLimit' property be provided in ascending order. If you enter the limits in descending order, these functions will now automatically convert the limits to ascending order, and return a warning message notifying you of this change.

Compatibility Considerations

In previous releases, if you entered the latitude limits of the 'MapLatLimit' property in descending order when using axesm or setm, you could end up with a map axes that was internally inconsistent, possibly resulting in unexpected errors during subsequent operations. This is no longer the case.

Second Input Argument of roundn No Longer Supports Complex Numbers, Non-integers, or Default Values
Compatibility Considerations: Yes

`roundn` no longer accepts certain types of input for the second input argument, `N`, which is supposed to be a real, integer-valued exponent. Now, if you use a complex number or non-integer as the second input to `roundn`, you will receive an error; and if you omit `N`, you will receive a warning. You will also receive an error if you call `roundn` with a second output argument to capture error message strings.

Compatibility Considerations

In previous releases, if you used a complex number or non-integer as the second input to `roundn`, this number would be converted into a real integer. If you called `roundn` with a second output argument to capture error message strings, you would receive an obsolete syntax warning.

In R2009b, `-2` is still the default value for the second input argument. This default is being phased out, however, and in the future you will receive an error if you fail to supply the second input argument. If there are any instances in your code with the usage `roundn(x)`, you should replace them with `roundn(x,-2)`.

The two-output syntax option was previously deprecated and has resulted in a warning in the past several releases.

Functions Removed

Functions Being Removed in a Future Release

Name	Stage	Compatibility Considerations
colorui	Still runs	Replace all existing instances of colorui with uisetcolor.
eastof	Warns	If you are using degrees, replace eastof(lon, meridian, 'degrees') with meridian + mod(lon - meridian, 360) and if you are using radians, replace eastof(lon, meridian, 'radians') with meridian + mod(lon - meridian, 2*pi)
imagem	Errors	Replace all existing instances of imagem with grid2image.
smoothlong	Warns	Use unwrapMultipart instead. This function requires its input to be in radians. When working in degrees, use radtodeg(unwrapMultipart(degtorad(lon)))
tgrline	Still runs	More recent Tiger/Line® data sets are available in shapefile format and can be imported using shaperead.

Name	Stage	Compatibility Considerations
unitstr	Warns	The syntax `str = unitstr(str,'times')` has already been removed.

westof	Warns	If you are using degrees, replace
		
		`westof(lon, meridian, 'degrees')`
		
		with
		
		`meridian - mod(meridian - lon, 360)`
		
		and if you are using radians, replace
		
		`westof(lon, meridian, 'radians')`
		
		with
		
		`meridian - mod(meridian - lon, 2*pi)`

Functions Removed in R2009b

```
deg2dm      hms2hm      hr2sec      sec2hms     cmapui
deg2dms     hms2hr      mat2dms     sec2hr      tigermif
dms2deg     hms2mat     mat2hms     time2str    tigerp
dms2dm      hms2sec     rad2dm      timedim
dms2mat     hr2hm       rad2dms     contorm
dms2rad     hr2hms      sec2hm      contor3m
```

The functions above have been completely removed from the toolbox and error if used.

R2009a

Version: 2.7.2
New Features: Yes
Bug Fixes: No

geoshow and mapshow Now Construct Ordinary Patch Objects
Compatibility Considerations: Yes

When displaying polygons, instead of constructing graphics objects whose classes derive from patch, geoshow and mapshow now construct ordinary patch objects.

Compatibility Considerations

This change has no effect on the display, but it does have some effect on your ability to load and save figures. If you have a figure containing a polygon displayed by geoshow or mapshow that was saved in R2008b or earlier, you will not be able to load it in R2009a.

You may also notice the change if you call get on a handle. The older (derived) class included several extra properties used only for internal bookkeeping. In R2009a, these properties have been removed, and the output of get looks different in terms of both layout and property order. (It now looks the same as for any ordinary patch.)

R2008b

Version: 2.7.1
New Features: Yes
Bug Fixes: No

Using the Map Axes Map Limit Properties with axesm, setm, and defaultm

Changes and enhancements have been made to axesm, setm, and defaultm with respect to map axes properties that affect the fundamental display geometry:

- MapProjection

- Zone

- Origin

- FLatLimit

- FLonLimit

- MapLatLimit

- MapLonLimit

The changes result in the following improvements:

- The use of the map limit properties to set up a map axes is more intuitive.

- The way in which defaultm resolves possible inconsistencies between these properties is now consistent with the behavior of axesm and setm.

- The map limit properties (MapLatLimit and MapLonLimit), the frame limit properties (FLatLimit and FLonLimit), and the Origin, MapProjection, and Zone properties interact in a more clear and predictable fashion.

For more information, see the section Using the Map Limit Properties in the *Mapping Toolbox User's Guide* and bug report 319891 on the MathWorks Web site.

Changing Projection Type of an Existing Map Axes with setm
Compatibility Considerations: Yes

In previous releases, calling the `setm` function to change the `MapProjection` property of a map axes, especially when switching between an azimuthal and non-azimuthal projection (e.g., a conic or cylindrical projection), often resulted in the following types of problems:

- The modified map axes might cover a different part of the Earth.

- The map frame and graticule might fail to update properly.

- Map limit properties changed at the same time as the projection might not have the proper effect.

The `setm` function now more effectively resets the projection, clearing out settings that were specific to the earlier projection, updating the map frame and graticule, and staying in the same general part of the world (even when switching between azimuthal and non-azimuthal projections).

Compatibility Considerations

You may need to change the way in which you reset various map axes properties, such as `Origin`, `FLatLimit`, and `FLonLimit` after changing projections, as discussed in the section Switching Between Projections in the Mapping Toolbox User's Guide. In many cases it will no longer be necessary to reset as many properties.

Other Bug Fixes
Compatibility Considerations: Yes

Compatibility Considerations

- The default FLatLimit for lambert and lambertstd has been changed to [-45 45]. In previous releases, axesm produced huge map frames, due to the FLatLimit default of [-90 90].

- The function gridm now returns handles to the line objects used to display the parallels and meridians. In previous releases, a call to gridm using linespec or property name/property value syntaxes returned empty.

- The function geotiff2mstruct no longer sets the maplatlimit and maplonlimit fields.

- A reference ellipsoid set to a non-default value (via the geoid property) no longer reverts to the default when the UTM zone is reset. For more information, see bug report 459353 on the MathWorks Web site.

- The daspectm function now works for azimuthal projections and units of radians.

coast.MAT Data File Revised

Portions of the global coastline latitude-longitude vectors in the `coast.MAT` data file have been revised to ensure proper polygon topology. The data edits comprise the following:

- Replacing or removing various "bow-tie" and degenerate linear (non-polygonal) island features.

- Opening a "pinched" section in the middle of Lake Balkhash in Central Asia.

- Merging the eastern and western sections of Wrangel Island near the Bering Strait (cut by the 180-degree meridian) into a single polygon with longitudes ranging from slightly less than 180 to slightly greater than 180.

- Eight additional edits to pull apart landmasses with points of contact and remove coastal "spikes."

Map Limit Syntaxes Removed
Compatibility Considerations: Yes

The following syntaxes are obsolete. An error occurs if you use them.

- `pcolorm(Z)`

- `pcolorm(Z,gratsize)`

- `surfacem(Z)`

- `surfacem(Z,gratsize)`

- `surflm(Z)`

- `surflm(Z,s)`

- `surfm(Z)`

- `surfm(Z,gratsize)`

These syntaxes displayed a data grid with geographic limits that matched the map latitude and longitude limits in the current map axes. Using the old syntaxes correctly involved knowing the latitude and longitude limits of your data and matching them to the values listed under `maplatlimit` and `maplonlimit` in the map axes properties. We have replaced these syntaxes with a more direct approach that requires you to enter the latitude and longitude limits for the data grid.

Compatibility Considerations

The table below suggests alternative code to replace the obsolete syntaxes. In the following table, `Z` is a regular data grid (a 2-D array of class `double`) and `gratsize` is a two-element vector specifying the size of the graticule on which `Z` displays:

```
gratsize = [number_of_parallels number_of_meridians]
```

`h` is a handle to the surface that is displayed. And `latlim` and `lonlim` are the geographic limits of the data grid (in degrees):

```
latlim = [southern_limit northern_limit]
```

lonlim = [western_limit eastern_limit]

Original Syntax	Replacement Syntax

`h = pcolorm(Z)`
 	
constructs a surface using the regular data grid Z and a graticule mesh (using `meshgrat`) with size equal to `size(Z)` and with geographic limits that match the map latitude and longitude limits in the current map axes.
	`[lat,lon] = meshgrat(latlim,lonlim,size(Z));`
	
	`h = pcolorm(lat,lon,Z)`

`h = pcolorm(Z,gratsize)`
 	
uses a graticule mesh with size equal to `gratsize`.
	`[lat,lon] = meshgrat(latlim,lonlim,gratsize);`
	
	`h = pcolorm(lat,lon,Z)`

`h = surfacem(Z)` constructs a surface using the regular data grid Z and a graticule mesh (using `meshgrat`) of size 50-by-100. The geographic limits match the map latitude and longitude limits in the current map axes.	`h = surfacem(latlim,lonlim,Z)`

`h = surfacem(Z,gratsize)`
 	
uses a graticule mesh with size equal to `gratsize`.
	`[lat,lon] = meshgrat(latlim,lonlim,gratsize);`
	
	`h = surfacem(lat,lon,Z)`

`h = surflm(Z)` constructs a surface using the regular data grid Z and a graticule mesh (using `meshgrat`) with size equal to `size(Z)` and with geographic limits that match the map latitude and longitude limits in the current map axes. It is displayed with a default light source.	`h = surflm(latlim,lonlim,Z)`

Original Syntax	Replacement Syntax
`h = surflm(Z,s)` specifies the direction of the light source. `s` is a two- or three-element vector that specifies the direction from the surface map to the light source as defined in the documentation for `surfl`.	`h = surflm(latlim,lonlim,Z,s)`
`h = surfm(Z)` constructs a surface using the regular data grid `Z` and a graticule mesh (using `meshgrat`) with size equal to `size(Z)` and with geographic limits that match the map latitude and longitude limits in the current map axes.	`h = surfm(latlim,lonlim,Z)`

`h = surfm(Z,gratsize)`
 	
uses a graticule mesh with size equal to `gratsize`.
	`[lat,lon] = meshgrat(latlim,lonlim,gratsize);`
	
	`h = surfm(lat,lon,Z)`

R2008a

Version: 2.7
New Features: Yes
Bug Fixes: No

Functions for Working with Geographic Quadrangles

A geographic quadrangle is an area on the surface of a sphere or ellipsoid bounded on the east and west by a pair of meridians and on the north and south by a pair of parallels. In many ways, such an object is similar to a bounding rectangle in the plane, but they can be difficult to work with because of the way longitudes wrap around and the way meridians converge at the poles. For example,

- The western longitude limit can have a larger numerical value than the eastern longitude limit.

