```matlab
function [vi, vf, psi] = lambert(ri_vec, rf_vec, dt, DM, Sun, psi_in)
%lambert Solve lambert problem using universal variables method
%   Output initial and final velocities given respective position vectors
%   Inputs in km, Results in km/s
%   DM = Direction of Motion (short way or long way)

fcnPrintQueue(mfilename('fullpath')) % Add this code to code app

tol = 1e-6;

ri = norm(ri_vec);
rf = norm(rf_vec);

cos_df = dot(ri_vec, rf_vec)/(ri*rf);

A = DM*sqrt(ri * rf * (1+cos_df));

if nargin < 6
    psi = 0;
else
    psi = psi_in;
end
c2 = 1/2;
c3 = 1/6;
psi_up_i = 4*pi*pi + psi; % Doubled from Vallado for higher TOF
psi_low_i = -4*pi; % Doubled from Vallado for lower TOF

dt_calc = 0;

first_pass = true;
% while abs(dt_calc-dt) > tol
    if first_pass
        psi_up = psi_up_i;
        psi_low = psi_low_i;
        first_pass = false;
%     elseif psi_up < 0% hit the lower boundary
%         psi_up_i = psi_low_i; % Upper bound is last time's lower bound
%         psi_low_i = 4*psi_low_i; % decrease lower bound
%         psi_up = psi_up_i;
%         psi_low = psi_low_i;
%     elseif psi_low > 0 %hit upper boundary
%         psi_low_i = psi_up_i; % lower bound is last time's upper
%         psi_up_i = 4*psi_up_i; % increase upper
%         psi_up = psi_up_i;
%         psi_low = psi_low_i;
    end


    while abs(dt_calc-dt) > tol
        y = ri + rf + A*(psi*c3-1)/sqrt(c2);
        if A > 0 && y < 0
            while y < 0
```

```
                psi = psi + 0.1;
                y = ri + rf + A*(psi*c3-1)/sqrt(c2);
            end
        end

        X = sqrt(y/c2);
        dt_calc = (X*X*X*c3 + A*sqrt(y))/sqrt(Sun.mu);

%         y_prime = A*(c3-1)/sqrt(c2);
%         psi = psi - y/y_prime;
        if (dt_calc <= dt)
            psi_low = psi;
        else
            psi_up = psi;
        end

%         if abs(psi_up_i - psi_low) < 1e-10 || abs(psi_low_i - psi_up) < 1e-10
%             break; %we hit one of the boundaries
%         end

        psi = (psi_up + psi_low)/2;

        if psi > 1e-6
            c2 = (1-cos(sqrt(psi)))/psi;
            c3 = (sqrt(psi) - sin(sqrt(psi)))/sqrt(psi*psi*psi);
        elseif psi < -1e6
            c2 = (1-cosh(sqrt(psi)))/psi;
            c3 = (sinh(sqrt(-psi)) - sqrt(-psi))/sqrt(-psi*psi*psi);
        else
            c2 = 1/2;
            c3 = 1/6;
        end

        if (psi_up-psi_low) < 1e-10 && abs(dt_calc-dt) > 100
            %Get out of here! fell into a bad minimum.
            fprintf('Lamber solver fell into a bad minimum, returning.\n')
            fprintf('psi = %.3f\n',psi)
            psi = 0;
            vi = zeros(3,1);
            vf = zeros(3,1);
            return
        end
    end

    f = 1-y/ri;
    g_dot = 1-y/rf;
    g = A*sqrt(y/Sun.mu);

    vi = (rf_vec-f*ri_vec)/g;
    vf = (g_dot*rf_vec-ri_vec)/g;
% end


end
```

Published with MATLAB® R2013b