Table of Contents

Basics

```
clc;
clear all;
close all;
```

Changeable paramteres

```
global v r_min r_max;
v = 1;
r_min = 20;
r_max = 40;
```

Global variables

```
global count flag1 flag2 sp;
count = 0;
flag1 = 0;
flag2 = 1;
```

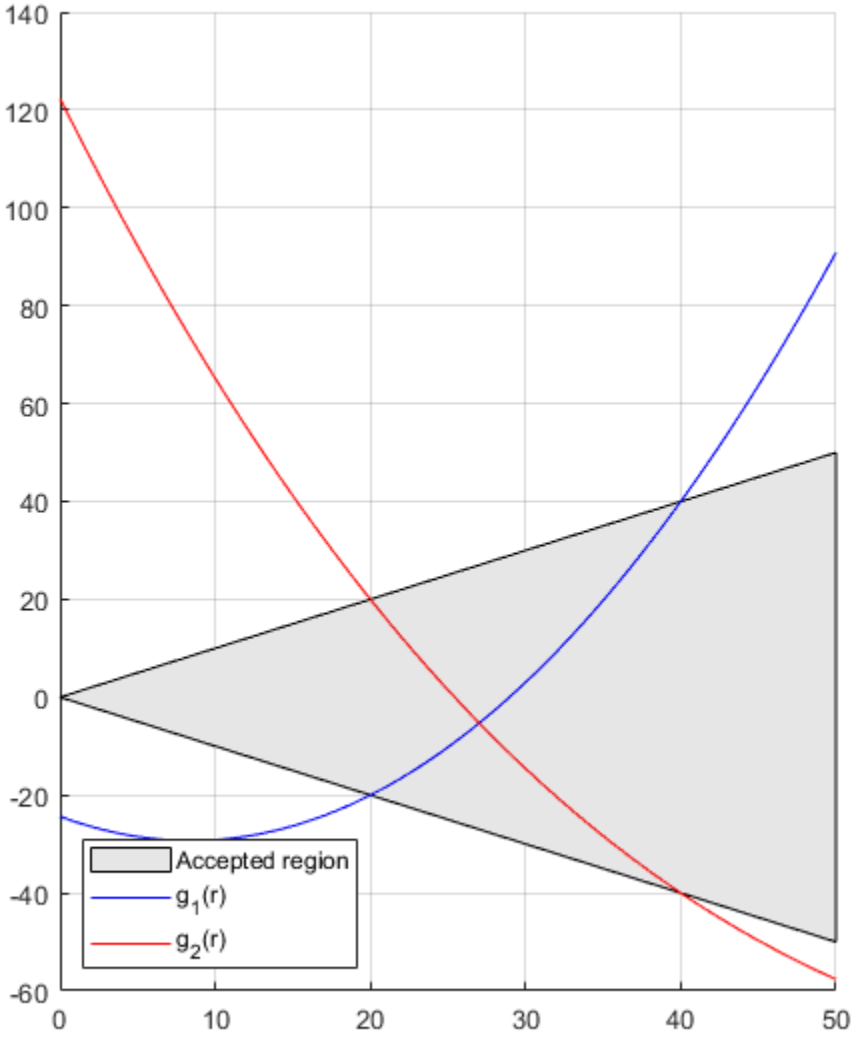
Symbolic functions and variables

```
global r g1 g2 f1 f2;
syms r g1(r) g2(r) f1(r) f2(r);
g1(r) = generate_function();
g2(r) = -1 * generate_function();
f1(r) = diff(g1, r) / r;
f2(r) = diff(g2, r) / r;
```

Plotting the generating function

```
r_range = linspace(0, r_max+10);
max_range = v * r_range;
min_range = -1 * v * r_range;
figure;
hold on;
```

```
x_ = [r_range, fliplr(r_range)];
inBetween = [max_range, fliplr(min_range)];
fill(x_, inBetween, [.9, .9, .9])
plot(r_range, double(g1(r_range)), 'b');
plot(r_range, double(g2(r_range)), 'r');
legend(["Accepted region", "g_{1}(r)", "g_{2}(r)"], 'Location', 'southwest');
grid on;
hold off;
set(gcf, 'Position', [100 100 500 600])
```