- If one of the bounding latitudes is +90 or -90 degrees, the quadrangle has three sides rather than four.

- As noted below, the intersection of two geographic quadrangles might possibly comprise two separate parts—with the eastern end of the first quadrangle intersecting the western end of the second quadrangle, and vice versa.

Mapping Toolbox software typically represents a geographic quadrangle in terms of its latitude and longitude limits, stored in 1-by-2 vectors having the forms

```
latlim = [southern_limit northern_limit]
lonlim = [western_limit eastern_limit]
```

Vectors like these have been used in various Mapping Toolbox functions since its inception, and can appear in the input or output argument lists of over dozen functions.

In R2008a, three new functions let you query, intersect, and display geographic quadrangles, and account for subtleties such as those described above:

- `ingeoquad` — Returns `true` for points inside or on latitude-longitude quadrangle

- `intersectgeoquad` — Returns intersection(s) of two latitude-longitude quadrangles

- `outlinegeoquad` — Returns sampled polygon vertices for a latitude-longitude geographic quadrangle

Use `ingeoquad`, for example, to check whether a geographic point is located within the area covered by a regular data grid, given the latitude and longitude limits computed by `limitm`.

Use `intersectgeoquad` to compute overlap, if any, between two quadrangles. Interestingly, three general results are possible: no intersection, an intersection that is itself a geographic quadrangle, and an intersection the comprises two distinct geographic quadrangles. (The intersection can have two parts if the input quadrangles wrap around in longitude to overlap on both their eastern and western sides. This case, of course, is not possible for bounding boxes in the plane.)

Use `outlinegeoquad` to generate a pair of latitude and longitude coordinate vectors that define a polygon that traces the outline of a geographic quadrangle. This can be useful for displaying the quadrangle graphically using `geoshow`, for example, especially on a projection where the meridians and/or parallels do not project to straight lines, because in addition to connecting the four corners `outlinegeoquad` lets you interpolate additional vertices along parallels, meridians, or both.

Fixes and Improvements to Function avhrrgoode
Compatibility Considerations: Yes

Function avhrrgoode has been rewritten to improve its efficiency and to
remove a number of problems and limitations:

- Fixed a spatial referencing problem when a nonglobal region has been
  specified which caused locations to be offset by half a pixel.

- The function no longer returns incorrect NaN coordinate values at the
  equator when given certain latitude limits that cross the equator.

- The function no longer errors when attempting to read a file name with
  certain legal latitude and longitude limits.

- The new version executes at least five times faster.

Compatibility Considerations

- The nonfunctional syntaxes avhrrgoode and avhrrgoode(region) have
  been removed from the documentation.

- The function now returns empty when the user-supplied limits are outside
  data limits.

- The function no longer permits longitude limits to be specified outside
  the interval [-180 180].

- Parameters other than region and filename can be specified as empty
  to use their default values.

- In versions prior to R2008a, when reading from the global data set and
  a smaller region data set, the size of the outputs differed by one column
  from each other when given identical latitude and longitude limits. Now
  the sizes are the same.

Improved Accuracy for the limitm and setpostn Functions
Compatibility Considerations: Yes

In previous releases, after calculating the latitude and longitude limits of the geographic quadrangle bounding a regular data grid, function `limitm` arbitrarily rounded those limits to the nearest one millionth of a degree (equivalent to about 10 cm in latitude or equatorial longitude). Although it is small, this rounding operation in effect applied an arbitrary shift to points on or very near the edge of the grid. The direction of the shift and its magnitude were arbitrary because rounding can either increase or decrease a value. In any given case, the shift depended on the specific referencing vector and the number of columns and rows in the data grid. This behavior unnecessarily degraded the numerical accuracy of `limitm` and those functions which depend on it, and it has now been removed. For more information, see bug report 420038 on the MathWorks Web site.

In the `setpostn` function , an identical rounding step has been removed. Additional changes eliminate a problem for certain input points near boundaries between grid cells that caused row and column subscripts returned by `setpostn` to be off by 1. For points near the northern and eastern edges of the data grid—but still within the grid—returned subscript values could exceed the corresponding grid size. For more information, see bug report 173338 on the MathWorks Web site.

Compatibility Considerations

These corrections can cause subtle changes in the behavior of other functions that work with regular data grids referenced to latitude-longitude, for example, `imbedm`.

If your referencing vector contains approximations to rational numbers that do not have an exact a 64-bit floating point representation (e.g., for cells that are 1.5 degrees wide, `refvec(1)` is 0.666666...), you may still find that certain points that are extremely close to a grid cell boundary cross into a neighboring cell just across the boundary. Such numerical ambiguity is inevitable given how the information in a referencing vector is encoded. Although it cannot be eliminated within `setpostn`, the inexactness only affects points that fall

within a few factors of `eps` (very much less than a millionth of a degree) away from a given cell boundary.

New Point Location Demo Data for Tsunami Events

The Mapping Toolbox demo data in the $MATLABROOT/toolbox/map/mapdata directory now includes a global tsunami data set in shapefile format with 'Point' geometry. The data set comprises four files:

```
tsunamis.dbf
tsunamis.shp
tsunamis.shx
tsunamis.txt
```

tsunamis.txt is not part of the shapefile set. It is a text file documenting the data set.

The data includes tidal wave events for which the maximum water height was at least one meter, ranging for the years 1950 to 2006, inclusive. The Global Tsunami Database, U.S. National Geospatial Data Center (NGDC), National Oceanic and Atmospheric Administration (NOAA), available at http://www.ngdc.noaa.gov/seg/hazard/tsu.shtml, is the source of the data. All the files consist of U.S. Government information that is in the public domain and is not subject to copyright protection.

The approximate location of each event is a single point in geodetic coordinates (latitude-longitude) with an unspecified datum. The .dbf file contains 18 separate text or numeric attributes for most events, including wave height, causes and seismic magnitudes, and location and country names.

The shapefiles were created at MathWorks from querying the online source data, importing the results into the MATLAB workspace, and exporting them using the Mapping Toolbox shapewrite function. For more information, type

```
edit tsunamis.txt
```

at the MATLAB prompt.

Better Trimming Benefits fillm Function

The changes described in the Versin 2.6 (R2007b) release note "Improvements to Data Trimming in patchm and patchesm" on page 149 resulting from improved polygon trimming also apply to the `fillm` function.

Restored units Options for Function angl2str

The `angl2str` function once again can format strings for angles in degrees-minutes (DM) and degrees-minutes-second (DMS) notations. These options were removed in Version 2.6 (R2007b), and have now been restored. In addition to the `'degrees'` and `'radians'` *units* options, you can now obtain DM- and DMS-formatted strings by specifying

- `'degrees2dm'` — for degrees-decimal minutes formatting
- `'degrees2dms'` — for degrees-minutes-decimal seconds formatting

To use these options, input angles must be in degrees. That is, `angl2str` uses the string *units* to indicate both the units in which the `angle` argument is provided *and* to control the output format.

This change restores the behavior of `angl2str` prior to Version 2.6 in a slightly different form. Before V. 2.6, the DM and DMS options were specified by a *units* strings of `'dm'` and `'dms'`, respectively. The new strings that replace them signify that the functions `degrees2dm` and `degrees2dms`, introduced in Version 2.5 (R2007a), perform the conversions of inputs given in degrees to DM and DMS notation.

New Longitude-Wrapping Option in the closePolygonParts Utility

The `closePolygonParts` function now accepts an optional third argument, `angleunits`, that must be string-valued and can be either `'degrees'` or `'radians'`. If you include this argument with a value appropriate for the first two (`lat, lon`) arguments, `closePolygonParts` can correctly account for longitude wrapping. For example, a polygon that begins at a given latitude with a longitude of -180 degrees, and ends at the same latitude with a longitude of 180 degrees is regarded as closed and an additional vertex is not added.

Changes to Terminology for Geographic Data Structures

From Version 2.0 onward, the Mapping Toolbox documentation has referred to "version 1 geographic data structures" and "version 2 geographic data structures," using the terms "geostruct1" and "geostruct2" respectively as shorthand for them. To reflect current usage, starting with this version of the toolbox, these terms are obsolete; new terms and distinctions have been defined to help clarify what these structures are and can be used for:

- Geographic data structure arrays, introduced in Version 2.0, contain vector features and are called either

  - *Geostructs*, if they contain geographic coordinates (latitudes and longitudes)

  - *Mapstructs*, if they contain projected map/planar coordinates (*x* and *y*)

- *Display structure arrays*, dating from Version 1, also used to be called geographic data structures, and can contain either vector features or raster geodata.

