MATH 337.A – Numerical Differential Equations

Spring 2014

HW # 0

Due: 01/17/14

Problem 1 (**0.5 point**)

Find the explicit form of the cubic term (i.e. the term with $(\Delta x)^n(\Delta y)^m$, $n+m=3$) in expansion (0.6).

Problem 2 (**0.5 point**)

Find the Lipschitz constant L for:
(a) $f(x,y) = x\,y^2$ on $R:\ 0 \le x \le 3,\ 1 \le y \le 5$;
(b) $f(x,y) = x + |\sin 2y|$ on $R:\ 0 \le x \le 3,\ -\pi \le y \le \pi$.
Hint: See Remarks after Theorem 0.1.

Problem 3 (**0.5 point**)

Solve the IVP

$$y' = 2y + e^{3x}, \qquad y(-1) = 4\,.$$

Problem 4 (**0.5 point**)

Find

$$\lim_{h\to 0}\left(\frac{1}{1-2h}\right)^{\pi/h}.$$

HW # 1

Due: 01/27/14

Problem 1

Consider an IVP

$$y' = ay, \qquad y(x_0) = y_0\,,$$

where a is a constant. Following the lines of the derivation of the global error (see Eq. (1.4)), find how the *sign* of the global error will depend on the signs of a and y_0.
Note 1: If the signs of the local truncation error at each iteration are the same, then the sign of the global error will be the same as the sign of the local truncation errors[1].
Note 2: For $f(x,y) = ay$, one can do more explicit calculations than in Sec. 1.3, and it *will* be possible to establish the sign of the error.

Verify your answer by solving the above IVP for $x \in [0,\,1]$ [2] using the simple Euler method in four cases: (i) $a=1$, y_0=1; (ii) $a=1$, $y_0=-1$; (iii) $a=-1$, y_0=1; (iv) $a=-1$, $y_0=-1$. Use $h=0.1$.

Note 3: An example of a code using the simple Euler method is posted as `hw1_EX_Euler_direct.m` on the course website under the link "Codes for examples and selected homework problems". Note, however, that it uses a somewhat different function than in this problem.

Problem 2

Find coefficient B for the Midpoint method (1.19). Follow the lines of the derivation of coefficient A for the Modified Euler method.

Problem 3

Derive the expression for the local truncation error of the Midpoint method. Follow the lines of the similar derivation for the Modified Euler method.

Problem 4 (**1.5 points**)

The exact (and unique) solution of the IVP

$$y' = \sqrt{y}, \qquad y(0) = 1$$

is $y = (x+2)^2/4$. Solve the above IVP for $x \in [0,\,1]$ using three methods: simple Euler, Modified Euler, and Midpoint. In each case, use two values for the step size: $h = 0.1$ and $h = 0.05$. Make a

[1] An additional condition that is required for this to hold is $(1+ah) > 0$ (you will see that when you will have carried out the derivation). Assume that this condition holds for the case(s) in question.

[2] Thus, the initial and final points of the computation are $x_0 = 0$ and $x_{\text{final}} = 1$.

table that will show the error of your numerical solution at $x = 1$ (i.e., the global error) versus the method used and the step size. By what factor does the error decrease when the step size decreases from 0.1 to 0.05? (**Problem 4 is continued on the next page.**)

Are your results consistent with the expressions for the local truncation errors of these three methods, derived in Lecture 1 and in Problem 3 above? Namely:
(i) Do the dependences of these errors on h agree with the theoretical predictions?
(ii) Does the sign of the error for the simple Euler method agree with that which follows from the derivation in Sec. 1.3 for this specific $y(x)$?
(iii) Do the signs and relative magnitudes of the errors of the Modified Euler and Midpoint methods agree with the result of Sec. 1.6 and your result in Problem 3?

Technical notes:
1. Two alternative examples of a code using the simple Euler method are posted under the link "Codes for examples and selected homework problems". You may mimic your work on either of those examples.
2.

When programming a new numerical scheme for the first time, it is a good idea to stay close to the notations in which this scheme is written in the lecture. In theory, you can use any notations. However, when you are stuck and ask for my help, I will provide useful feedback about your code most efficiently if it is written in my notations. I will not learn your notations just to help you.
Please keep this concept in mind when programming <u>all assignments</u> in this course.

