Contents

[Math Functions 2](#_Toc116002334)

[deg2dms 2](#_Toc116002335)

[deg2hms 2](#_Toc116002336)

[R1 2](#_Toc116002337)

[R2 2](#_Toc116002338)

[R3 2](#_Toc116002339)

[Time Functions 2](#_Toc116002340)

[greg2jd 2](#_Toc116002341)

[jd2greg 2](#_Toc116002342)

[Coordinate Transformations 2](#_Toc116002343)

[ECEF2LLA 2](#_Toc116002344)

[ECI2ECEF 2](#_Toc116002345)

[ECI2RaDec 2](#_Toc116002346)

[ENU2AzEl 2](#_Toc116002347)

# Math Functions

## deg2dms

Convert an angle measured in degrees to degrees, arc minutes, and arc seconds.

**Syntax:**

[d, m, s] = deg2hms(deg)

**Inputs:**

deg - angle in degrees

**Outputs:**

d - degrees

m - minutes (1/60)

s - arc seconds (1/3600)

**Notes:**

## deg2hms

Convert an angle measured in degrees to hours, minutes, and seconds.

**Syntax:**

[h, m, s] = deg2hms(deg)

**Inputs:**

deg - angle in degrees

**Outputs:**

h - hours (15)

m - minutes (1/60)

s - seconds (1/3600)

**Notes:**

## R1

Create a counterclockwise rotation about the X-axis

**Syntax:**

[rot] = R1(theta)

**Inputs:**

theta - rotation angle (radians)

**Outputs:**

rot - rotation matrix

**Notes:**

## R2

Create a counterclockwise rotation about the Y-axis

**Syntax:**

[rot] = R2(theta)

**Inputs:**

theta - rotation angle (radians)

**Outputs:**

rot - rotation matrix

**Notes:**

## R3

Create a counterclockwise rotation about the Z-axis

**Syntax:**

[rot] = R3(theta)

**Inputs:**

theta - rotation angle (radians)

**Outputs:**

rot - rotation matrix

**Notes:**

# Time Functions

## greg2jd

Convert a Gregorian calendar date to a Julian date. Good for **XXX to XXX**.

**Syntax:**

[jd] = greg2jd(year, month, day, hour, min, sec)

**Inputs:**

year - Gregorian calendar year

month - Gregorian calendar month

day - Gregorian calendar day

hour - UTC hour

min - UTC minute

sec - UTC second

**Outputs:**

jd - Julian date

**Notes:**

## jd2greg

Convert a Julian date and UTC day fraction to a Gregorian calendar date

**Syntax:**

[y, m, d, h, minute, s, str] = jd2greg(jd)

**Inputs:**

jd - Julian date: day number plus UTC day fraction

**Outputs:**

y - Gregorian calendar year

m - Gregorian calendar month

d - Gregorian calendar day

h - UTC hour

mintute - UTC minute

s - UTC second

str - String representing date/time: DD Mon YYYY

hh:mm:ss.sss UTC

**Notes:**

Currently does not work for the year 2000 or on days close to the start/end of a month. Going to troubleshoot this

# Coordinate Transformations

## ECEF2LLA

Convert a vector in the ECEF frame to latitude, longitude, and altitude. Options for spherical Earth model and ellipsoidal Earth model.

**Syntax:**

[lat, lon, alt] = ECEF2LLA(r, opt)

**Inputs:**

r - position vector in the ECEF frame

opt - geocentric/geodetic reduction option string

**Outputs:**

lat - geocentric or geodetic latitude (deg)

lon - geocentric or geodetic longitude (deg)

alt - geocentric or geodetic altitude (m)

**Notes:**

Options for the algorithm are as follows:

‘gc’ - geocentric conversion, spherical earth:

‘gd’ - geodetic conversion, JGM-3 model:

. Follows algorithm 12 in Vallado.

## ECI2ECEF

## ECI2RaDec

## ENU2AzEl