CS 370 Winter 2009: Assignment 2

Instructor: G. Labahn Office: DC3629 e-mail: *glabahn@uwaterloo.ca*
Lectures: MWF 9:30 MC2038 Office Hours: Tues 2:30-3:30 PM
Office Hours: Thurs 9:30-10:30 AM

Web Site: www.cs.uwaterloo.ca/~glabahn/cs370/
Due Friday Feb 13, 5:00PM, in assignment boxes, 3rd floor MC

1 Analytic Problems

1. (10 marks) The suspension system in a car can be modelled by the system of differential equations for $x_1(t), x_2(t)$

$$\begin{aligned} x_1''(t) + c_1(x_1'(t) - x_2'(t)) + k_1(x_1(t) - x_2(t)) + k_2 x_1(t) &= sin(t) \\ x_2''(t) + c_2(x_1(t) - x_2(t)) - c_1(x_1'(t) - x_2'(t)) &= 0 \end{aligned} \tag{1}$$

with initial conditions

$$x_1(0) = x_2(0) = x_1'(0) = x_2'(0) = 0 \tag{2}$$

and c_1, c_2, k_1, k_2 constants.

(a) Write system (1) as a system of first order equations.

(b) Put the above system into the form

$$\mathbf{y}'(t) = \mathbf{f}(t, \mathbf{y}(t)) \tag{3}$$

where $\mathbf{y}, \mathbf{f}$ are vectors. Write out the function $\mathbf{f}$ explicitly.

2. (15 marks) In order to compute the solution to

$$y'(t) = f(t, y) \tag{4}$$

we use

$$y_{n+1} = y_n + \frac{h}{4}\left[f(t_n, y_n) + 3f\left(t_n + \frac{2h}{3}, y_n + \frac{2}{3}hf(t_n, y_n)\right)\right] \tag{5}$$

where y_n is an approximation to $y(t_n)$, and $h = t_{n+1} - t_n$. Determine the stability condition for this method for the test equation where

$$f(t, y) = -\lambda y \ ; \ \lambda > 0 \ . \tag{6}$$

Show all details for determining the final stability condition.

3. (15 marks) A method is proposed for solving

$$y'(t) = f(t, y) \tag{7}$$

of the form

$$\frac{y_{n+1} - y_{n-1}}{2h} = f(t_n, y_n) \tag{8}$$

where where y_n is an approximation to $y(t_n)$, and $h = t_{n+1} - t_n = t_n - t_{n-1}$. This is an explicit multistep method.

(a) What is the local error of this method?

(b) Let $f(t, y) = -\lambda y$, $\lambda > 0$, and determine the stability condition for this method.

2 Programming Problems

IMPORTANT: In this and in future assignments, most of the marks for programming questions are allocated for explanations of algorithms (i.e. pseudo-code) and explanation of results. If all you hand in is the listing of the "Raw Code" or "Raw Output" by itself, you will get poor marks.

Pursuit Problem

Consider a target moving in a two dimensional plane with location (x_T, y_T). The target is moving with constant speed in direction given by the unit vector $(\cos\theta_T, \sin\theta_T)$, where θ_T is the angle of the velocity vector with respect to the x axis.

The target is pursued by a pursuer, with location (x_p, y_p). The pursuer is moving with constant speed twice that of the target, in direction given by the unit vector $(\cos\theta_p, \sin\theta_p)$, where θ_p is the angle of the velocity vector with respect to the x axis.

The pursuer attempts to hit the target, by turning as fast as possible towards the target. The minimum turning radius of the pursuer is R_p, and the minimum turning radius of the target is R_T.

The trajectories of the pursuer and target can be described by the following system of ODEs, where t is the time.

$$\begin{aligned} x_p'(t) &= 2\cos\theta_p(t), \qquad y_p'(t) = 2\sin\theta_p(t), \qquad \theta_p'(t) = \frac{2}{R_p}\frac{\theta_d(t)-\theta_p(t)}{(|\theta_d(t)-\theta_p(t)|+0.002)} \\ x_T'(t) &= \cos\theta_T(t), \qquad y_T(t) = \sin\theta_T(t), \qquad \theta_T(t) = \frac{1}{R_T}\omega_T(t)\ . \end{aligned} \tag{9}$$

where

$$\begin{aligned} (x_p, y_p) &= \texttt{x, y coordinates of pursuer} \\ \theta_p &= \texttt{pursuer moving in direction } \cos\theta_p\hat{x} + \sin\theta_p\hat{y} \\ R_p &= \texttt{minimum turning radius, pursuer} \\ (x_T, y_T) &= \texttt{x, y coordinates of target} \\ \theta_T &= \texttt{target moving in direction } \cos\theta_T\hat{x} + \sin\theta_T\hat{y} \\ R_T &= \texttt{minimum turning radius, target} \\ \theta_d &= \texttt{direction from pursuer to target} \\ &\quad\ \texttt{(see Figure 1)} \\ \omega_T &= \texttt{angular speed of target} \\ &\quad\ (\texttt{fraction of maximum } \frac{1}{R_T}) \end{aligned} \tag{10}$$

The constant 0.002 in equation (9) is a damping factor which prevents the pursuer direction from changing too rapidly when $\theta_d \simeq \theta_p$.

