Problem 1

```
function f = ffunctgravgradsc02(t,x,IMoIbody,norbit)
%
%  Copyright (c) 2019 Mark L. Psiaki.  All rights reserved.
%
%  This function implements a nonlinear dynamic model
%  of a rigid-body spacecraft that includes the
%  gravity-gradient torques produced by a 1/r^2
%  gravity field for a spacecraft that is flying
%  in a circular orbit.  This particular model
%  uses quaternion to parameterize the transformation
%  from local-level orbit following coordinates
%  to the body-fixed coordinates in which
%  the moment-of-inertia matrix IMoIbody is
%  defined so that the transformation from
%  local-level coordinates to body-fixed coordinates
%  is defined by the orthonormal rotation matrix:
%
%    R = R(q)
%
%
%  Inputs:
%
%    t                      The time, in seconds, at which f is
%                           to be computed.
%
%    x                      = [q1;q2;q3;q4;omegabody1;omegabody2;
%                           omegabody3], the 7-by-1 state
%                           vector of this system.  The first four
%                           elements give the non-dimensional unit-
%                           normalized attitude quaternion for the
%                           rotation from local-level coordinates
%                           to spacecraft body coordinates.  The last
%                           three elements give the body rotation rate
%                           vector relative to inertial coordinates
%                           in radians/second and expressed in
%                           the same body-fixed axes in which
%                           IMoIbody is defined.
%
%    IMoIbody               The 3-by-3 symmetric spacecraft
%                           moment-of-inertia matrix about the
%                           spacecraft center of mass and given
%                           in body-fixed coordinates in units
%                           of kg-m^2.
%
%    norbit                 The orbital motion rate in
%                           radians/sec.  This is known as
%                           the mean motion in Keplerian
%                           orbital dynamics parlance.
%
%  Outputs:
```

```
%
%    f                      =
 [q1dot;q2dot;q3dot;q4dot;omegabody1dot;...
%                           omegabody2dot;omegabody3dot],
%                           the 7-by-1 vector that contains the
%                           computed time derivatives of the state
%                           from the kinematics and dynamics models
%                           of the spacecraft.  f(1:4,1) is given
%                           in 1/second units.  f(5:7,1) is given in
%                           radians/second^2.
%

%
%  Determine the rotation matrix from local-level coordinates
%  to body-fixed coordinates.
%
   q = x(1:4);
   qnorm = q*(1/sqrt(sum(q.^2)));
   R = rotmatquaternion(qnorm);
%
%  Determine the rotation rate of the body-fixed coordinates
%  relative to local-level coordinates and given along
%  body axes.
%
   omegavec = x(5:7);
   deltaomegavec = omegavec - R*[0;-norbit;0];
%
%  Determine the quaternion time rate of change from
%  the quaternion kinematics model.
%
   Omegamat = [0 deltaomegavec(3) -deltaomegavec(2)
 deltaomegavec(1);...
                -deltaomegavec(3) 0 deltaomegavec(1)
 deltaomegavec(2);...
                deltaomegavec(2) -deltaomegavec(1) 0
 deltaomegavec(3);...
                -deltaomegavec(1) -deltaomegavec(2) -deltaomegavec(3)
 0];
   qdot = 0.5*Omegamat*q;
%
%  Compute the unit direction vector from the Earth to
%  the spacecraft and given in spacecraft body coordinates:
%
   rhatcmvec = R*[0 0 -1]';
%
%  Compute the gravity-gradient torque in body coordinates.
%
   IMoI_rhatcmvec = IMoIbody*rhatcmvec;
   Tgravgradvec = 3*(norbit^2)*cross(rhatcmvec,IMoI_rhatcmvec);
%
%  Compute the angular velocity rate using Euler's equation.
%
   hvec = IMoIbody*omegavec;
   omegavecdot = IMoIbody\(cross(-omegavec,hvec)+Tgravgradvec);
```

```
%
%  Assemble the computed state time derivative elements
%  into the output vector.
%
   f = [qdot;omegavecdot];
```

Published with MATLAB® R2019a