Final Report
Computational Physics - Phys 562

Benjamin Deutsch
Department of Physics
California State University Long Beach

May 16, 2017

Abstract

No abstract

1 Problem 1

1.1 Method(Description of code)

For the first question, we notice the separable differential equations, and make use of the Runge Kutta subroutine as given in class. Carefully, we edit the derivative statements `f(1), f(2), f(3)` to appear the same as the testing sheet as;

$$\frac{dx}{dt} = -(y+z), \quad \frac{dy}{dt} = x + ay, \quad \frac{dz}{dt} = b + z(x-c) \tag{1}$$

It should be noted that we have preserved the calculation for K, as it is fundamental for the RK4 step method. It will be used within the subroutine to adjust the increment of interpolations based on the slope at given interval points. After this has been done, we must check that the intent in and out statements do match with your written functions, for call by the main routine. Within the main routine, we have changed the module to notify the program that we have three equations that will be used externally, as well as the parameters `a, b, c` to be passed into the subroutine as well. The real benefit of the RK subroutine is seen in utilizing a simplified program, here

along with parameters `a, b, c` being defined, we have also included time step criteria and initial conditions, all of which will be implemented in the external subroutine. Finally the creation of a do loop cycling over an defined interval of time, calls to the subroutine with the above parameters and writes the output to specific files. These are then graphed for analysis.

1.2 Results

In parts A and B of question 1, we were asked to alter certain parameter and investigate the resulting behavior. In order to do this we have altered these variables manually within the code (as guided by the comments). The effects of part A, varying `c` from 5 to 8 can be seen below. A few things can be said here, firstly that in lowering the c value to 5 we can see a increasingly repetitive trend in the graph, as bolder lines in specific regions begin to form. Increasing the duration of this value only seems to localize the values, controlling the level of disorder in the system. This is further reinforced by the repetitive nature of the c = 8 graph (right), the spacing of each pathway is distributed almost evenly. Lastly comparing the two time dependent graphs for each set of parameters indicates that as a and b grow it reaches some limit where in c grows exceptionally quickly, spiking and suppressing grow in a and b parameters. This oscillational property has indicates a precise area of chaotic behavior.
In the section for B, we are asked to change the initial condition `x, y, z`, doing this by way of three defined sets, beginning at zeros, x at zero while y and z are negative and lastly x and y being positive and z negative. We can easily see the behavior of the zeros starting system in the below time dependent images, concluding that slowly the graphs returned to "normal" behavior after some time. This is reinforced by reviewing the other two initial conditions (0, -,-) and (+, +, -). Where the graphs seem to differ is within there complete plots (X, Y, Z), It seems here that as the initial conditions are allowed to be separated farther apart numerically, we observe pathway intervals at more diverse separations.

