

Math 449 Fall 2006 - Final Exam

Solutions

1. Consider the differential equation

$$\theta'' + \sin(\theta) = \delta S(t) - \gamma \theta'.$$

We first assume that $\delta = \gamma = 0$ (no friction and no forcing term.)

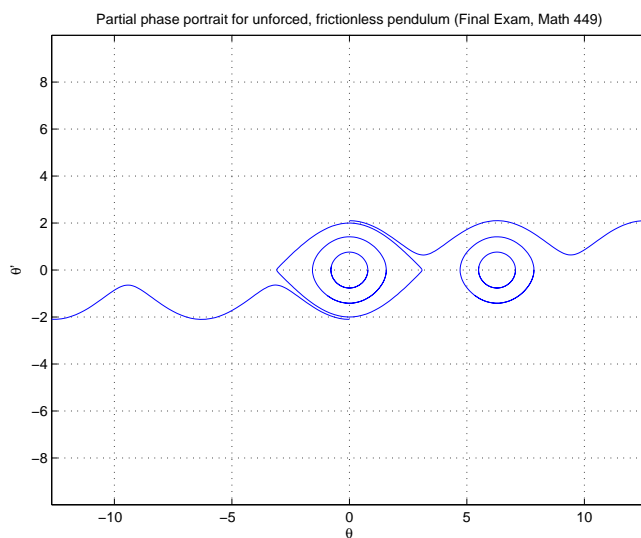


Figure 1: Problem 1 (a)

The initial conditions $(\pi/4, 0)$, $(\pi/2, 0)$ (the two concentric, approximately circular trajectories with center $(0, 0)$) describe oscillations about the resting position $\theta = 0$. The trajectories with initial conditions $(9\pi/4, 0)$, $(5\pi/2, 0)$ are the translates of the previous two by 2π , so they describe the same motion. The two wavy trajectories correspond to initial conditions $(0, \pm 2.1)$. Physically, they describe a motion with enough energy to go all around the pivot, rather than oscillate back and forth. The initial conditions $(\pm(\pi - 0.05), 0)$ are close to the unstable equilibrium point $(\pi, 0)$. When the pendulum begins there, it swings almost a complete turn. The whole *phase portrait* of the system (only part of which is shown in the picture) is periodic under horizontal translation by 2π .

Now, we fix the initial condition to be $(0, 1)$ and vary the parameters δ and γ . Draw separate graphs for the angular velocity θ' as function of time over the time interval $[0, 100]$ for the following parameter values: $(\delta, \gamma) = (0.1, 0.1), (0.8, 0.1), (1.5, 0.1)$. (Suggested step size: $h = 0.1$.) Note that the pendulum behaves more erratically as the torque parameter increases.

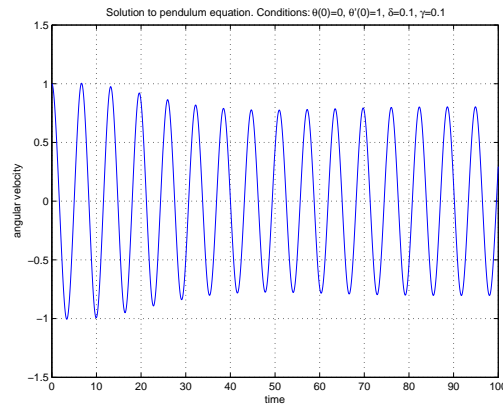


Figure 2: Problem 1 (c); small value for the forcing term. ($\delta = 0.1$)

