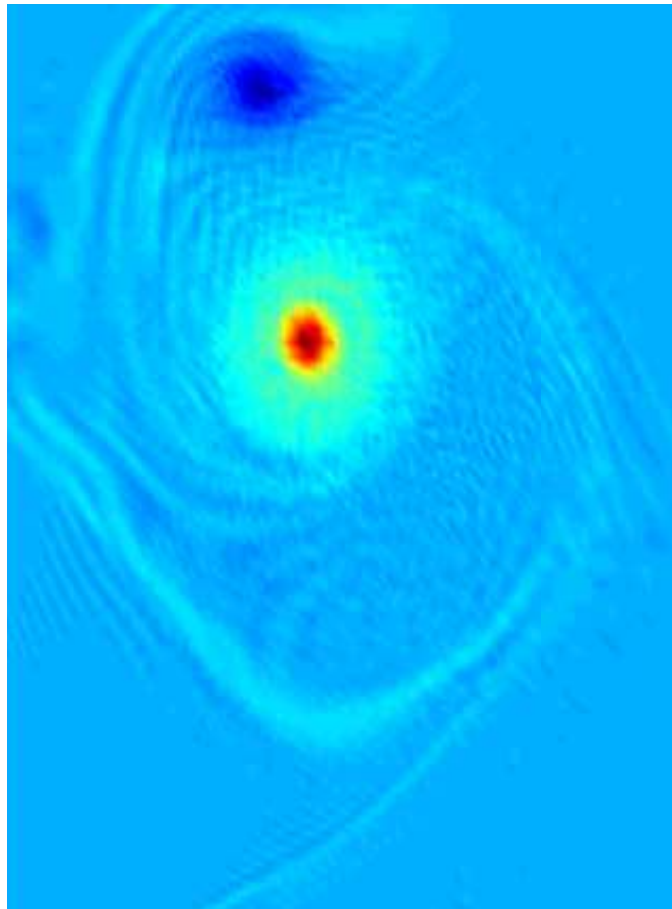# Mastered Learning Objectives
from
Partial Differential Equations

MATH 4900, Fall 2020 with Prof. Emma Zbarsky

Your Name Here

youremailhere@wit.edu

Contents

1 Mastery Check Planning

Standard	Suggested Date	Mastery Date	Writeup Completed	Productive Failure?
NM: Order of Error	Wk2			☐
NM: Explicit Difference	Wk 3			☐
MC: Linear, constant speed	Wk 3			☐
MC: Linear, polynomial speed	Wk 4			☐
MC: Nonlinear	Wk 4			☐
MC: Shock	Wk 5			☐
MC: Rarefaction	Wk 5			☐
NM: Neumann stability	Wk 6			☐
NM: CFL condition	Wk 6			☐
NM: Implicit Difference	Wk 6			☐
FS: Real Fourier Series	Wk 7			☐
FS: Complex Fourier Series	Wk 7			☐
FS: Convergence of Fourier Series	Wk 8			☐
FS: Integrability & Differentiability of FS	Wk 8			☐
FS: Boundary Conditions	Wk 9			☐
SV: Heat Equation	Wk 10			☐
SV: Equilibrium behavior	Wk 10			☐
SV: Wave equation	Wk 11			☐
SV: d'Alembert's equation	Wk 12			☐
SV: Laplace equation	Wk 13			☐
Final Preparations	Wk 14/15			

2 Numerical Methods of Solving Partial Differential Equations

A first order initial value problem takes the form $\dfrac{dy}{dt} = f(y,t)$ with $y(t_0) = y_0$. Such functions are studied in a first class on differential equations in a variety of special cases (e.g. separable equations or linear equations) but there is no general analytic method that works for every f. We shall explore some numerical methods of finding a solution.

2.1 Order of Error

The Euler Method takes the most basic ideas of calculus and uses them to step through time to find an approximate solution.

Differential equation:	$\dfrac{dy}{dt} = f(y,t)$	Initial condition:	$y(t_0) = y_0$
Time interval of solution:	$[t_0, t_1]$	Step size:	Δt
Number of steps:	$\dfrac{t_1 - t_0}{\Delta t}$	Iterative step:	$y(t + \Delta t) = f(y(t), t)\Delta t + y(t)$

2.1.1 Error

The error is of order Δt. In the limit as $\Delta t \to 0$, the error in Euler's method also goes to 0. In practice, it is impractical to let Δt become too small as you reach limits in your computer's ability to resolve numbers. You should check on the *machine epsilon* for your computer and the limitations of the data type you are using. This is also why you always want to add numbers from the smallest to the largest on a computer as well to minimize the loss of precision due to rounding.