<u>Problem 5</u>
A sky diver jumps from a plane. Assume that during the time before the parachute opens, the air resistance is proportional to the diver's velocity v. (In reality, it is proportional to v^a with $a > 1$, but we sacrifice the physics in exchange for being able to obtain a simple analytical solution to this problem.)

Write down the ODE for the diver's velocity. Supply the numerical values for the coefficients in this ODE if it is known that the maximum diver's velocity is 80 mph. (This velocity could be achieved only asymptotically, assuming that the diver will be falling down forever.)

Solve the above ODE, assuming that the diver's initial velocity is zero. First, find the analytical solution. Then, solve the corresponding IVP by the Modified Euler method for the first 2 seconds of the diver's fall. Use the step size of 0.2 sec. Plot both solutions in the same figure. Also, in a separate figure, plot the numerical error for all $t \in [0, 2]$.

HW # 2
Due: 01/31/14

<u>Problem 1</u>
Derive Eqs. (2.4). Follow the lines of the derivation of coefficient A in Sec. 1.4. (In fact, Eq. (1.21) should be used without any changes.)

<u>Problem 2</u>
(a) Write a function file named `yourname_cRK.m` that can integrate any given IVP $y' = f(x, y)$, $y(x_0) = y_0$ using the classical Runge-Kutta method.

Note: Use as an example the posted sample code, mentioned in Problem 4 of HW# 1, which implements the simple Euler method to integrate the above IVP.

(b) Consider the IVP you derived in Problem 5 of HW# 1. Solve it using the function `yourname_cRK.m` with the same step size as in HW# 1 (i.e., 0.2 sec). Plot the error of your numerical solution.

Compare the final error (at $t = 2$) of the cRK method with the error of the Modified Euler method, obtained in HW# 1. Do you think that the relative magnitudes of these errors are consistent with the orders of the respective methods?
Hint for the last question: What are the orders of magnitude (in terms of h) of the erros of these two methods and what is their ratio?

Problem 3
Consider the following modification of Problem 5 in HW# 1. Assume that at $t = 2$ sec, the parachute opens. This results in the air resistance coefficient changing abruptly in such a way that the maximum possible velocity of the parachutist is now 4 mph (versus 80 mph without the parachute).

Find the numerical value of the new air resistance coefficient and then analytically solve the modified ODE up to $t = 4$ sec (i.e., the first 2 seconds without a parachute, as in HW# 1, and the last 2 seconds with the parachute). [*Hint*: You will need to simply patch two analytical solutions in the aforementioned time intervals.]

Note: Even though now in equation $dv/dt = g - \alpha(t)\, v \equiv f(t, v)$ the function $f(t, v)$ is discontinuous in t (at $t = 2$ seconds), it is still continuous and Lipshitz in v for any one t. Thus, by the existence and uniqueness theorem for ODEs, we are still guaranteed to have a unique solution $v(t)$.

Solve the above IVP up to $t = 4$ sec using the function file `yourname_cRK.m` that you wrote in Problem 2 (see the *Technical notes* below on how to handle a discontinuous $f(t, v)$). Use the step sizes of 0.2 sec. Plot the exact and the two numerical solutions in the same figure. Compute its error for all $t \in [0, 4\,\text{sec}]$ and plot it in another figure.

Technical notes:

To define a discontinous function in a separate M-file, a new M-file and name it `yourname_fun4_hw2_p3`. Type the following into the file:

```
function f=yourname_fun4_hw2_p3(t,y)
if t >= 0 &  t < 2
   f= "r.h.s. of ODE before parachute opens";
else
   f= "r.h.s. of ODE after parachute opens";
end
```

Of course, you supply the actual expressions on the r.h.s..
Please use the same name for your function as the name of the M-file you store it in.
Before the line `function f=yourname_fun4_hw2_p3(t,y)`, you may want to declare as global varaibles those parameters which you want to communicate from the main code to the function; see the posted sample codes for HW# 1. For example, you do *not* want to enter the numeric value of your terminal velocity (e.g., 80 mph) into the function code. Instead, it is much more literate to enter that value in the main code and communicate it to the function through the `global` declaration.