Table 1: Part 1; Changing Parameter C

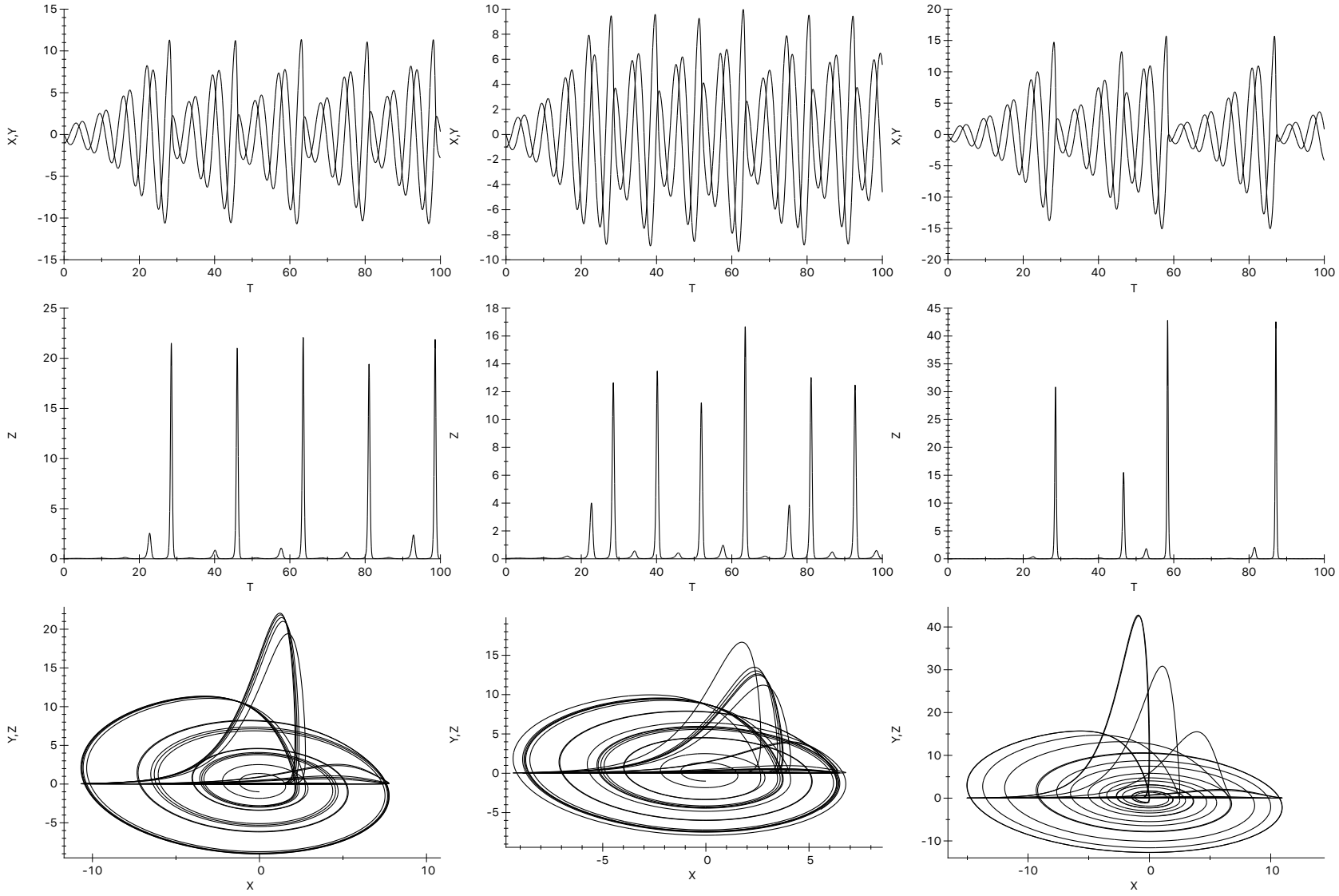


Table 2: Part 2; Initial conditions (000), (0–), (++-)