Solving for switching point

```
eqn = g1(r) == g2(r);
sp = double(vpasolve(eqn, r, [r_min r_max]));
```

Initial Conditions

```
r0 = r_min;
phi0 = 3 * pi / 2;
alpha0 = 0;
theta0 = 0;
options = odeset('RelTol', 1e-7, 'AbsTol', 1e-7);
```

Trajectory Generation

```
[t1, Y1] = ode23(@(t1, Y1) odefunc(t1, Y1), [0, 1000], [r0 phi0 alpha0
 theta0], options);
[x1, y1] = pol2cart(Y1(:, 4), Y1(:, 1));
plot_graph(x1, y1);
```

Function to generate generating functions

```
function ret = generate_function()
   global r r_min r_max v;

   n = rand();
   if (n < 0.1)
        % Linear Function generation - working
        m = v * (r_max+r_min) / (r_max-r_min);
        c = (v-m) * r_max;
        ret = m*r + c;
   elseif (n < 0.4)
        % Quadratic Function generation - working
        r1 = 0;
        if (n < 0.25)
            r1 = rand() * r_min;
        else
            r1 = r_max * (rand()+1.5);
        end
        a = v * (r_max+r_min) / ((r_max-r_min) * ((r_max+r_min)/2 - r1));
        b = v*r_max + a*r1*r_max - a*r_max*r_max/2;
        ret = a*r*r/2 - a*r1*r + b;
   else
        % Cubic Function generation - working
        r1 = 0;
        r2 = 0;
        if (n < 0.6)
            r1 = -1 * rand() * r_max;
            r2 = rand() * r_min;
        elseif (n < 0.8)
            r1 = rand() * r_min;
            r2 = r_max * (rand()+1.5);
        else
            r1 = r_max * (rand()+1.5);
            r2 = r_max * (rand()+1.5);
        end
```

```
        a = v * (r_max+r_min) / ((r_max-r_min) *
 ((r_max^2+r_min^2+r_max*r_min)/3 + r1*r2 - (r1+r2)*(r_max+r_min)/2));
        b = v*r_max - (a/3)*(r_max^3) + (a*(r1+r2)/2)*(r_max^2) -
 a*r1*r2*r_max;
        ret = a*r*r*r/3 - a*(r1+r2)*r*r/2 + a*r1*r2*r + b;
    end
end
```

ODE Function

```
function ret = odefunc(t, Y)
    global count flag1 flag2;
    global v sp;
    global f1 f2;

    f_ = 0;
    if (flag2 == 1)
        m = mod(count, 2);
        if (m == 0)
            if (Y(1) >= sp)
                count = count + 1;
                if (mod(count, 6) == 1)
                    flag1 = 1 - flag1;
                    flag2 = 0;
                end
            end
        elseif (m == 1)
            if (Y(1) <= sp)
                count = count + 1;
                if (mod(count, 6) == 1)
                    flag1 = 1 - flag1;
                    flag2 = 0;
                end
            end
        end
    end

    if (flag2 == 0)
        if (Y(1)>=(sp+2) || Y(1)<=(sp-2))
            flag2 = 1;
        end
    end

    if (flag1 == 0)
        f_ = double(f2(Y(1)));
    elseif (flag1 == 1)
        f_ = double(f1(Y(1)));
    end

    ret(1, 1) = -1 * v * cos(Y(2));          % r' equation
    ret(2, 1) = f_ + (v/Y(1))*sin(Y(2));     % phi' equation
    ret(3, 1) = f_;                          % alpha' equation
    ret(4, 1) = -1*(v/Y(1))*sin(Y(2));       % theta' equation
```