2.2 Neumann Stability

2.3 The CFL Condition

2.4 Implementing an Explicit Difference Method

3

of what values must be changed to solve the two different problems. What follows is a sample MATLAB program that I wrote as an example of how to include commented code in a LaTeX file.

```matlab
%% main_program.m
%%    Find the first 5 normalized eigenfunctions for Schrodinger's equation
% %                       using the shooting method
clear variables; close all; clc

tol = 10^(-4);    % set the tolerance to work within

col=['r','b','g','c','m','k','y']; % eigenfunction colors

K = 1;                    % normalize to km/hbar^2 = 1
epsn = .9;                % define the initial parameter epsilon_n
psi=1;                    % define the initial guess for psi
L = 4;                    % range to work in
A = sqrt(L^2-epsn)*psi;   %left boundary differential
x0 = [psi A];             % provide the initial conditions x1(-4) = 1, x1'(-4) = 2
xp = -L:.1:L;             % define the span of the computational domain

eigvalues = zeros(5,1);              % define the results matrices
eigfunctions = zeros(length(xp),5);


epsilon_start = epsn;     % beginning value of epsilon

for modes = 1:5                  % begin mode loop
    epsilon = epsilon_start; % initial value of eigenvalue epsilon
    deps = .05;              % default step size in epsilon
    for j = 1:1000           % begin convergence loop for epsilon
        x0 = [psi sqrt(L^2 - epsilon)*psi]; % Update initial condition for our new epsilon
        [t,y] = ode45(@(t,y) schrodinger(t,y,K,epsilon), xp, x0); % solve ODEs
        % Are we within tolerance?
        if (abs(-sqrt(L^2-epsilon)*y(end,1)-y(end,2))<tol)
            break                    % get out of convergence loop
        end
        % here we check to see if epsilon needs to be higher or lower
        % then we bisect to get closer to our required tolerance
        if (-1)^(modes)*(-sqrt(L^2-epsilon)*y(end,1)-y(end,2))>0
            epsilon = epsilon+deps;
        else
            epsilon = epsilon - deps/2;
            deps = deps/2;
        end                                  % end bisection loop
    end                  % end convergence loop

    eigvalues(modes) = epsilon;
    epsilon_start = epsilon+.1;    % after finding eigenvalue, pick
                                   % new starting value for next higher
                                   % mode
    norm = trapz(t,y(:,1).*y(:,1));    % calculates the normalization
```

```
    if modes > 0
    plot(t,y(:,1)/sqrt(norm),col(modes)); hold on    % plot modes
    end

    eigfunctions(:,modes) = y(:,1)/sqrt(norm);

end         % end modes loop
```

2.5   Implementing an Implicit Difference Method

Explain what an implicit difference method is. Then present the sample problem from the text, with its solution and the sample problem you solved with me for mastery with its solution. Include your code here with clear explanations of what values must be changed to solve the two different problems. What follows is a sample using a block quotation and reference.

J. D. Lambert described the phenomenon of stiffness for a system of differential equations as follows:

> If a numerical method with a finite region of absolute stability, applied to a system with any initial conditions, is forced to use in a certain interval of integration a steplength which is excessively small in relation to the smoothness of the exact solution in that interval, then the system is said to be stiff in that interval. [1]

3   Method of Characteristics

Give an introduction to the method of characteristics.

3.1   Solving $u_t + cu_x + du = g(x)$ with $u(0, x) = f(x)$

Explain how to solve such a problem when c and d are constants. Then present your solution to the mastery problem I gave you.

3.2   Solving the linear transport equation with polynomial coefficients

Explain how to solve such a problem when the wave speed is polynomial in t or polynomial in x. Then demonstrate the solution with the mastery problem I gave you to solve.