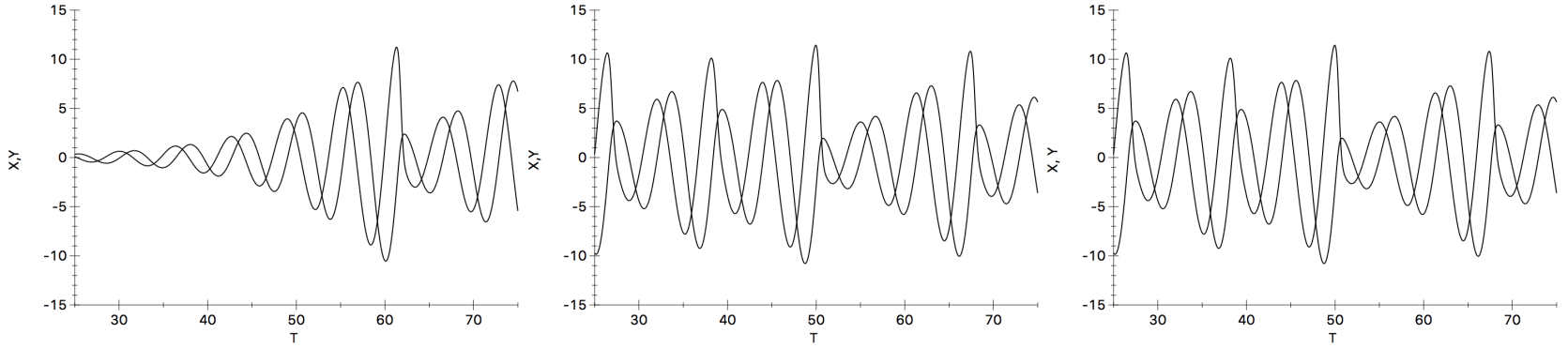
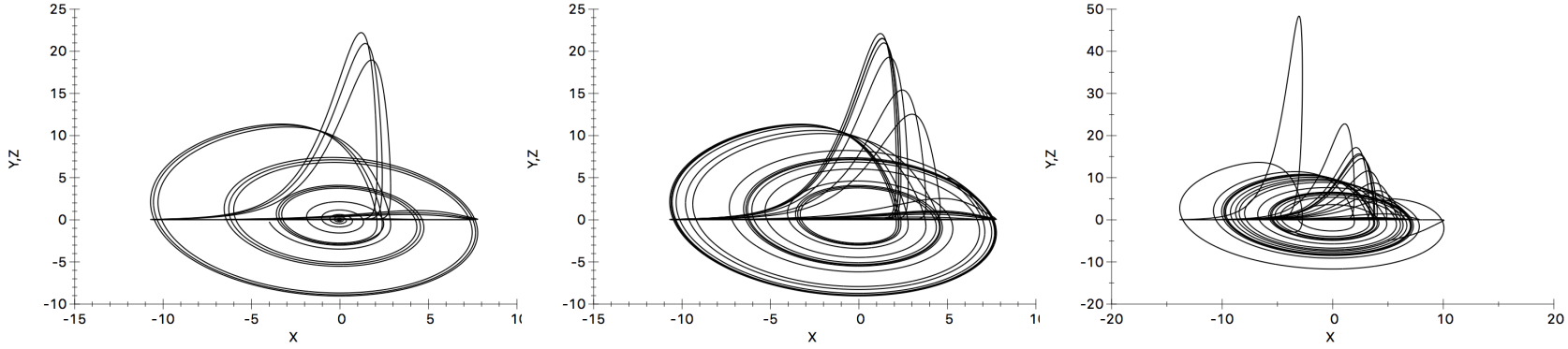


Table 3: Part 2; (000), (0–), (++-)



2 Problem 2

2.1 Description of code

To avoid a lengthy discussion on the set up of problem 2, we have utilized past code `Q.f95` (Midterm 4) as a starting point for this code. We can do this for a number of reasons, primarily its purpose was to calculate was to produce a set of eigenvalues and component eigenvectors. It did this, by compiling a working hamiltonian matrix for the 1-D harmonic oscillator. Keys components of the pervious working code utilized,

$$\psi_n(x) = \frac{1}{\sqrt{2^n\, n!}} \cdot \left(\frac{m\omega}{\pi\hbar}\right)^{1/4} \cdot e^{-\frac{m\omega x^2}{2\hbar}} \cdot H_n\left(\sqrt{\frac{m\omega}{\hbar}}x\right) \tag{2}$$

and $\hat{E_n}$ as

$$\hat{E_n} = \left(n + \frac{1}{2}\right)\hbar\omega \tag{3}$$

to produce (with our oscillator potential) for brevity,

$$T_{m,n} = E_{m,n} + U_{m,n} \tag{4}$$

and the hamiltonian as;