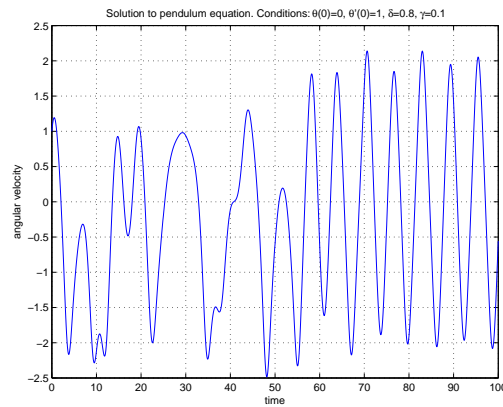


Figure 3: Problem 1 (c); $\delta = 0.8$

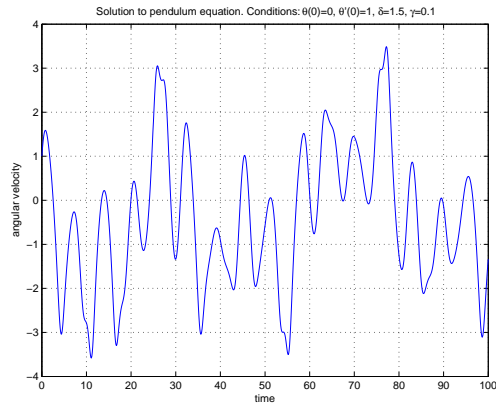


Figure 4: Problem 1 (c); $\delta = 1.5$

2. Lorentz system: $x' = 10(y - x)$, $y' = 28x - y - xz$, $z' = xy - (8/3)z$.

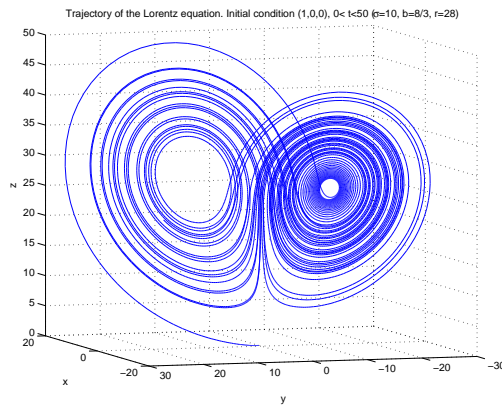


Figure 5: Problem 2. Parametric solution curve for the Lorentz system with initial condition $(1, 0, 0)$ over the time interval $[0, 50]$.

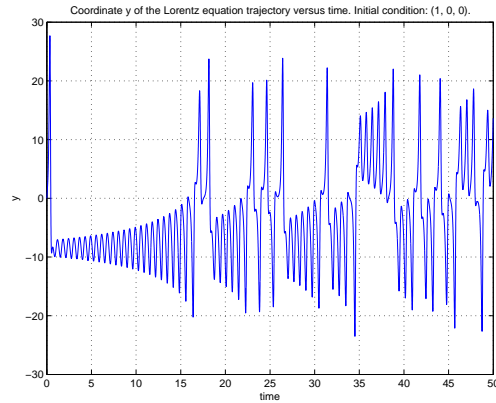


Figure 6: Same solution as in the previous figure, now showing y as a function of t .

3. Linear wave equation (Algorithms and Problems 8, 10.1, page 556.)

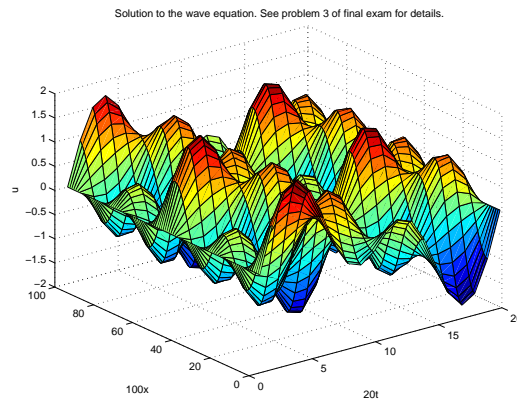


Figure 7: Problem 3.