3.3   Solving first order nonlinear PDEs using the method of characteristics

Explain how to solve a problem of the form $u_t + g(u)u_x = 0$. Then solve the mastery problem I gave you to solve. What follows is a sample use of equation array and piecewise defined functions.

$$
\begin{aligned}
y_{n+1} &= y_n + e + \Delta t \lambda y_{n+1} \\
(1 - \Delta t \lambda)^n \, y_{n+1} &= y_n + e \\
y_{n+1} &= \left(\frac{1}{1 - \Delta t \lambda} \right)(y_n + e) \\
y_{n+1} &= \left(\frac{1}{1 - \Delta t \lambda} \right)^{n+1}(y_0 + e)
\end{aligned}
$$

and hence a global error after N steps of

$$
E = \left(\frac{1}{1 - \Delta t \lambda} \right)^N e.
$$

Therefore

$$\lim_{N \to \infty} E = \begin{cases} \infty & |1 - \Delta t \lambda| < 1 \\ 0 & |1 - \Delta t \lambda| > 1 \\ e & |1 - \Delta t \lambda| = 1 \end{cases}.$$

3.4 Shocks

Explain how to recognize and solve a first order differential equation whose solution will involve a shock. Then solve the mastery problem I gave you. What follows is sample code for including a figure, Figure 1, with a caption and label.
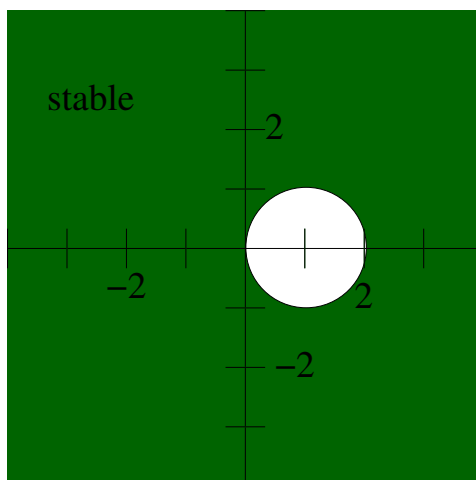

Figure 1: Stable regions for forward (left) and backward (right) Euler methods where green is convergent and white is divergent.

3.5 Rarefaction Waves

Explain how to recognize and solve a first order differential equation that will result in a rarefaction wave. Then solve the mastery problem I gave you. What follows is sample code for a table. I have left some horizontal and vertical lines out for demonstration purposes, and doubled others.

Command	Local Error	Global Error	Scheme
ode23	$O(\Delta t^3)$	$O(\Delta t^2)$	Runge-Kutta
ode45	$O(\Delta t^5)$	$O(\Delta t^4)$	Runge-Kutta
ode15s	variable	variable	Gear's method, stiff
ode113	variable	variable	predictor/corrector

For more information, see the Matlab documentation [2].

4 Fourier Series

Give an introduction to Fourier series.

4.1 Real Fourier Series

Explain how to calculate the real Fourier series for a suitable function. Then present the example that you solved for your mastery check.

4.2 Complex Fourier Series

Explain how to calculate the complex Fourier series for a suitable function. Then present the example that you solved for your mastery check.

4.3 Convergence of Fourier Series

Discuss the convergence of Fourier series. Then present the example that you solved for your mastery check.

4.4 Integrability and Differentiability of Fourier Series

Discuss the integrability and differentiability of Fourier series. Then present the example that you solved for your mastery check.

4.5 Boundary Conditions

Discuss the various standard types of boundary conditions and what conditions tell you about the form of a Fourier Series solution. Then present the example that you solved for your mastery check.

5 Separation of Variables

Give an overview of the method of separation of variables.

5.1 Equilibrium behavior of a solution

Explain what equilibrium behavior is and how one goes about finding the equilibrium behavior of a system. Then give the example that you solved for your mastery check.

5.2 Solving the 1D heat equation

Explain how to solve the 1D heat equation with boundary conditions using the method of separation of variables. Then present the example that you solved for your mastery check.

5.3 Solving the Laplace equation

Explain how to solve the Laplace equation with boundary conditions using the method of separation of variables. Then present the example that you solved for your mastery check.

5.4 Solving the 1D wave equation

Explain how to solve the 1D wave equation with boundary conditions using the method of separation of variables. Then present the example that you solved for your mastery check.

5.5 Solving the 1D wave equation using d'Alembert's formula

Explain how to solve the 1D wave equation with boundary conditions using d'Alembert's formula. Then present the example that you solved for your mastery check.

References

[1] wikipedia.org, *Stiff equation* (11 Dec 2013), available at http://en.wikipedia.org/wiki/Stiff_equation.

[2] mathworks.org, *Matlab R2013b Documentation: Numerical Integration and Differential Equations: Ordinary Differential Equations* (11 Dec 2013), available at http://www.mathworks.com/help/matlab/ordinary-differential-equations.html.