Due to their greater generality, geostructs and mapstructs are the preferred form in which to represent vector features in the toolbox. The preferred way to package raster geodata is with regular or geolocated data grids (2-D numeric arrays accompanied by referencing matrices or vectors). There are only a few Mapping Toolbox functions that can still generate display structures (by importing data from external file formats):

- `dcwdata` — Returns line/patch display structures

- `dcwgaz` — Returns line/patch display structures

- `demdataui` — Returns "regular"—as in regular data grid, that is—display structures

- `mlayers` — GUI to control plotting of display structure elements

- `tgrline` — Returns line/patch display structures

- `vmap0data` — Returns line/patch display structures

- `vmap0ui` — GUI for selecting data from Vector Map Level 0

Even fewer functions accept display structures as inputs:

- displaym — Displays elements of a display structure

- extractm — Extracts lat-lon coordinates from line/patch display structure

In addition to displaym and extractm, the updategeostruct function converts a line or patch display structure to a geostruct.

For more information, see Mapping Toolbox Geographic Data Structures.

Identifiers Provided for all Warnings

All warnings issued from within Mapping Toolbox functions now include identifiers, enabling you to suppress them at your own discretion. Previously, this was possible for only certain warnings, but with the addition of new identifiers in over two dozen functions in R2008a, all warnings are now covered. For example, you can turn off the warning that `setpostn` issues when given a latitude-longitude position outside the limits of the specified data grid. In this case, the warning identifier is

```
'map:setpostn:pointOutsideLimits'
```

You can suppress it using the following statement:

```
warnstate = warning('off','map:setpostn:pointOutsideLimits');
```

Then, after making your call to `setpostn`, you can restore the original warning state with

```
warning(warnstate);
```

See the MATLAB `warning` function reference page for the for more information on turning warnings off and on and managing the warning state.

Documentation for Functions tigermif and tigerp Removed
Compatibility Considerations: Yes

The reference pages for following functions, which themselves were removed in R2007b, have been removed from the Mapping Toolbox User's Guide:

- `tigerp` — Read TIGER *p* and *pa* thinned boundary files (ArcInfo format)

- `tigermif` — Read the TIGER MIF thinned boundary file (MapInfo format)

Compatibility Considerations

See the R2007b release note "Functions tigermif and tigerp Are Obsolete and Error if Used" on page 163 for alternatives to `tigermif` and `tigerp`.

Removed Syntaxes that Returned Error Messages in Optional Argument
Compatibility Considerations: Yes

In earlier versions, the following Mapping Toolbox functions supported syntaxes that included an optional output argument called `msg`. If this output argument was included in a call to one of these functions, and certain error conditions were encountered while executing the function, then instead of issuing an error, the function would return the corresponding error message in `msg`. The following functions are affected:

- `axesm`
- `defaultm`
- `displaym`
- `gcm`
- `handlem`
- `lightm`
- `linem`
- `maps`
- `meshm`
- `namem`
- `patchesm`
- `roundn`
- `surfacem`
- `surflsrm`
- `textm`
- `unitstr`
- `utmzone`
- `utmzoneui`

For example, even with no map axes present, the command

```
[mstruct, msg] = gcm
```

returned without error in R2007b and earlier, but placed an error message string in `msg`.

These syntaxes have been disabled in R2008a. If you try to use them, a warning is issued. The warning may be followed by an error, depending on whether or not an error condition is encountered within the function. For example, if a map axes is present, the command above results in

```
Warning: Function GCM no longer returns error message strings in
output argument MSG. Instead any errors are thrown where they occur.
You should remove the last output argument (MSG) from your call to
GCM in order to avoid this warning. If you want to handle errors
yourself, call GCM in a try-catch block.
> In mapdisp/private/warnObsoleteMSGSyntax at 6
  In gcm at 20
```

If there is no map axes, it results in

```
Warning: Function GCM no longer returns error message strings in
output argument MSG. Instead any errors are thrown where they occur.
You should remove the last output argument (MSG) from your call to
GCM in order to avoid this warning. If you want to handle errors
yourself, call GCM in a try-catch block.
> In mapdisp/private/warnObsoleteMSGSyntax at 6
  In gcm at 20
??? Error using ==> gcm>checkaxes at 41
No axes in current figure.
Select a figure with map axes or use AXESM to define one.

Error in ==> gcm at 24
h = checkaxes(varargin{:});
```

Compatibility Considerations

As suggested by this warning, if you have any scripts or functions of your own that depend on the old syntax, you should remove the `msg` argument and place the function call in a `try-catch` block instead.

R2007b

Version: 2.6
New Features: Yes
Bug Fixes: No

Exporting Vector Geodata to Earth Browsers

kmlwrite is a new function for exporting vector point data to a file in KML format. KML stands for Keyhole Markup Language; it is an XML dialect used to structure geographic data for display in an Earth browser, such as Google Earth™, Google Maps™, and Maps for Google Mobile™. KML has a hierarchical structure of nested elements and attributes. kmlwrite has a simple API that lets Mapping Toolbox users write vector data to a KML file in order to subsequently display the data onto an Earth browser.

When used with Google Earth, files output from kmlwrite can be seen immediately in Google Earth, if that application is available to the user. If the files are uploaded to a publicly accessible Web server, they can be viewed by anyone on the Internet via Google Maps or other Web sites and browser utilities that can read and display KML files. Google Maps and Google Maps for mobile do not support the range of KML markup that Google Earth supports (for example, placemark locations must be specified to them as coordinates, not as addresses). See the Google KML documentation at http://code.google.com/apis/kml/documentation/mapsSupport.html for more information.

kmlwrite accepts latitude and longitude point vectors, passed either in geostructs or as column arrays. It also accepts addresses, which can be as general as a country's name or as specific as a street address. When geostructs are the input, the attribute data in the geostruct can be formatted as HTML tables and included in the KML output. When latitude-longitude arrays are input, you can pass attributes to kmlwrite with strings. When addresses are the input, geostructs are not used.

To customize placemarks, you can control formatting of geostruct attributes in the KML file with an attribute specification, a struct used to format them (for example, to add units to length attributes or to control the number of decimal places for numeric values). A new support function, makeattribspec lets you change the names used as labels in placemarks (geostruct field names are used by default), omit fields from placemarks, and add HTML markup to the attributes displayed in placemark tables.

See Exporting Vector Geodata in the Mapping Toolbox User's Guide and the mapexkmlexport demo, "Exporting Vector Point Data to KML" for more information.

Improved Conversion Between Angle Units

The `angledim` function has been replaced by four, more specific, functions: `fromRadians`, `fromDegrees`, `toRadians`, and `toDegrees` (described below in "Four New Angle-Unit Conversion Functions" on page 144). However, `angledim` has been retained in Version 2.6 for backward compatibility. The functions `degtorad`, `radtodeg`, and `unitsratio` provide additional alternatives.

Because it must resolve both the input and output units, `angledim` is excessive for most applications. It works only for class `double` and it quietly discards the imaginary part of any complex input. You can use any of several more efficient alternatives:

If you are working from the command line, you can often replace `angledim` with `degtorad` or `radtodeg`. If you are converting angle units within a script or function and you know both the *from* and *to* unit names at the time of coding, then you can also replace `angledim` with `degtorad` or `radtodeg`. If you know either *from* or *to* at the time of coding, then you can use `fromRadians`, `fromDegrees`, `toRadians`, or `toDegrees`. Apply one of the following transformations to your code:

- `angledim(angleIn,'radians',`*to*`)` ⇒ `fromRadians(`*to*`,angleIin)`

- `angledim(angleIn,'degrees',`*to*`)` ⇒ `fromDegrees(`*to*`,angleIin)`

- `angledim(angleIn,`*from*`,'radians')` ⇒ `toRadians(`*from*`,angleIn)`

- `angledim(angleIn,`*from*`,'degrees')` ⇒ `toDegrees(`*from*`,angleIn)`

Also note that the functions in the `fromRadians` family can convert multiple variables in a single function call. For example, you can replace this code

```
angle1 = angledim(angle1InRadians,'radians',to);
angle2 = angledim(angle2InRadians,'radians',to);
```

with

```
[angle1,angle2] =
fromRadians(to,angle1InRadians,angle2InRadians);
```

If you do not know either *from* or *to* at the time of coding, then you can call `unitsratio` to obtain the correct conversion factor, then multiply the values of one or more variables. For example, you can replace:

```
angle1Out = angledim(angle1In, from, to);
angle2Out = angledim(angle2In, from, to);
```

with

```
 r = unitsratio(to, from);
 angle1Out = r * angle1In;
 angle2Out = r * angle2In;
```

Four New Angle-Unit Conversion Functions

The following functions have been added for efficient conversion of angle units (degrees or radians) when either the target or destination units (but not both) are unknown before run time.

- `toDegrees` — Convert angles to degrees

- `toRadians` — Convert angles to radians

- `fromDegrees` — Convert angles from degrees

- `fromRadians` — Convert angles from radians

If the output units match the inputs units, as in `toDegrees(units, angle1, angle2,...)`, where `units` turns out to equal `'degrees'`, then the input angles are simply copied to the output angles.

Use these functions in place of `angledim`. The new functions are more efficient, especially when the value of either the `from` or `to` argument of `angledim` is known in advance and the value of the other angle-unit argument is not.

Improvements in Handling Length Units
Compatibility Considerations: Yes

Alternatives to the distdim Function

There are now more efficient ways to convert length and distance units than the `distdim` function. In place of `distdim`, you can use `unitsratio` to compute multiplicative factors to apply when converting between different units of distances and angles, which you can use in subsequent calculations. For other alternatives, see Replacing distdim in the `distdim` reference page for details.

The unitstr function Is Obsolete

The `unitstr` function, which validates names and abbreviations for units of distance, angle, and time, is obsolete and will be removed in a future release. The syntax `str = unitstr(str,'times')` has already been removed. Instead, see the documentation for `unitsratio` for a list of valid unit strings.

Compatibility Considerations

There is no replacement for `unitstr`, but `unitsratio` recognizes all the unit strings known to the toolbox.

