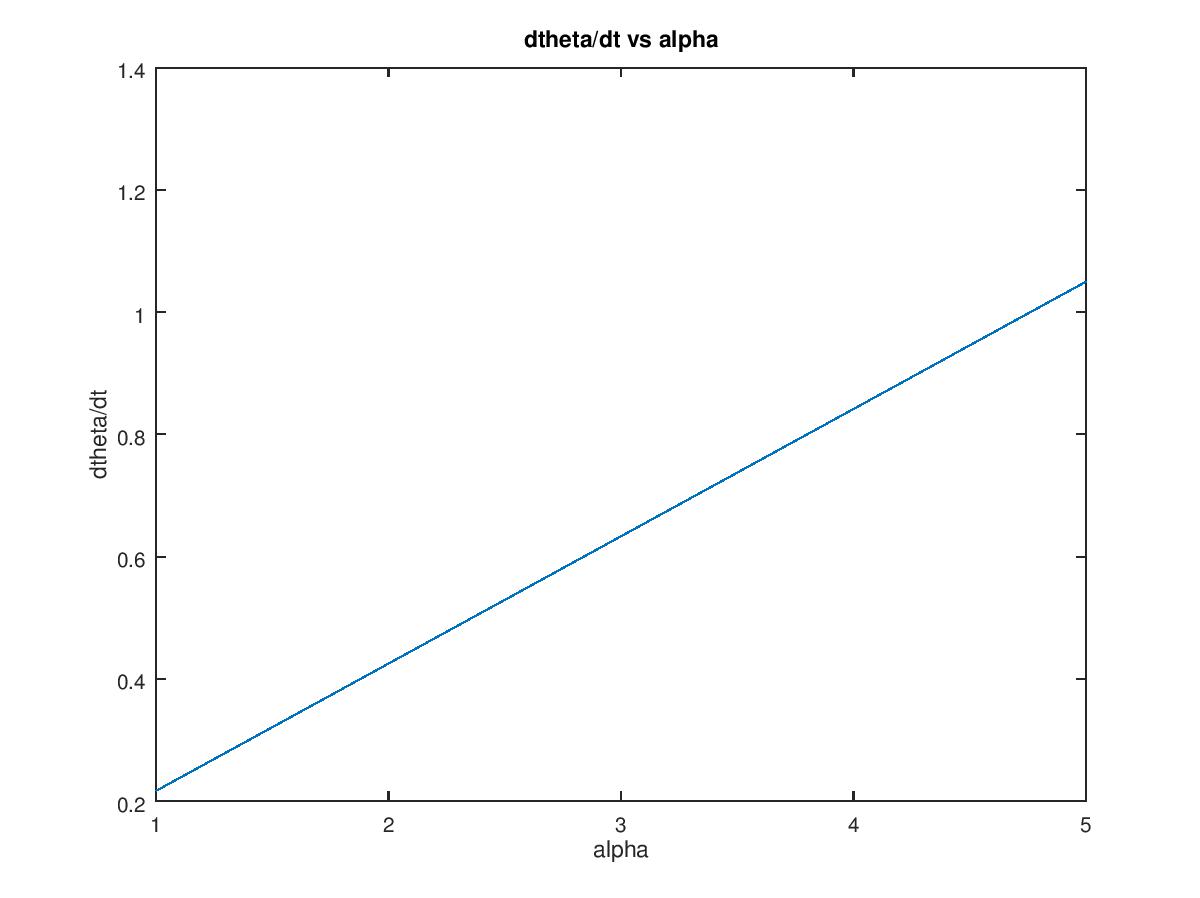
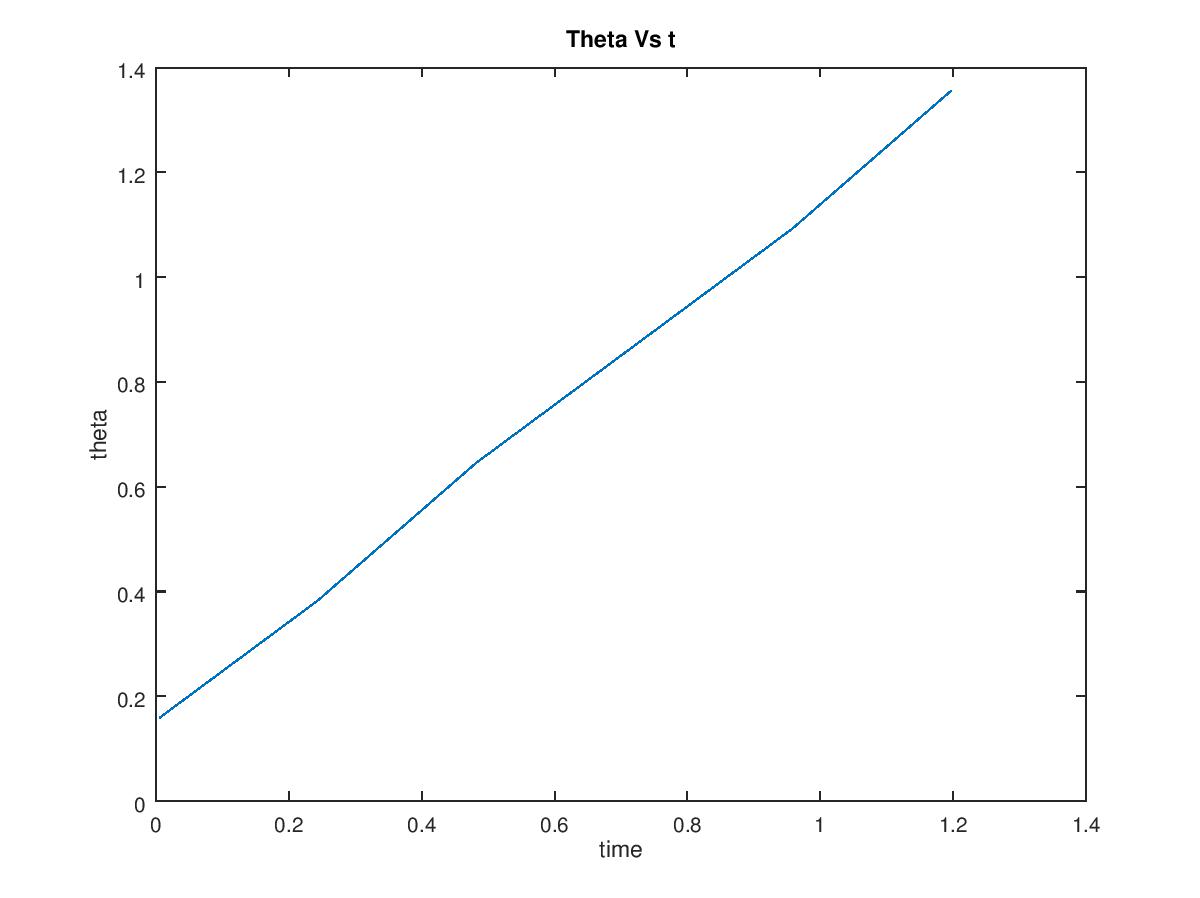
Jonathan Shipley