The above graph is for the approximate solution to the equation $u_{tt}(x, t) = 4u_{xx}(x, t)$ for $0 \leq x \leq 1$ and $0 \leq t \leq 1$, with the following boundary conditions:

$$\begin{aligned}
 u(0, t) &= 0 && \text{for } 0 \leq t \leq 1 \\
 u(a, t) &= 0 && \text{for } 0 \leq t \leq 1 \\
 u(x, 0) &= \sin(2\pi x) + \sin(4\pi x) && \text{for } 0 \leq x \leq 1 \\
 u_t(x, 0) &= 0 && \text{for } 0 \leq x \leq 1
 \end{aligned}$$

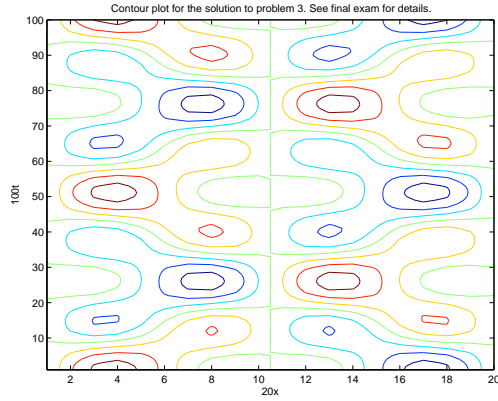


Figure 8: Contour plot for the same approximate solution.

4. The linear diffusion equation. (Algorithms and Programs 3, page 567.)

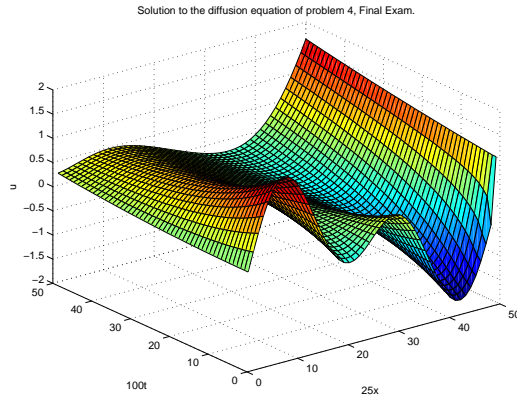


Figure 9: Problem 4. Approximate solution to the diffusion equation shown below.

The equation is $u_t(x, t) = c^2 u_{xx}(x, t)$, for $0 < x < 2$ and $0 < t < 0.5$, with the initial condition $u(x, 0) = \sin(\pi x) + \sin(2\pi x)$ for $0 \leq x \leq 2$, and the boundary conditions

$$\begin{aligned} u(0, t) &= t^2 && \text{for } x = 0 \text{ and } 0 \leq t \leq 0.5 \\ u(2, t) &= e^t && \text{for } x = 2 \text{ and } 0 \leq t \leq 0.5. \end{aligned}$$

The modified `crnich` program (original program: 10.3, page 565) needed

to accept the boundary conditions $u(0, t) = g_1(t)$ and $u(a, t) = g_2(t)$ is given on a separate page.

Matlab programs used in the Final Exam - Math 449 Fall 2006.

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [T,Z]=rks4(F,a,b,Za,M)
%Input - F is the system input as a string 'F'
%       - a and b are the endpoints of the interval
%       - Za=[x1(a) ... xn(a)] are the initial conditions
%       - M is the number of steps
%Output - T is vector of steps
%       - Z=[x1(t) ... xn(t)], where xk(t) is the approximation
%         to the kth dependent variable.
h=(b-a)/M;
T=a:h:b;
Z=zeros(M+1,length(Za));
Z(1,:)=Za;
for j=1:M
    k1=h*feval(F,T(j),Z(j,:));
    k2=h*feval(F,T(j)+h/2,Z(j,:)+k1/2);
    k3=h*feval(F,T(j)+h/2,Z(j,:)+k2/2);
    k4=h*feval(F,T(j)+h,Z(j,:)+k3);
    Z(j+1,:)=Z(j,:)+(k1+2*k2+2*k3+k4)/6;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function U=g(t,Z)
a=1.5;
b=0.1;
c=1;
U=[Z(2), -sin(Z(1)) + a*cos(c*t) - b*Z(2)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function U=lorentz(t,Z)
sig=10;

```