Interpretation of "Miles" Units has Changed

As of R2007b, the following functions interpret distance units specified as `'miles'` as *International Miles*, not *Statute Miles*:

- almanac
- daspectm
- elevation
- mapprofile
- paperscale
- scaleruler

Compatibility Considerations

This will not materially affect the accuracy of results in most cases; the lengths of the two types of miles only differ by about two parts per million (three millimeters). The `distdim` function's interpretation of miles has not changed. However, there are better alternatives to it; see the release note "Alternatives to the distdim Function" on page 145.

New Angle Wrapping Functions

Four new low-level functions have been added that force longitudes, azimuths, or phase angles to span intervals of [0 360] or [-180 180] degrees or [0 2*pi] or [-pi pi] radians.

- `wrapTo180` — Wrap angle in degrees to [-180 180]

- `wrapTo360` — Wrap angle in degrees to [0 360]

- `wrapToPi` — Wrap angle in radians to [-pi pi]

- `wrapTo2Pi` — Wrap angle in radians to [0 2*pi]

The first two functions work in degrees, the next two in radians. None of them perform argument checking.

You can use the new wrapping and functions in place of `npi2pi` and `zero22pi` for greater efficiency. The older functions will eventually be removed from the toolbox.

New Function to Unwrap Sequences of Angles

The new `unwrapMultipart` function unwraps vectors of angles similarly to the MATLAB function `unwrap`, except that it handles vectors that include NaN separators, unwrapping each section separately. Use it to remove discontinuities from vectors of longitudes, azimuths, or phase angles that contain NaN-delimited sequences and as a replacement for the obsolete function `smoothlong`.

Improvements to Data Trimming in patchm and patchesm
Compatibility Considerations: Yes

The `patchm` and `patchesm` functions now completely trim away polygons and parts of polygons that fall outside your current map limits. This improvement also affects `fillm`, which calls `patchm`. Previously the patch functions simply shifted coordinates inward so that vertices collected at the edge of the limits, where they would appear as lines along map borders, unless obscured by the map frame. This change allows OpenGL to better render the patch objects constructed by `patchm` and `patchesm`, making them more compatible with the use of `AlphaData` to achieve transparency. See the release note "Map Axes Now Display Transparent Objects More Easily" on page 153 for more details.

Compatibility Considerations

The more complete trimming in `patchm` and `patchesm` means that there are circumstances under which automatic reprojection can no longer display all the data provided to these functions. Automatic reprojection causes map objects created with `plotm`, `linem`, `patchm`, `patchesm`, and certain other display functions (but not `geoshow`) to be removed, projected, and redisplayed whenever a call to `setm` changes certain map axes properties, including the map limits and projection type. In the case of `patchm`, a set of polygons will become unavailable for automatic reprojection if *all* of the polygons are trimmed away completely. In the case of `patchesm`, which constructs a separate object for each polygon, *any* polygon that is trimmed away completely will be unavailable for reprojection, even if it would lie within newly defined map limits. In either of these cases, you should delete the handle(s) returned by `patchm` or `patchesm`, then repeat the original calls after changing your map axes properties.

Other potential compatibility issues:

- `patchm` and `patchesm` exhibit greater sensitivity to incomplete or incorrect polygon topology.
- You might need to manually set the renderer for proper display of some patch data

149

See the release note "Map Axes Now Display Transparent Objects More Easily" on page 153 for information about rendering and the Mapping Toolbox demo mapexgshhs for an example of a situation where polygon topology necessitates manual setting of the renderer.

Higher Quality boston.tif GeoTIFF Satellite Image
Compatibility Considerations: Yes

The original `boston.tif` GeoTIFF satellite image has been replaced by a higher resolution image, created by and provided courtesy of GeoEye™. The new image has the same name as the old one, `boston.tif`. The new `boston.tif` file, and an overview image in JPEG format, `boston_ovr.jpg`, include material copyright © by GeoEye, all rights reserved. The new image is 2881-by-4481 pixels, with a ground pixel size of 3.2808333333 U.S. survey feet (one meter). The original image was 720-by-1120 pixels and had a ground pixel size of four meters. Both images cover the downtown section of Boston, Massachusetts, the Charles River, and parts of Cambridge. The new image is a "pan-sharpened" multispectral image with visible red, green, and blue bands, and is stored in RGB form. The original image was also multispectral, but was a simple composite of red, green, and blue bands, and it was written to the GeoTIFF file as an indexed-color image. One additional change is that rather using meters, the new image is spatially referenced to the Massachusetts State Plane Mainland coordinate system with units of U.S. survey feet. The overview image, `boston_ovr.jpg`, is referenced to latitude-longitude, with a ground pixel size of approximately 16 meters. For further information, refer to the text files `boston.txt`, `boston_ovr.txt`, and `boston_metadata.txt` in `toolbox/map/mapdata`.

Compatibility Considerations

Older satellite images of Boston and a demo have been removed from Mapping Toolbox directories. The new `boston.tif` and `boston_ovr.jpg` images replace the images having the same names previously included in `toolbox/map/mapdemos`. In addition, several older images related to `boston.tif` have been removed:

- `boston_red.tif`
- `boston_green.tif`
- `boston_blue.tif`
- `boston_pan.tif`
- `boston_enhanced_pan.tif`

The mapexenhance demo ("Enhancing Multispectral GeoTIFF Images"), which used several of these images, has also been removed.

Map Axes Now Display Transparent Objects More Easily
Compatibility Considerations: Yes

It is now much easier to achieve transparency effects from the toolbox by setting the `AlphaData` property of an object. Previously, functions `axesm`, `lightm`, `contourm`, and `contour3m` set the figure's `Renderer` property: `axesm` and `lightm` set it to `'zbuffer'`, while `contourm` and `contour3m` set it to `'painters'`. You then had to manually reset `Renderer` to `'opengl'` in order for transparency to take effect.

Now the `RendererMode` of the figure retains the default MATLAB value of `'auto'`, causing MATLAB to select the most appropriate renderer for you; it will use OpenGL when appropriate, given your `AlphaData` settings. Using OpenGL not only enables transparency effects, it also can make use of hardware graphics acceleration capabilities should they be available.

Compatibility Considerations

If you need a particular map display to look the same as it did in Mapping Toolbox Version 2.5 (R2007a), in most cases you can just issue the command

```
set(gcf,'Renderer','zbuffer')
```

after you construct your map axes. If you are calling `contourm` or `contour3m`, issue the command

```
set(gcf,'Renderer','painters')
```

after you call the contouring function.

The consequence of doing this is that you will not be able to use transparency with that map figure until you reset its renderer to `'opengl'` or set its `'RendererMode'` back to `'auto'`, which is its default state.

153

The arcgridread Function Now Imports Noninteger Data Grids

In previous releases of the toolbox, `arcgridread` could only import data grids that had integer values (often of meters or feet). This limitation has now been removed, such that input grids can contain arbitrary values in decimal notation.

Change to avhrrlambert Function Behavior When No Data Is Available
Compatibility Considerations: Yes

In previous releases of the toolbox, `avhrrlambert` would error if the quadrangle defined by `latlim` and `lonlim` (when projected to form a polygon in the appropriate Lambert Equal Area Azimuthal projection) failed to intersect the bounding box of the data in the projected coordinates. In this release, `avhrrlambert` does not error when this occurs but returns empty matrices.

Compatibility Considerations

If you depend on `avhrrlambert` to error when there is no data in your quadrangle, you will need to change your code.

Enhancements to Mapping Toolbox User's Guide

Several sections of the chapter Understanding Geospatial Geometry have been rewritten and new material has been added to better explain critical topics such as ellipsoid models, units of and notations for angles and length, and the conversions that are possible between various units. There is also a new section, Exporting Vector Geodata, explaining and illustrating how to use the new `kmlwrite` and `makeattribspec` functions.

Functions deg2rad and rad2deg No Longer Convert Complex to Real
Compatibility Considerations: Yes

In prior versions, when given complex inputs, functions `deg2rad` and `rad2deg` issued a warning and then converted their inputs to real. Now they no longer do either of these things.

Compatibility Considerations

In the unlikely event of complex input, these functions simply scale the imaginary part by the same factor as the real part. For example, in R2007a and earlier releases, they behave as follows:

```
>> deg2rad(180i)
Warning: Imaginary parts of complex ANGLE argument ignored
> In deg2rad at 16
ans =
            0
```

Going forward from this release, the result is

```
>> deg2rad(180i)
ans =
        0 + 3.1416i
```

Degrees-Minutes-Seconds Conversion Functions Are Obsolete and Error if Used
Compatibility Considerations: Yes

The following functions, which accepted or produced double scalars to represent degrees, minutes, and seconds now error when used, and will be removed completely from the toolbox in a future release:

- deg2dm

- deg2dms

- dms2deg

- dms2dm

- dms2mat

- dms2rad

- mat2dms

- rad2dm

- rad2dms

The scalar DM and DMS encodings are being eliminated from the toolbox because they were never used for internal computations, and always had the potential to generate serious numerical errors if passed accidentally to functions that expected normal latitude-longitude tuples. They also made the functions that accepted them less efficient due to the need to convert from DM or DMS to fractional latitudes and longitudes before processing the input data.

In every case, an alternative that does not use the old degrees-minutes-seconds scalar encoding exists. See the following section on compatibility for replacements and "New Functions for Degrees-Minutes-Seconds Conversions" on page 170 in the V2.5 Release Notes for descriptions of replacement functions, and the compatibility considerations below for descriptions of alternative syntaxes and expressions you can use for degrees-minutes-seconds conversions.

Compatibility Considerations

DM and DMS representations are widely used in published reports and can occur in geodata that you want to read into the MATLAB workspace. You can still import and export DM and DMS data, but Mapping Toolbox functions no longer accepts the old encodings as alternatives to floating-point representations of latitude and longitude for internal manipulations.

The following functions (which all use scalar DMS encoding) are being retired. They remain in the product for R2007b, but now generate errors when used. They will be removed completely in the next version. Use the alternative suggested in lieu of these functions.