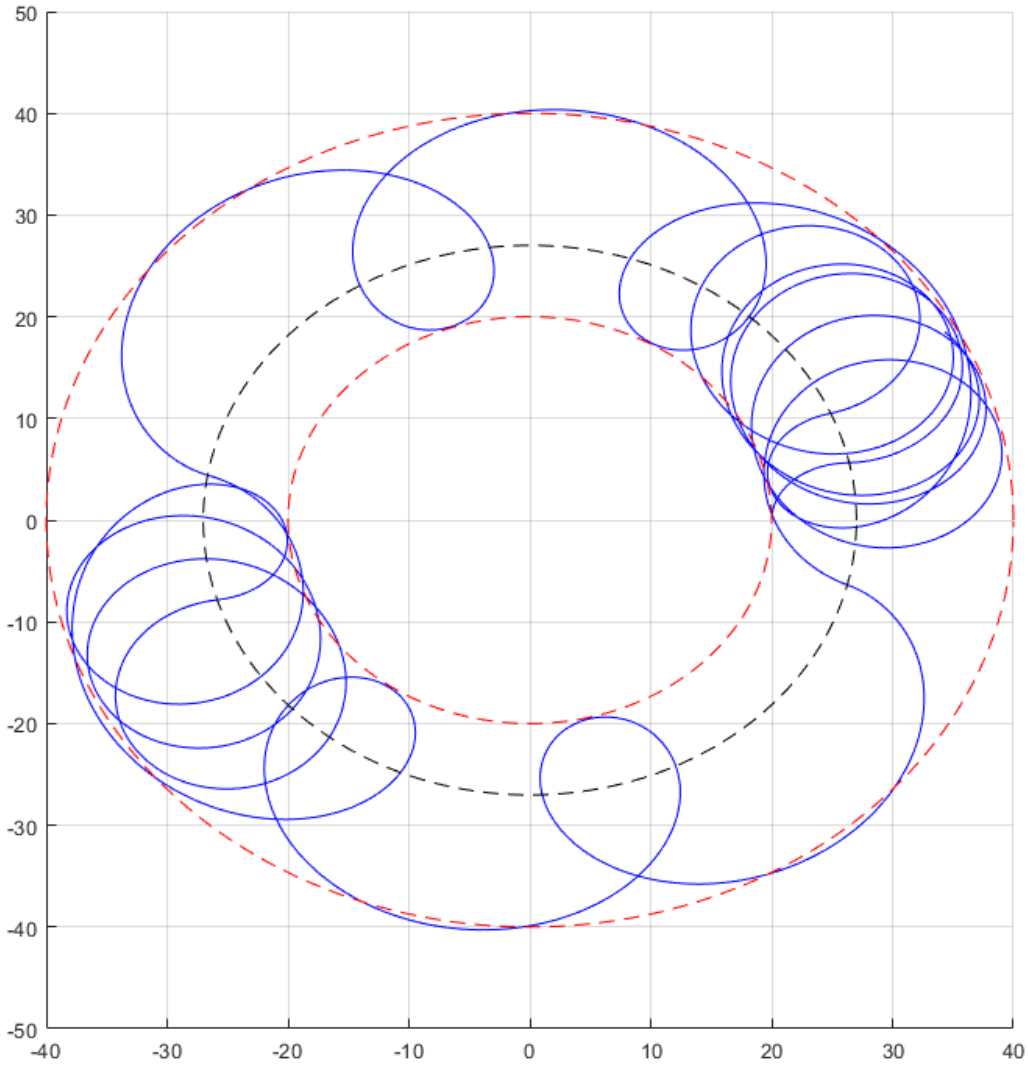
```
end
```

Graph plotting function

```
function plot_graph(x, y)
    global r_min r_max sp;

    angle = 0:0.01:2*pi;

    figure;
    hold on;
    plot(x, y, 'b');
    plot(r_min*cos(angle), r_min*sin(angle), '--r');
    plot(r_max*cos(angle), r_max*sin(angle), '--r');
    plot(sp*cos(angle), sp*sin(angle), '--black');
    set(gcf, 'Position', [1000 100 800 800]);
    grid on;
    hold off;
end
```



Published with MATLAB® R2021b