$$H_{m,n} = T_{m,n} + V_{m,n} \tag{5}$$

Here we are asked to assume a perturbation to the system resulting in changed elements of $H_{0,5}$ and $H_{5,0}$ by $\frac{\hbar w_H}{2}$ with omega (w2=2). Simplified we have;

$$H_p = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{6}$$

With this we can write,

$$H_{new} = H_{old} + H_p \tag{7}$$

Part A, Finding the eigenvalues and eigenvector can be easily done here by implementing the `dsyev` subroutine, a component to the LAPACK framework, used to do just this. To check for completeness we have done a little matrix algebra (Fortran's intrinsic `Matmul`) as follows;

$$\mathbb{1} = H_{new} * Transpose(H_{new}) \tag{8}$$

We have omitted the conjugate in this operation as this is not a complex matrix. Achieving the identity we have completed the first task.
Part B, was slightly more involved. We are asked to calculate the matrix representation of e^{-iHt}, with H being our newly perturbed matrix. In order to do this, after research it is given that;

$$B = M^{\dagger} * A * M \tag{9}$$

Where B is your matrix, M are the eigenvalues and A the eigenvectors. So it follows that;

$$f(B) = M^{\dagger} * f(A) * M \tag{10}$$

In order to add this to the main routine, we noticed that since the matrix A (the eigenvalues) needs to contain only diagonal entries we can simply include the required function e to each individually along with the constants. Finally taking each output and multiplying it against the identity matrix to align the eigenvectors along the diagonals. Note; the matrix while maintaining the dimension of the identity will now be recorded (ZZ matrix) as a complex matrix, due to the multiplication of the i, seen above.
Now with a diagonalized matrix of the eigenvalues and a previously computed eigenvector matrix, we can straight forwarding use the equation above (10), and a nested `Matmul` function to produce the final matrix representation.
As asked we can alter the t (`tt`) between 1 and 5 manually, as indicated by the comments.

2.2 Results

	Eigenvalues
1	0.57386019704424396
2	2.1226186643438076
3	3.5561285389571826
4	5.0242884193833461
5	6.9833016503265242
6	8.7398024703352171

Table 4: Eigenvalues for $\omega = 2$

For results see terminal output.
These have been compared against Maple software and Wolfram Alpha's Mathematica.

3 Problem 3

3.1 Description of code

For problem 3 we are asked to produce a code for calculating the residua for $f(x) = \pi cot(\pi x)$ at $x = 0$, $x = 0$, and $x = \pi$. Again we can amend a pervious code in order to fulfill this contour integral, however before doing this we must do a little math. Given the function;

$$f(x) = \frac{\pi}{tan(x)} \tag{11}$$

We create the complex contour intergral;

$$f(z) = \frac{1}{2\pi} \oint \frac{\pi}{tan(\pi z)} dz \rightarrow \frac{1}{2i} \oint \frac{1}{tan(\pi z)} \tag{12}$$

Differentiating z

$$\frac{1}{2} \int_0^{2\pi} \frac{1}{tan(\pi * re^{it})} re^{it} dt \quad = \quad \frac{1}{2} \int_0^{2\pi} \frac{re^{it}}{tan(\pi * re^{it})} dt \tag{13}$$

With this result we can readily use the `d01bcf` subroutine for calculation of the weights and abscissae needed in approximating the Gauss-Legendre

function type. Through the summation;

$$\sum_{i=1}^{n} w_i f(x_i) \quad w_i = weights \quad x_i = abscissae \tag{14}$$

Here the do loop runs over this sum for our specific problem, held in a external function in order to integrate the resulting residue. The parameters for which can be changed manually in the code itself, as indicated by the comments.

3.2 Results

The results are as follows, for $f(x) = \frac{\pi}{tan(x)}$

x=	f(x)=
0	1
$\frac{\pi}{2}$	0
π	0

References

[1] Z. Papp and A. Bill, *Computational Physics Lecture Notes*, California State University Long Beach.