- `deg2dm` — Instead use `degrees2dm` to convert degrees to degrees-minutes vector.

- `deg2dms` — Instead use `degrees2dms` to convert degrees to degrees-minutes-seconds vector.

- `dms2deg` — Instead use `dms2degrees` to convert degrees-minutes-seconds vector to degrees.

- `dms2dm` — Instead combine `dms2degrees` and `degrees2dm`, as in `degrees2dm(dms2degrees([-29 42 18.7]))` to remove the seconds component from a degree-minutes-second vector.

- `dms2mat` — Instead use `degrees2dms` to convert degrees to degrees-minutes-seconds vector.

- `dms2rad` — Instead use `dms2degrees` to convert degrees-minutes-seconds vector to degrees and call `degtorad` or multiply by `pi/180`.

- `mat2dms` — Instead use `dms2degrees` to convert degrees-minutes-seconds vector to degrees.

- `rad2dm` — Instead, call `radtodeg` or multiply input arguments by `180/pi`, and then call `degrees2dm`.

- `rad2dms` — Instead, call `radtodeg` or multiply input arguments by `180/pi`, and then call `degrees2dms`.

In addition, the `axesm` and `setm` functions no longer accept the strings `'dms'` and `'dm'` for setting either the *AngleUnits* or *LabelUnits* properties of a map axes.

Many other Mapping Toolbox functions optionally accept angle strings for their units parameter; the following 57 functions now only accept 'degrees' and 'radians', whereas in prior versions they would also accept 'dm' and 'dms' as values for *units*:

angl2str	distance	histr	putpole	stdist
angledim	eastof	imbedm	rcurve	stdm
antipode	elevation	interpm	reckon	timezone
areaint	ellipse1	intrplat	rhxrh	track
areamat	epsm	intrplon	rotatem	track1
areaquad	eqa2grn	mapprofile	rsphere	track2
axesm	gc2sc	meanm	scaleruler	unitstr
azimuth	gcxgc	meshgrat	scircle1	westof
convertlat	gcxsc	neworig	scircle2	zero22pi
crossfix	gradientm	newpole	scxsc	
daspectm	grn2eqa	npi2pi	setm	
departure	hista	org2pol	smoothlong	

These functions now error when provided 'dm' or 'dms' for their units argument.

Time Conversion Functions Are Obsolete and Error if Used
Compatibility Considerations: Yes

The following functions, which converted time representations, now error when used and will be removed completely from the toolbox in a future release:

- `hms2hm`

- `hms2hr`

- `hms2mat`

- `hms2sec`

- `hr2hm`

- `hr2hms`

- `hr2sec`

- `mat2hms`

- `sec2hm`

- `sec2hms`

- `sec2hr`

- `time2str`

- `timedim`

Compatibility Considerations

These functions now raise errors when they are invoked. They will be completely removed in a future version of the toolbox. No substitutes have been provided, as no operations of the toolbox have ever depended on them.

cmapui GUI is now Obsolete
Compatibility Considerations: Yes

`cmapui` GUI will be completely removed from the next Mapping Toolbox version.

Compatibility Considerations

It now errors if you attempt to use it. Use the MATLAB `colormapeditor` GUI instead, which provides better functionality. You can also use the **Colormap** drop-down menu in the Property Editor (part of the MATLAB plotting tools and available via the `propedit` command) to select a built-in colormap; the `custom` option on that drop-down menu opens `colormapeditor`. To set up a colormap for terrain displays, you can use the `demcmap` function. To generate an appropriate (but random) colormap for political maps, use the `polcmap` function.

Functions tigermif and tigerp Are Obsolete and Error if Used
Compatibility Considerations: Yes

The following functions error and issue an error message when you attempt to use them:

- `tigerp` — Read TIGER p and pa thinned boundary files (ArcInfo format)

- `tigermif` — Read the TIGER MIF thinned boundary file (MapInfo format)

Compatibility Considerations

In place of these format readers, download U.S. Census cartographic boundary files in shapefile format and use `shaperead` to import them.

R2007a

Version: 2.5
New Features: Yes
Bug Fixes: No

Performance Improvements for los2 and viewshed

This release includes a faster `los2` function (which computes intervisibility between locations on or above a terrain grid). The `viewshed` function (which computes the portions of a terrain grid that can be seen from a given viewpoint) has also been accelerated as a result.

Utility Functions for Computing Distance and Position Along Meridians

Two functions that reckon position and distance along a meridian on the ellipsoid are now available:

- `meridianarc` — Computes distance along a meridian between two latitudes

- `meridianfwd` — Reckons position along meridian given a starting point and distance

Some GUIs Are No Longer Available from the Command Line
Compatibility Considerations: Yes

In prior releases, when you typed certain Mapping Toolbox function names with no argument list, a specialized GUI appeared that enabled you to interactively set parameters related to the function. This feature was seldom used and sometimes raised errors when users attempted to operate the GUIs. Starting in this release, a GUI will no longer appear when you issue the following commands:

- `comet3m`
- `cometm`
- `contourfm`
- `contour3m`
- `contourm`
- `demcmap`
- `fill3m`
- `fillm`
- `lightm`
- `limitm`
- `linem`
- `meshlsrm`
- `meshm`
- `patchesm`
- `patchm`
- `pcolorm`
- `plot3m`
- `plotm`
- `quiver3m`

- quiverm

- scatterm

- stem3m

- surfacem

- surflm

- surfm

- surflsrm

- symbolm

- textm

Compatibility Considerations

Use the above functions with arguments to avoid raising errors. Their GUIs will continue to be available via maptool (which places menus on a figure containing map axes), but they are not being actively supported and will be eliminated in a future release.

New Functions for Degrees-Minutes-Seconds Conversions
Compatibility Considerations: Yes

Four new functions have been added to convert to and from decimal degrees and degrees-minutes-seconds (DMS):

- `dms2degrees` — Convert degrees-minutes-seconds to degrees
- `dm2degrees` — Convert degrees-minutes to degrees
- `degrees2dms` — Convert degrees to degrees-minutes-seconds
- `degrees2dm` — Convert degrees to degrees-minutes

The DMS inputs and outputs of these functions are vectors of one row and three columns for each row in the decimal degrees input or output. The first column contains the "degrees" element and is integer-valued. The second column contains the "minutes" element and is integer-valued. The third column contains the "seconds" element, and may have a nonzero fractional part. Similarly, DM inputs and outputs are two-column vectors with integer degrees and fractional minutes parts.

The new conversion functions dispense with the DMS encoding used in prior versions of the toolbox. These represented DMS angles by a single real number, the format of which is `dddmm.ss`. Such an encoding is no longer used internally by Mapping Toolbox functions, as it is not self-documenting and can lead to erroneous computations. For example, two DMS-encoded real numbers cannot be added to obtain a meaningful result.

Compatibility Considerations

DM and DMS representations are widely used in published reports and can occur in geodata that you want to read into the MATLAB workspace. You can still import and export DM and DMS data, but Mapping Toolbox functions no longer accepts the old encodings as alternatives to floating point representations of latitude and longitude for internal manipulations.

The scalar DM and DMS encodings are being eliminated from the toolbox because they were never used for internal computations, and always had the potential to generate serious numerical errors if passed accidently to functions

that expected normal latitude-longitude tuples. They also made the functions that accepted them less efficient due to the need to convert from DM or DMS to fractional latitudes and longitudes before processing the input data.

The following existing functions (which all use scalar DMS encoding) are being retired. They remain available but now issue warnings that they are obsolete when used:

- `deg2dm` — Instead use `degrees2dm` to convert degrees to degrees-minutes vector

- `deg2dms` — Instead use `degrees2dms` to convert degrees to degrees-minutes-seconds vector

- `dms2deg` — Instead use `dms2degrees` to convert degrees-minutes-seconds vector to degrees

- `dms2mat` — Instead use `degrees2dms` to convert degrees to degrees-minutes-seconds vector

- `dms2rad` — Instead use `dms2degrees` to convert degrees-minutes-seconds vector to degrees and call `degtorad` or multiply by `pi/180`

- `mat2dms` — Instead use `dms2degrees` to convert degrees-minutes-seconds vector to degrees

- `rad2dm` — Instead, call `radtodeg` or multiply input arguments by `180/pi`, and then call `degrees2dm`

- `rad2dms` — Instead, call `radtodeg` or multiply input arguments by `180/pi`, and then call `degrees2dms`

In addition, the `axesm` and `setm` functions no longer accept the strings `'dms'` and `'dm'` for setting either the `AngleUnits` or `LabelUnits` properties of a map axes.

Many other Mapping Toolbox functions optionally accept angle strings for their `units` parameter; the following 57 functions now only accept `'degrees'` and `'radians'`, whereas in prior versions they would also accept `'dm'` and `'dms'` as values for *units*:

angl2str	distance	histr	putpole	stdist
angledim	eastof	imbedm	rcurve	stdm
antipode	elevation	interpm	reckon	timezone
areaint	ellipse1	intrplat	rhxrh	track
areamat	epsm	intrplon	rotatem	track1
areaquad	eqa2grn	mapprofile	rsphere	track2
axesm	gc2sc	meanm	scaleruler	unitstr
azimuth	gcxgc	meshgrat	scircle1	westof
convertlat	gcxsc	neworig	scircle2	zero22pi
crossfix	gradientm	newpole	scxsc	
daspectm	grn2eqa	npi2pi	setm	
departure	hista	org2pol	smoothlong	

These functions now issue warnings when provided `'dm'` or `'dms'` for their `units` argument.

Time Conversion Functions to be Removed
Compatibility Considerations: Yes

The following functions to convert between time units and encodings will be removed from a future release of the toolbox:

- hms2hr

- hms2hm

- hms2mat

- hms2sec

- hr2hm

- hr2hms

- hr2sec

- mat2hms

- sec2hm

- sec2hms

- sec2hr

- time2str

- timedim

Compatibility Considerations

These functions remain available, but when they are invoked now issue warnings that they are obsolete.