Scientific Modeling

Perihelion Of Mercury

47.253 arc seconds per century was found to be the final precession of Mercury using its real value of α.





%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Perhelion of Mercury \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% Jonathan Shipley

% Scientific Modeling

% 2/23/17

% This program runs through the orbit of Mercury. It also takes general relativity into account

% and takes the precession into account as well. This program runs through 5 different alpha

% values that are incredibly large compared to the real value. It runs the orbit through and

% ultimately tracks the precession of Mercury in arcseconds per century.

% Important Parameters: Perihelion and aphelion are set. Initial velocity of Mercury is also set

% Input: Alpha changes to get different slopes. 1500 years seems to allow for at least 5

% perihelions to be reached.

% Output: The program outputs the values of alpha and the slope of dtheta/dt.

% The program also keeps track of each theta and time that a perihelion is reached.

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% clear variables from previous runs

clear;

PERIHELION = 0.313; %AU

APHELION = 0.48; %AU

alpha = [0.001, 0.002, 0.003, 0.004, 0.005]

NumAlpha = 5;

dt = 0.001; % years

steps = 1200;

x = PERIHELION; %AU

y = 0; % AU

r = PERIHELION;

vx = 0; %AU/yr

vy = 12.4; %AU/yr

% variables to keep track of perihelions

periCheck = 0; % checks to see if perihelion is hit

p =1;

theta = 0;

rho = 0;

t = 0;

for ai = 1:NumAlpha;

p = 1;

for k=1:steps

r(k) = sqrt(x(k)^2 + y(k)^2);

if k > 1

periCheck(k) = (r(k) - r(k-1))/dt;

if periCheck(k) > 0 && periCheck(k-1) < 0 % if dr/dt at current step is positive but previous step is negative

[theta(p),rho(p)] = cart2pol(x(k), y(k)); % converts coordinates into polar. radians and AU

t(p) = dt \* (k); % keeps track of the time at which the perihelions are hit

p += 1;

end

end

vx(k+1) = vx(k) - ((4\*pi^2\*x(k)\*dt)/(r(k)^3)) \* (1 + (alpha(ai)/(r(k)^2)));

vy(k+1) = vy(k) - ((4\*pi^2\*y(k)\*dt)/(r(k)^3)) \* (1 + (alpha(ai)/(r(k)^2)));

x(k+1) = x(k) + vx(k+1)\*dt;

y(k+1) = y(k) + vy(k+1)\*dt;

end

% Finding slope of the line

N = p;

dthetadt(ai) = (N\*sum(t.\*theta) - sum(theta)\*sum(t))/(N\*sum(t.\*t) - sum(t)^2);

intercept(ai) = (1/N) \* (sum(theta) - dthetadt(ai)\*sum(t));

end

thetaVSalpha = (NumAlpha\*sum(alpha.\*dthetadt) - sum(dthetadt)\*sum(alpha))/(NumAlpha\*sum(alpha.\*alpha) - sum(alpha)^2);

dthetaZERO = (1/NumAlpha) \* (sum(dthetadt) - thetaVSalpha\*sum(alpha));

radianperyear = 1.1\*10^-8 \* thetaVSalpha;

ArcsecPerCentury = radianperyear \* 2.06265\* 10^7 % conversion factor for radianperyear to ArcsecPerCentury

plot(t, theta) % theta vs t

title('Theta Vs t')

xlabel('time')

ylabel('theta')

figure

plot(thetaVSalpha \* alpha + dthetaZERO) % dtheta/dt vs alpha

title('dtheta/dt vs alpha')

xlabel('alpha')

ylabel('dtheta/dt')