```

r=28;
b=8/3;
U=[sig*(Z(2)-Z(1)), r*Z(1) - Z(2) - Z(1)*Z(3), Z(1)*Z(2) - b*Z(3)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function U = finedif(f,g,a,b,c,n,m)
%Input - f=u(x,0) as a string 'f'
%       - g=u_t(x,0) as a string 'g'
%       - a and b right endpoints of [0,a] and [0,b]
%       - c the constant in the wave equation
%       - n and m number of grid points over [0,a] and [0,b]
%Output - U solution matrix

%Initialize parameters and U
h=a/(n-1);
k=b/(m-1);
r=c*k/h;
r2=r^2;
r22=r^2/2;
s1=1-r^2;
s2=2-2*r^2;
U=zeros(n,m);

%Compute first and second rows
for i=2:n-1
    U(i,1)=feval(f,h*(i-1));
    U(i,2)=s1*feval(f,h*(i-1))+k*feval(g,h*(i-1)) ...
        +r22*(feval(f,h*i)+feval(f,h*(i-2)));
end

%Compute remaining rows of U
for j=3:m,
    for i=2:(n-1),
        U(i,j)=s2*U(i,j-1)+r2*(U(i-1,j-1)+U(i+1,j-1))-U(i,j-2);
    end
end

U=U';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function X = trisys(A,D,C,B)

```

```

% X = trisys(A,D,C,B)
% Solution of a triangular linear system AX = B.
% It is assumed that D and B have dimension n,
% and that A and C have dimension n-1;
% A sub diagonal, input.
% D diagonal vector, input.
% C super diagonal, input.
% B right hand side vector, input.
% X solution vector, output.
n = length(B);
for k = 2:n,
    mult = A(k-1)/D(k-1);
    D(k) = D(k) - mult*C(k-1);
    B(k) = B(k) - mult*B(k-1);
end
X(n) = B(n)/D(n);
for k = (n-1):-1:1,
    X(k) = (B(k) - C(k)*X(k+1))/D(k);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function U=crnich(f,g1,g2,a,b,n,m)
%Input - f=u(x,0) as a string 'f'
%       - g1=u(0,t) and g2=u(a,t) as strings 'g1' and 'g2'
%       - a and b right endpoints of [0,a] (space) and [0,b] (time)
%       - c the constant in the heat equation
%       - n and m number of grid points over [0,a] and [0,b]
%Output - U solution matrix.

%Initialize paramters and U
h=a/(n-1);
k=b/(m-1);
r=1;
s1=2+2/r;
s2=2/r-2;
U=zeros(n,m);

%Boundary conditions
U(1,1:m)=feval(g1,k:k:m*k)';
U(n,1:m)=feval(g2,k:k:m*k)';

%Generate the first row
U(2:n-1,1)=feval(f,h:h:(n-2)*h)';

```

```

%Form the diagonal and off-diagonal elements of A
%and the constant vector B and solve tridiagonal system AX=B
Vd(1,1:n)=s1*ones(1,n);
Vd(1)=1;
Vd(n)=1;
Va=-ones(1,n-1);
Va(n-1)=0;
Vc=-ones(1,n-1);
Vc(1)=0;
for j=2:m
    Vb(1)=feval(g1,(j-1)*k);
    Vb(n)=feval(g2,(j-1)*k);
    for i=2:n-1
        Vb(i)=U(i-1,j-1)+U(i+1,j-1)+s2*U(i,j-1);
    end
    X=trisys(Va,Vd,Vc,Vb);
    U(1:n,j)=X';
end
U=U';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

The previous program is used by calling the command:

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
U=crnich('h','g1','g2',2,0.5,50,50);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

The functions g1, g2, h are:

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function y=g1(x)
y=x.^2;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function y=g2(x)
y=exp(x);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```
function y=h(x)
y=sin(pi*x)+sin(2*pi*x);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```