Problem 4 (worth **2 points**)
Solve the IVP in Problem 3 above by the Runge–Kutta–Fehlberg method, assuming the accuracy $\varepsilon_{\text{loc}} = 10^{-3}$ mph. Use $\kappa = 0.8$ (κ is defined in Sec. 2.2). Plot both the exact and numerical solutions in the same figure. Compute the global error, as well as the error controlled by the RKF, for all $t \in [0, 4\,\text{sec}]$. Plot each of these errors in its own figure. Also, save the global error in a file (see Problem 6 below, where you will be asked to plot that error).

Technical notes:
1. In order to compute the error for all t, you will have to compute the exact solution on the grid which you will create while computing the numerical solution. So, at each step, record the computed value of t at this step.
2. In the codes you have written so far, you knew the step size h and so knew exactly how many steps you had to make to obtain your numerical solution. This allowed you to use a `for`-loop in your ODE-solver (see, e.g., `hw1_EX_Euler_callsfun.m`). In this problem you do not know the number of steps and hence cannot use a `for`-loop. Use either a `while`- or `if`-loop instead. The calculations should exit that loop when the total computed time exceeds 4 seconds.

Problem 5 (worth **0.5 point**)
Repeat Problem 4 using MATLAB's built-in ODE-solver `ode45`. To see its syntax, type '`help ode45`' at MATLAB's prompt. As the function for the `ode45`, supply the same function file as for Problem 3 above.

Plot the exact and numerical solutions in the same figure. Compute the error for all $t \in [0, 4\,\text{sec}]$ and save it in a file. It will be plotted in the next problem.

Problem 6 (worth **0.5 point**)
In a single figure, plot the global errors of the numerical solutions obtained in Problems 3–5. State how these methods perform for this ODE relative to one another.
Hint: You may first save the error (or the solution) for each case in a separate file using the command `save` and then load these files (one by one), using the command `load`, to plot the errors. Type `help save` or `help load` to find out the syntax of these commands.

Bonus[3] (worth **0.5 point**)
Compare the speeds of Runge–Kutta–Fehlberg and `ode45` methods. Specifically, do the following. First, put your codes for each of these methods into a loop, so that the same calculation is repeated 200 times (this is needed for statistical averaging, since the computational speed will fluctuate from run to run). Second, use any of the following 3 commands: `etime`, `cputime`, or `tic` and `toc`. Examples of their usage can be found by typing '`help tic`', etc. Which of the two methods appears to be faster? *Note*: To receive full credit for this problem, you must submit its code(s) to me.

HW # 3

Due: 02/07/14

Problem 1
Using Taylor expansions of y'_{i-1} and y'_{i-2} about $x = x_i$, verify that

$$y'''_i = \frac{y'_i - 2y'_{i-1} + y'_{i-2}}{h^2} + O(h)\,.$$

Problem 2
Obtain the counterpart of the linear system (3.15) *without* assuming that $x_i = 0$. Verify that it actually coincides with (3.15). *Note*: You are *not* asked to solve the system of equation for $x_i \neq 0$. You just need to show that it reduces to that for $x_i = 0$.

Problem 3
Find the coefficients b_{-1}, b_0, b_1 in

$$Y_{i+1} = Y_i + h(b_{-1} f_{i+1} + b_0 f_i + b_1 f_{i-1})$$

that produce a 3rd-order method (called the 3rd-order Adams-Moulton method). Use a technique from either Secs. 3.1 or 3.2 (your choice).

Problem 4 (worth **1.5 points**)
Use the P–C method given by Eqs. (3.33), (3.39), and (3.40) to solve

$$y' = \sin y, \qquad y(0) = 1\,, \qquad x \in [0,\, \pi]\,.$$

Select the step size so that the local truncation error, given by (3.39), be at most $\varepsilon_{\rm loc} = 10^{-4}$. Provide an explanation for your choice of the step size (see below).

Technical notes:

1. You need Y_1, in addition to Y_0, to start the method. Select a suitable method from Lecture 1 to obtain Y_1. This issue is discussed in a Remark at the end of Sec. 3.6. You must explain what considerations went into your decision of selecting a particular starting method.