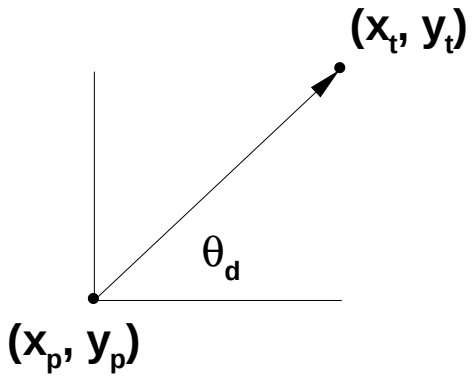

Figure 1: Direction from pursuer to target.

Parameter	Value
Initial Target Position	$(x_T, y_T) = (5, 5)$
Initial Target Direction	$\theta_T = 0.0$
Initial Pursuer Position	$(x_p, y_p) = (0, 0)$
Initial Pursuer Direction	$\theta_p = -\pi$
Integration Interval	$[0, 20]$
R_p	1.0
R_T	.25
Hitting Distance	$d_{stop} = 10^{-2}$
Angular Speed of Target	$\omega_T = 0.1$
Error Tolerance	$tol = 10^{-6}$

Table 1: Data for pursuit problem.

Objective

The objective of this assignment is to develop Matlab code which solves the pursuit-target problem, and determines the time (if it occurs) when the pursuer hits the target. The pursuer is considered to have hit the target if the distance between the target and the pursuer is less than d_{stop}, that is,

$$\sqrt{(x_T - x_p)^2 + (y_T - y_p)^2} \leq d_{stop} \ . \tag{11}$$

The data for this problem is given in Table 1.

4. (20 marks) Write Matlab code to solve the ODE system (9). Use an event function (see course notes) to stop the simulation if the hitting criterion (11) is satisfied. In order to determine the direction θ_d to the target (equation (9)), we have to be careful, since θ_d, θ_p can be less than or larger than 2π (the pursuer/target could wind around a central point several times). As well, we want to make sure the pursuer turns towards the target in the shortest way possible. Use the supplied function *direction.m* (available on the course web site) to compute θ_d.

Use the stiff solver *ode*15*s*. Use the following options to be passed to the ODE solver (see the course notes).

```
options = odeset('AbsTol', tol,'RelTol',tol,'MaxOrder',5,'Stats','on',...
              'Events',@event, 'Refine',4);
```

Submit a hard copy of your code, a plot of the trajectories of the pursuer and target, and the hitting time. The trajectory of the pursurer is given by the the parametric curve $(x_p(t), y_p(t))$, and the trajectory of the target is given by the parametric curve $(x_T(t), y_T(t))$ for $t \in [0, T]$ where T is the stopping time or end of the integration interval (whichever is first).

5. (10 marks) Now, carry out some tests to see the effect of the error control *tol* on the solution accuracy. Fill in Table 2.

tol	Hitting Time	Number of Function Evaluations
10^{-3}		
10^{-4}		
10^{-5}		
10^{-6}		
10^{-7}		
10^{-8}		
10^{-9}		

Table 2: Test of error control

Explain what you see.

6. (10 marks) Now, try solving the problem again, except use *ode*45 as the solver. Use $tol = 10^{-5}, 10^{-6}, 10^{-7}$. Compare with the solution using *ode*15*s*. Explain what you see.

7. (10 marks) Attempt to devise an evasion strategy for the target. The target is constrained so that its speed is half the speed of the pursuer and the $\omega_T \leq 1$. Some ideas: when the distance between the target and the pursuer is less than a small multiple of R_T, try turning the target in a direction away from the pursuer. Increase the maximum integration interval to $[0, 50]$.

Submit a pseudo-code description of your evasion strategy, a hard copy of your matlab code, and a plot of the target and pursuer trajectories.

The five most interesting strategies (as determined by the TAs), will earn 3 bonus marks. In the case of disputes, the TAs decisions are final.