R2006b

Version: 2.4
New Features: Yes
Bug Fixes: No

Standard Formulations of Five Major Map Projections

New formulations of five conic map projections are provided. The existing implementations remain available under their old names. The new versions use the same names as the ones they supplement, appended with "std":

- Cassini Transverse Cylindrical (`cassinistd`)

- Albers Equal-Area Conic (`eqaconicstd`)

- Equidistant Conic (`eqdconicstd`)

- Lambert Conformal Conic (`lambertstd`)

- Polyconic (`polyconstd`)

Computations used for the new versions differ from the old ones only when the latitude origin (the first element of the *origin vector*) is nonzero. In this case, the old versions shift the origin off the equator through a solid body rotation of the sphere (or, for an ellipsoidal earth model, a suitable auxiliary sphere). This is technically correct, but differs from accepted industry standards for these projections. The new versions use the standard formulations and give results that are consistent with projection results from other software packages, regardless of the latitude origin. The old versions are retained in the toolbox, with no change in behavior, to ensure backward compatibility.

See the Projections Reference documentation for more information.

Two New Geodetic/Geocentric Latitude Conversion Functions

Two new functions provide a more direct route to functionality already available via the `convertlat` function:

- `geocentric2geodeticlat` converts an array of geocentric latitude in radians to geodetic latitude in radians on a reference ellipsoid given a first eccentricity

- `geodetic2geocentriclat` converts an array of geodetic latitude in radians to geocentric latitude in radians on a reference ellipsoid given a first eccentricity.

Accelerated Performance for geoshow, mapshow, and bufferm

Functions `geoshow`, `mapshow`, and `bufferm` run substantially faster in many cases, especially when vector display is being controlled via symbol specs in `mapshow` and `geoshow`.

Changes in Behavior of mapshow and geoshow
Compatibility Considerations: Yes

In addition to operating faster, the `mapshow` and `geoshow` functions now behave slightly differently regarding their defaults, handles returned, warnings issued, and several other aspects:

Compatibility Considerations

Default Symbols and Colors

- Point marker type changes from `'X'` to `'+'`

- Point marker color changes from `'black'` to `'red'`

- Line color changes from `'black'` to `'blue'`

- Polygon facecolor changes from `'black'` to pale yellow

Polygon edgecolor remains `'black'`

Contour DisplayType Behavior Changes

- The `DisplayType` option `'contour'` now returns an hggroup handle. The children of the hggroup are patches. In prior versions, an array of line handles was returned.

- You can specify any contourgroup property as a parameter value pair. In previous versions, `mapshow` allowed you to set the `LineStyle` property, but no other contour properties.

- Both `mapshow` and `geoshow` might return a different number of contour levels by default than in previous versions, in which you could not specify contour intervals; in R2006b, you can control contour intervals and levels via the `LevelStep` or `LevelList` contourgroup properties, among others.

- In R2006b, when plotting contours on a regular axes (not a map axes), geoshow projects the contour lines using a Plate Carree projection; in previous versions it simply displayed longitudes as *x* and latitudes as *y* without doing any trimming or longitude wrapping.

Graphic Objects and Return Values for Vector Inputs

- Vector coordinate array input (*x-y* or latitude-longitude pairs) with a `DisplayType` of `'Line'` or `'Point'` now generates an ordinary line object instead of a map graphics line.

- For geostruct input, an hggroup object is constructed; its handle is returned instead of an array of handles to map graphic objects:

  - For polygon geostructs, map graphics polygon objects are still constructed, but become children of the hggroup.

  - For point, multipoint, and line geostructs, the children of the hggroup are ordinary line objects; map graphics objects are no longer constructed.

  In both cases each child of the hggroup, rather than each element in an array of handles, corresponds to a distinct feature in the geostruct.

Handles Returned for Graphic Objects

- Geostruct inputs result in an hggroup handle containing either line objects (for point, multipoint, and line inputs) or modified patch objects (for polygon inputs) as their children.

- Coordinate arrays (*x,y* pairs) displayed as lines now result in ordinary line objects.

- Geostructs containing lines result as ordinary line objects within hggroups.

New Warnings Issued

- `mapshow` and `geoshow` now warn when given a geostruct within which the `Geometry` field differs from a specified `'DisplayType'` parameter.

- `mapshow` will warn if it is given a geostruct containing `Lat` and `Lon` fields instead of `X` and `Y` fields.

- `geoshow` will warn if it is given a geostruct containing `X` and `Y` fields instead of `Lat` and `Lon` fields.

geoshow Supplies Default Projection

`geoshow` now projects vector and raster inputs using a default projection (Plate Carree) if the parent axes is not a map axes. The axes itself is unchanged (it is not modified to become a map axes), but the scale factor of

the projection is set such that latitudes and longitudes in degrees can be read directly from the axes ticks and grid lines.

Duplicate Parameter/Value Pair Inputs Allowed

`mapshow` no longer errors or warns if given duplicate Parameter/Value pair inputs; in such circumstances, `mapshow` now uses the last value (even with SymbolSpecs)

geoshow Supports True Surface Display

`geoshow` now creates a true 3-D surface if given a `'surface'` `DisplayType` rather than setting the `ZData` values to `0`.

Texturemap DisplayType Behavior Changes

The `'texturemap'` `DisplayType` now uses the pixel edges to create `XData` and `YData` grids rather than using the pixel centers, which correctly registers the display to map coordinates. The `ZData` contains an array of zeros having the same dimensions as the `XData` and YData `arrays`, which exceed the input grid in size by one in both the *x* and *y* dimensions.

You should use `'texturemap'` displays when the attribute being displayed is coded by color (i.e. 2-D displays); use `'surface'` displays when you need to show data with relief (nonzero `ZData`).

More General Support for Graphics Properties

All Handle Graphics® patch properties are now supported for polygon inputs.

All Handle Graphics line properties are now supported for point and line inputs, except that `'linestyle'` is ignored for point inputs.

Limitations on Referencing Matrices for Geoshow Removed

`geoshow` is now capable of accepting any referencing matrix. Previously it could only accept those referencing matrices that were convertible to referencing vectors.

mapshow and geoshow Ignore Empty Inputs Rather than Erroring

In previous versions, `mapshow` and `geoshow` would throw errors when provided with empty (`[]`) arrays. This behavior could be inconvenient when running these functions via scripts. The new behavior is also more consistent with that of MATLAB plotting functions such as `plot`, `surf`, `mesh`, and `contour`.

dted Automatically Fixes Incorrectly Specified Longitude Directions in DTED Data

Some DTED level 0 files available via the National Geospatial-Intelligence Agency's (NGA) web interface may have minor errors. Specifically, Level 0 data for cells just to the east of the prime meridian may have longitude coordinate strings with 'W' substituted for 'E'. The `dted` function now detects and automatically corrects this data error.

R2006a

Version: 2.3
New Features: Yes
Bug Fixes: No

Full Support for 64-Bit Windows

Version 2.3 adds support for the mex- and library-based functions `geotiffinfo`, `geotiffread`, `sdtsinfo`, and `sdtsdemread` on this new MATLAB platform via library upgrades (described below) and a custom port of STDS++.

Third-Party Library and Code Upgrades

Third-party libraries and software packages have been upgraded to their current versions to ensure best performance and compatibility with external geospatial data sources and applications software:

- General Polygon Clipper (GPC) upgraded to Version 2.32
- PROJ.4 library upgraded to Version 4.4.9
- SDTS++ library upgraded to Version 1.5.1
- GeoTIFF library upgraded to Version 1.2.2

Support for 32-Bit Floating-Point GeoTIFF Images
Compatibility Considerations: Yes

The MATLAB function imread can now import TIFF images containing 32-bit floating-point data. As a result, geotiffread now reads the corresponding variety of GeoTIFF.

Compatibility Considerations

The structure returned by geotiffinfo in V. 2.3 has changed. The following table describes the differences between the current and previous versions:

Version 2.3	Previous Versions
The TiePoints structure contains two substructures, ImagePoints and WorldPoints. ImagePoints contains [1-by-N] arrays Row and Col; WorldPoints contains [1-by-N] arrays X and Y.	The TiePoints structure contained two [3-by-1] arrays, ImagePoints and WorldPoints.
The CornerCoords structure contains six [1-by-4] row vectors, respectively, X, Y, Col, Row, Lat, and Lon.	The CornerCoords structure contained six [4-by-1] column vectors: PCSX, PCSY, X, Y, LAT, and LON.
The Zone field contains [] if the UTM zone is not applicable or was missing from the metadata.	The Zone field contained 32767 if the UTM zone was not applicable or was missing from the metadata.

Utility Functions for NaN-Separated Polygons and Lines

closePolygonParts

Closes all rings in a multipart polygon to ensure proper analysis and rendering.

isShapeMultipart

Boolean-valued function that returns `true` if a polygon or line has multiple parts.

removeExtraNanSeparators

Eliminates redundant `NaN` separators that might exist in polygons and lines.

Standardized Vector Topology in coast.mat

Polygons in the low-resolution coastline sample data file `coast.mat` now follow the convention used by `geoshow`, `mapshow`, and `mapview` to display polygons with "holes" (inner rings representing lakes, inland seas, and islands within them). Outer contours now always run clockwise and inner contours run counterclockwise. These edits, which reversed the order of vertices in some rings, enable the display functions to fill outer rings properly while leaving inner rings blank.

Three New Demos

If you are viewing these release notes using the Help browser, clicking any of the demo links below will open the demo in a browser window. Click the links at the top of that window to view or run the code for the demo.

Converting Coastline Data (GSHHS) to Shapefile Format

Shows how to extract coastlines from the Global Self-consistent Hierarchical High-resolution Shorelines (GSHHS) data set, manipulate the polygon features, and save the result to a polygon shapefile.

Plotting a 3-D Dome as a Mesh Over a Globe

Illustrates how to construct a 3-D feature in a system of local vertical coordinates, then transform and combine it with a globe display in Earth-Centered, Earth-Fixed (ECEF) coordinates.