2. Regarding the requirement about the step size:
You do *not* want to adjust the step size as you are running the calculation. (Why not? Again, the answer is in the notes.) Then you need to use Eq. (3.37) to estimate the step size required for the error of the P–C method not to exceed a given value. To this end, you will need an estimate for y''', which you can obtain using the very special form of the function $f(x, y)$ in your ODE, and formula (1.28).

Plot your solution. Also, plot the estimate for the error (starting with $i = 2$), deducing it from Eq. (3.39), and verify that the above requirement on the error to be less than $\varepsilon_{\rm loc}$, is indeed satisfied.

[3] The following applies to any Bonus problem assigned in this course. (i) A Bonus problem is considered to be worth one regular problem unless stated otherwise. (ii) A Bonus problem must be done mostly correctly to receive credit; that is, no extra credit for it will be given if its solution has major errors or gaps.

HW # 4

Due: 02/

Note to all problems in this homework set:
The stability is implied with respect to the model problem (4.15) of the notes.

Problem 1

For the Modified Euler method, obtain Eqs. (4.21) and (4.22). Then use Mathematica (using command `ContourPlot`) or Matlab (using command `contour`) to generate the graph of the stability region boundary given by Eq. (4.22).

Problem 2 (worth **0.5 point**)

Use inequality (4.23) to obtain the bound (4.24) for the stability of the cRK method when $\lambda < 0$ (i.e., is real). *Hint:* In this case, you do not need Mathematica. A simple plot in Matlab will suffice.

Problem 3 (worth **0.5 point**)

Show analytically that the region of stability for the Midpoint method coincides with that for the Modified Euler method.

Problem 4 (worth **0.5 point**)

Use Eqs. (4.27) and inequalities (4.30) to plot the boundary of the stability region of the 2nd-order Adams–Bashforth method (3.5).

Technical notes:

1. To compute the absolute value of a complex number, use command `abs` in Matlab or `Abs` in Mathematica.
2. If you need to show two graphs together in Mathematica, the procedure is the following. Name each plot as follows: `p1=ContourPlot[ ...here goes your command for graph 1 ... ]`, and similarly for `p2`. Then, on a new line, type `Show[p1,p2]`.

Problem 5

Obtain the analog of Eqs. (4.26) and (4.27) for the P–C method (3.33) of Lecture 3.
Plot its stability region following the suggestions of Problem 4. Your graph should have the shape of a heart pointing to the left.

Is this stability region larger or smaller than that of the 2nd-order Adams–Bashforth method? In general, try to make an educated guess about how the stability region of a P–C method is related to the stability regions of its predictor and corrector equations. (A similar conclusion will also follow from the Bonus problem for this HW assignment.)

Problem 6

Find *analytically* (i.e., without Mathematica or Matlab) the stability region of the Modified Implicit Euler method (3.43) and make a sketch of this region. Explain why your result implies that this method is A-stable.

Problem 7

Solve the I.V.P.

$$y' = -20y, \qquad y(0) = 1$$

with $h = 0.125$ up to $x = 1.5$ using the: (a) simple Euler, (b) cRK, and (c) implicit Euler methods. Plot your result in (a). In a separate figure, plot your results in (b) and (c) along with the exact solution. Which method, the 4th-order (b) or the 1st-order (c), gives the more accurate solution in this case?

Without doing additional calculations, what do you expect to change in those results if you use $h = 0.15$ instead of $h = 0.125$? Write a brief but coherent paragraph explaining your answer.

Bonus (worth **1.5 points**)

As it is shown in the notes, the Leap-frog method, introduced in Lecture 3, is unstable for $\lambda < 0$. Now, construct a P–C method where the Leap-frog method is used as the predictor and the trapezoidal rule is used as the corrector (as in method (3.33)). Repeat Problem 5 for this new P–C method. (Your graph should look qualitatively similar to the stability region of the 3rd-order Adams–Bashforth method found in the notes.) How does this region compare with the stability regions of the predictor equation (Leap-frog) and of the corrector equation (implicit modified Euler; see Problem 6) alone? Does this agree with what you observed in Problem 5?