Unprojecting a Digital Elevation Model (DEM)

Shows how to unproject a georeferenced terrain elevation grid from Universal Transverse Mercator (UTM) into a regular latitude-longitude grid having comparable spatial resolution.

R14SP3

Version: 2.2
New Features: Yes
Bug Fixes: No

Geodetic-Geocentric Coordinate Conversion Functions

New three-dimensional coordinate conversion functions (`geodetic2ecef`, `ecef2geodetic`, `ecef2lv`, `lv2ecef`) transform 3-D point locations between geodetic (latitude, longitude, height), geocentric Cartesian (Earth Centered, Earth Fixed), and local vertical Cartesian coordinate systems.

Additional User Control Over Shapefile Content

Function `shapewrite` now allows user control over field names, lengths, and decimal precision when writing feature attributes to the DBF file, via a "DBF specification." The new function `makedbfspec` constructs a default DBF specification from a geographic data structure (geostruct2) array. Users can customize the output and pass it to `shapewrite`.

Shapefile Read/Write Efficiency Enhanced

Improved implementations of functions `shaperead` and `shapewrite` process data substantially faster (about four times faster for a 10-MB shapefile of major roads in Massachusetts).

Improved Rendering of Polygons with Inner Rings
Compatibility Considerations: Yes

The Map Viewer (function `mapview`) and functions `mapshow` and `geoshow` now properly render polygons containing inner rings (e.g., lakes and inland seas within a continent, islands within a pond). Features in underlying layers "show through" inner rings because they are not obscured by the patch faces used to render the polygons.

Compatibility Considerations

Polygon Vertex Ordering Is Now Significant for Properly Rendering Filled Polygons

The map display functions `geoshow`, `mapshow`, and `mapview` now require that coordinate vectors representing polygons have consistent directionality, such that

- Vertices defining outer rings (to be filled) be encoded in a *clockwise* direction.

- Vertices defining inner rings (often termed "lakes" or "islands," to be rendered as transparent holes) be encoded in a *counterclockwise* direction.

If you have vector map data sets that violate these conditions, the map display functions `geoshow`, `mapshow`, and `mapview` might not be capable of rendering them as filled polygons. To determine the directionality of polygon vertices, use the logical function `ispolycw`, which returns a separate result for each `NaN`-delimited polygon in an array of vertices. If you find inner rings which are clockwise or outer rings which are counterclockwise, use the utility functions `poly2ccw` or `poly2cw`, respectively, to reverse the direction of those rings.

Map Viewer Now Georeferences Images It Saves

When the Map Viewer saves the visible or selected area as a raster map (an image file), it now also writes a worldfile to georeference the image.

TIGER/Line File Support Upgraded

Function `tgrline` now supports the most recent (2003/2004) TIGER/Line data sets from the U.S. Bureau of the Census.

R14SP2

Version: 2.1
New Features: Yes
Bug Fixes: No

New Function Reads Both 5-Minute and 2-Minute ETOPO Data

The new function `etopo` reads from either the 5-minute (ETOPO5) or the 2-minute (ETOPO2) global terrain data set. This function supersedes function `etopo5` and fixes several significant bugs.

Function gshhs Now Returns a Version 2 Geostruct

Function `gshhs`, which reads the Global Self-consistent Hierarchical High-resolution Shoreline data set, has been upgraded. It now returns a Version 2 geographic data structure (geostruct2) array instead of a Version 1 geostruct. Polygons returned from `gshhs` now follow the shapefile vertex-ordering convention (supported by functions `polybool`, `shaperead`, and `shapewrite`, for example). Under this convention the coordinates of outer rings (e.g., continent outlines) are given in clockwise order, while counterclockwise ordering is used for inner rings (e.g., lakes and inland seas within a continent). Note that function `gshhs` does not yet support Version 1.3 of the data set, released on Sept. 27, 2004.

Geodata Can Now Be Exported in Shapefile Format

The new function `shapewrite` writes a geographic data structure to a shapefile. It exports a Version 2 geographic data structure array (geostruct2), creating `.shp`, `.shx`, and `.dbf` files. Like `shaperead`, the function supports the Point, MultiPoint, PolyLine, and Polygon shape types. The contents of string-valued attribute fields and scalar numerical attribute fields are written to the dBase (`.dbf`) file.

Accessing Geodata Resources on the Internet

Links and URLs to documentation and data files for various Internet sources of digital map data are now collected in the following technical note on the MathWorks Web site:

`http://www.mathworks.com/support/tech-notes/2100/2101.html`

This technical note replaces many individual links formerly scattered across the User's Guide, reference pages, and MATLAB function help. Collecting this information on a Web page rather than on product CDs or printed documentation should substantially mitigate recurrent problems with stale links. Please report any stale links that you might find in the technical note to MathWorks Technical Support (`http://www.mathworks.com/contact_TS.html`), so that it can be updated promptly.

Changes to Atlas Data and Associated Functions
Compatibility Considerations: Yes

Through Version 2.0.3, the toolbox included a set of *atlas data* with global geopolitical data embedded as MATLAB arrays in four MAT-files: `worldlo`, `worldhi`, `worldmtx`, and `worldmtxmed`. However, geopolitical data is difficult to keep current, and is subject to inaccuracies and interpretations that can cause contention. Therefore, starting with Version 2.1, Mapping Toolbox demo data now excludes geopolitical data that would specify national sovereignty over specific regions of the Earth. The only exceptions are the boundaries of the 50 U.S. states and the District of Columbia.

Compatibility Considerations

This change means that the `worldlo`, `worldhi`, `worldmtx`, and `worldmtxmed` MAT-files are no longer part of the toolbox. However, the nonpolitical data on global coastlines, major lakes and inland seas, major rivers, and major cities and populated places that was in `worldlo.mat` has been retained in the toolbox and transformed into shapefile format. This includes the addition of name attributes for many previously unnamed features. There are four new shapefiles in this category: `landareas.shp`, `worldlakes.shp`, `worldrivers.shp`, and `worldcities.shp`.

For consistency, the atlas data for the United States that was originally stored in the `usalo` and `usahi` MAT-files has also been transformed, although none has been removed. These data sets now reside in the following shapefiles and MAT-files: `usastatelo.shp`, `usastatehi.shp`, `conus.mat`, and `greatlakes.mat`.

The toolbox originally included four functions dedicated to extracting data from the atlas data MAT-files: `worldlo`, `worldhi`, `usalo`, and `usahi`. With the data removal/transformation described above, these functions are no longer needed and have been removed from the toolbox in Version 2.1. You can easily access the new shapefiles using the `shaperead` function, which includes powerful and flexible options for selecting features and even controlling which attributes are read. In addition, function `country2mtx`, whose sole purpose was to rasterize the country boundary polygons in `worldlo.mat`, has been removed.

Related changes extend to the `worldmap` function, which formerly combined two purposes:

- Select an appropriate map projection and parameters with which to display a given latitude-longitude area.

- Automatically display atlas data for that area.

In Version 2.1, `worldmap` supports only the first of these actions. A call to `worldmap` constructs a map axes object and can easily be followed with a variety of Mapping Toolbox commands to display the map data of your choice. Because the `usamap` function is so similar to `worldmap`, corresponding changes have been made there as well.

To help those who have relied heavily on `worldmap` and `usamap` to plot base maps with automatically selected vector map data, examples throughout the User's Guide, reference pages, and MATLAB function help entries have been updated to illustrate the new behavior of `worldmap` and `usamap`, and to show how to create maps including vector shapefile data layers. These examples cover a wide variety of ways to read and subset data with `shaperead` and display data with `geoshow` and other Mapping Toolbox display functions. A good place to start is with the examples for the `worldmap` and `usamap` functions. Also see example code in .

To help you update commands, scripts, and data for constructing and maintaining base maps, a recently published technical note on the MathWorks Web site provides links to data and documentation for many sources of vector and raster digital map data that you can access over the Internet:

`http://www.mathworks.com/support/tech-notes/2100/2101.html`

Changes to worldmap and usamap
Compatibility Considerations: Yes

`worldmap` and `usamap` have been simplified to construct appropriate map axes for a given area without displaying any map data.

In all cases, map frames, latitude-longitude grid lines, meridian labels, and parallel labels are turned on. You can use the following command sequence to remove them:

```
framem off; gridm off; mlabel off; plabel off
```

Other changes include the following:

- `usamap` now accepts two-letter U.S. Postal Service abbreviations for state names (e.g., AL, AK, AR, etc.).

Compatibility Considerations

- The following input options are now obsolete (if used, a warning is issued):
  - A first argument equal to `'lo'` or `'hi'`
  - The `regiononly` and `stateonly` syntax: a state or country name with the string `'only'` appended
  - All `type` options: `'none'`, `'line'`, `'lineonly'`, `'patch'`, `'patchonly'`, `'mesh'`, `'meshonly'`, `'dem'`, `'demonly'`, `'dem3d'`, `'dem3donly'`, `'lmesh3d'`, `'lmesh3donly'`, `'ldem3d'`, and `'ldem3donly'` (the new behavior matches the `'none'` option)

Changes to `worldmap` and `usamap` Display Types

As of this release, the `worldmap` and `usamap` functions no longer supports the *type* input argument. This argument provided an easy way to control display behavior.

The *type* option in `worldmap` was a single argument that could be one of the following strings: `'none'`, `'line'`, `'lineonly'`, `'patch'`, `'patchonly'`, `'mesh'`, `'meshonly'`, `'dem'`, `'demonly'`, `'dem3d'`, `'dem3donly'`, `'lmesh3d'`,

'lmesh3donly', 'ldem3d', and 'ldem3donly'. In usamap, *type* was a subset of the above names (the 3-D options were not supported).

In the current release, the various *type* display options can be simulated by following a call to worldmap or usamap with an appropriate set of Mapping Toolbox commands. The following table specifies how you can achieve the effects of the old worldmap *type* argument using such auxiliary methods:

Mapping 1.x to 2.0.x Usage	Mapping 2.1 Usage
```load topo``` ```worldmap(topo, ...``` ```topolegend, 'dem')```	```load topo``` ```worldmap(topo, topolegend)``` ```meshm(topo, topolegend)``` ```demcmap(topo)``` ```land = shaperead('landareas.shp',...``` ```'UseGeoCoords', true);``` ```geoshow([land.Lat], [land.Lon])```
```load topo``` ```worlmdap(topo, ...``` ```topolegend,``` ```'demonly')```	```load topo``` ```worldmap(topo, topolegend)``` ```meshm(topo, topolegend)``` ```demcmap(topo)```
```load topo``` ```worldmap(topo, ...``` ```topolegend,``` ```'dem3d')```	```load topo``` ```worldmap(topo, topolegend)``` ```meshm(topo, topolegend,size(topo),topo)``` ```da = daspect;``` ```pba = pbaspect;``` ```da(3) = 7.5*pba(3)/da(3);``` ```daspect(da);``` ```demcmap(topo)``` ```land = shaperead('landareas.shp',...```

Mapping 1.x to 2.0.x Usage	Mapping 2.1 Usage
	`'UseGeoCoords', true);`
	`geoshow([land.Lat], [land.Lon])`
`load topo`	`load topo`
`worldmap(topo, ...`	`worldmap(topo, topolegend)`
`topolegend, 'dem3donly')`	`meshm(topo, topolegend,size(topo),topo)`
	`demcmap(topo)`
`load korea`	`load korea`
`worldmap (map, refvec,...`	`worldmap(map, refvec)`
`'mesh')`	`meshm(map, refvec)`
	`land = shaperead('landareas.shp',...`
	`'UseGeoCoords', true);`
	`geoshow([land.Lat], [land.Lon])`
	(Text North Korea and South Korea will be missing)
`load korea`	`load korea`
`worldmap(map, refvec,...`	`worldmap(map, refvec)`
`'meshonly')`	`meshm(map,refvec)`
`load topo`	`load topo`
`worldmap(topo,...`	`worldmap(topo, topolegend)`
`topolegend, 'mesh3d')`	`meshm(topo, topolegend,size(topo),topo)`
	`da = daspect;`
	`pba = pbaspect;`
	`da(3) = 7.5*pba(3)/da(3);`
	`daspect(da);`

Mapping 1.x to 2.0.x Usage	Mapping 2.1 Usage
| ```
load topo
worldmap...
(topo,topolegend,...
'mesh3donly')
``` | ```
load topo
worldmap(topo, topolegend)
meshm(topo, topolegend,size(topo),top
``` |
| ```
load topo
worldmap(topo,...
topolegend,
'ldem3d')
``` | ```
load topo
worldmap(topo, topolegend)
meshm(topo, topolegend,size(topo),topo)
da = daspect;
pba = pbaspect;
da(3) = 7.5*pba(3)/da(3);
daspect(da);
demcmap(topo)
camlight(90,5);
camlight(0,5);
lighting phong
material([0.25 0.8 0])
``` |
| ```
load topo
worldmap(topo, ...
topolegend,'ldem3donly')
``` | ```
load topo
worldmap(topo, topolegend)
meshm(topo, topolegend,size(topo),topo)
da = daspect;
pba = pbaspect;
da(3) = 7.5*pba(3)/da(3);
daspect(da);
demcmap(topo)
``` |

Mapping 1.x to 2.0.x Usage	Mapping 2.1 Usage
`load topo`	`load topo`
`worldmap(topo,...`	`worldmap(topo, topolegend)`
`topolegend,` `'lmesh3d'`	`meshm(topo, topolegend,size(topo),topo)`
	`da = daspect;`
	`pba = pbaspect;`
	`da(3) = 2*pba(3)/da(3);`
	`daspect(da);`
	`camlight(90,5);`
	`camlight(0,5);`
	`lighting phong`
	`material([0.25 0.8 0])`
`load topo`	`load topo`
`worldmap(topo,...`	`worldmap(topo, topolegend)`
`  topolegend,...`	`meshm(topo, topolegend,size(topo),topo)`
`  'lmesh3donly')`	`da = daspect;`
	`pba = pbaspect;`
	`da(3) = 2*pba(3)/da(3);`
	`daspect(da);`

Mapping 1.x to 2.0.x Usage	Mapping 2.1 Usage

`load korea`
 	
`worldmap(map, refvec,...`
	`load korea`
	
	`worldmap(map, refvec)`
	
	`land =...`
	
	`shaperead('landareas.shp',...`
	
	`'UseGeoCoords', true);`
	
	`geoshow([land.Lat], [land.Lon])`
	
	(Text North Korea and South Korea will be missing. Land area boundaries resolution is lower.)

`load korea`
 	
`worldmap(map, refvec,...`
 	
`'lineonly')`
	`load korea`
	
	`worldmap(map, refvec)`
	
	`land = ...`
	
	`shaperead('landareas.shp',...`
	
	`'UseGeoCoords', true);`
	
	`geoshow([land.Lat], [land.Lon])`

`load korea`
 	
`'line')`
 	
`worldmap(map, refvec,...`
 	
`'none')`
	`load korea`
	
	`worldmap(map, refvec)`

Mapping 1.x to 2.0.x Usage	Mapping 2.1 Usage

`load korea`
 	
`worldmap(map,`
`refvec,...`
 	
`  'patch')`
	`load korea`
	
	`worldmap(map, refvec)`
	
	`land = ...`
	
	`shaperead('landareas.shp',...`
	
	`'UseGeoCoords', true);`
	
	`faceColors = ...`
	
	`makesymbolspec('Polygon',...`
	
	`{'INDEX', [1 numel(land)],...`
	
	`'FaceColor', ...`
	
	`polcmap(numel(land))});`
	
	`geoshow(land,'SymbolSpec',...`
	
	`makesymbolspec('Polygon',...`
	
	`faceColors)`
	
	(Text North Korea and South Korea will be missing. Country coloring will be missing.)

`load korea`
 	
`worldmap(map,`
`refvec,...`
 	
`'patchonly')`
	`load korea`
	
	`worldmap(map, refvec)`
	
	`land = ...`
	
	`shaperead('landareas.shp',...`
	
	`'UseGeoCoords', true);`
	
	`faceColors =...`
	
	`{'INDEX', [1 numel(land)],...`
	
	`'FaceColor', ...`
	
	`polcmap(numel(land))});`
	
	`geoshow(land,'SymbolSpec',...`

Mapping 1.x to 2.0.x Usage	Mapping 2.1 Usage
	`faceColors)`
	(Country coloring will be missing.)

Data Files Added in This Release

The following files were added to the `mapdemos` directory, for use in toolbox demos and examples:

- `landareas` — Polygon shapefile: global coastlines, both exterior and interior, including names for larger land masses

- `worldlakes` — Polygon shapefile: coastlines and names of major lakes and inland seas worldwide

- `worldrivers` — PolyLine shapefile: major world rivers and their names

- `worldcities` — Point shapefile: locations and names of major cities and populated places worldwide

- `usastatelo` — Polygon shapefile: low-resolution outlines and names of the 50 U.S. states plus D.C.

- `usastatehi` — Polygon shapefile: moderate-resolution outlines and names of the 50 U.S. states plus D.C.

- `conus` — MAT-file: Low-resolution latitudes and longitudes, in degrees, for the perimeter of the conterminous United States (CONUS), the Great Lakes, and interstate borders

- `greatlakes` — MAT-file: A Version 1 geographic data structure (geostruct1) with outlines and names for the Great Lakes of North America

Atlas Data MAT-Files Removed in This Release

MAT-files containing Atlas Data have been removed in Version 2.1. Some of the data has been retained in a different form. The disposition of these data sets and variables is described below.

World MAT-File Data

- `worldlo.mat`, which contained the following variables:
  - `DNline` — Data moved to `worldrivers.shp`
  - `DNpatch` — Data moved to `worldlakes.shp`
  - `POline` — Data removed from toolbox
  - `POtext` — Data removed from toolbox
  - `PPpoint` — Data moved to `worldcities.shp`
  - `PPtext` — Data moved to `worldcities.shp`
  - `gazette` — Data removed from toolbox
- `worldhi.mat` — Data removed from toolbox
- `worldmtx.mat` — Data removed from toolbox
- `worldmtxmed.mat` — Data removed from toolbox

United States MAT-File Data

- `usalo.mat`, which contained the following variables (all retained):
  - `conus` — Data moved to `conus.mat`
  - `greatlakes` — Data moved to `greatlakes.mat`
  - `gtlakelat` — Data moved to `conus.mat`
  - `gtlakelon` — Data moved to `conus.mat`
  - `state` — Data moved to `usastatelo.shp`
  - `stateborder` — Data moved to `conus.mat`
  - `statelat` — Data moved to `conus.mat`

- statelon — Data moved to `conus.mat`
- uslat — Data moved to `conus.mat`
- uslon — Data moved to `conus.mat`

- `usahi.mat` — Data moved to `usastatehi.shp`

Functions Deleted in This Release
Compatibility Considerations: Yes

The following functions, which performed specific operations on Atlas Data sets, have been removed in Version 2.1:

- `country2mtx` — Create a raster map grid of a country from `worldlo` atlas data

- `usahi` — Return high-resolution vector atlas data for the United States

- `usalo` — Return vector atlas data for the United States

- `worldhi` — Return high-resolution vector atlas data for the world

- `worldlo` — Return vector atlas data for the world or oceans

Functions Made Obsolete in This Release
Compatibility Considerations: Yes

- etopo5 — Replaced by etopo. Use etopo instead.

- tigerp — Download U.S. Census cartographic boundary files in shapefile format and use shaperead instead.

- tigermif — Download U.S. Census cartographic boundary files in shapefile format and use shaperead instead.