Homework 4: Computational Physics

Pablo Lopez Duque

October 7^{th}, 2020

1 Problem 1

The Hopf model is given by:

$$\frac{dx}{dt} = ax + y - x(x^2 + y^2) \qquad \frac{dy}{dt} = -x + ay - y(x^2 + y^2)$$

(a) **Rewrite these equations in polar coordinates. Analytically show that they fall into the origin if $a < 0$, and limit to a circle of radius $\sqrt{a}$ for $a > 0$.**

Let $x = r\cos(\theta)$, $x = r\sin(\theta)$:

$$\Rightarrow \frac{\partial x}{\partial r} = \cos\theta \qquad \frac{\partial x}{\partial \theta} = -r\sin\theta$$
$$\frac{\partial y}{\partial r} = \sin\theta \qquad \frac{\partial y}{\partial \theta} = r\cos\theta$$

$$\Rightarrow \frac{dx}{dr} = \frac{\partial x}{\partial r}\frac{dr}{dt} + \frac{\partial x}{\partial \theta}\frac{d\theta}{dt} = \cos\theta\frac{dr}{dt} - r\sin\theta\frac{d\theta}{dt} = r\cos\theta(a - r^2) + r\sin\theta \tag{1}$$

$$\Rightarrow \frac{dy}{dr} = \frac{\partial y}{\partial r}\frac{dr}{dt} + \frac{\partial y}{\partial \theta}\frac{d\theta}{dt} = \sin\theta\frac{dr}{dt} + r\cos\theta\frac{d\theta}{dt} = r\sin\theta(a - r^2) - r\cos\theta \tag{2}$$

Now, we multiply equations (1) $\times \cos\theta$ and (2) $\times \sin\theta$ and sum:

$$\cos^2\theta\frac{dr}{dt} - r\sin\theta\cos\theta\frac{d\theta}{dt} = r\cos^2\theta(a - r^2) + r\sin\theta\cos\theta$$

$$\sin^2\theta\frac{dr}{dt} + r\sin\theta\cos\theta\frac{d\theta}{dt} = r\sin^2\theta(a - r^2) - r\sin\theta\cos\theta$$

$$\underbrace{(\sin^2\theta + \cos^2\theta)}_{1}\frac{dr}{dt} = \underbrace{(\sin^2\theta + \cos^2\theta)}_{1}r(a - r^2)$$

$$\Rightarrow \boxed{\frac{dr}{dt} = r(a - r^2)} \tag{3}$$

And, to obtain the equation for $\theta(t)$, we multiply equations (1) $\times -\sin\theta$ and (2) $\times \cos\theta$ and sum:

$$-\sin\theta\cos\theta\frac{dr}{dt} + r\sin^2\theta\frac{d\theta}{dt} = -r\sin\theta\cos\theta(a - r^2) - r\sin^2\theta$$

$$\sin\theta\cos\theta\frac{dr}{dt} + r\cos^2\theta\frac{d\theta}{dt} = r\sin\theta\cos\theta(a - r^2) - r\cos^2\theta$$

$$r\underbrace{(\sin^2\theta + \cos^2\theta)}_{1}\frac{d\theta}{dt} = -r\underbrace{(\sin^2\theta + \cos^2\theta)}_{1}$$

$$\Rightarrow \boxed{\frac{d\theta}{dt} = -1} \qquad if \quad r \neq 0 \tag{4}$$

Now that we have the equations in polar coordinates, we can see that they are a system of uncoupled ODEs.

Let's find the fixed points and analyze their stability:

Case $a > 0$

Let $\rho = \frac{r}{\sqrt{a}} \Rightarrow d\rho = \frac{dr}{\sqrt{a}}$

$$\Rightarrow \boxed{\frac{d\rho}{dt} = a\rho(1-\rho^2)}$$

We can discretize the system as follows:

$$\Rightarrow \frac{\rho_{n+1} - \rho_n}{\Delta t} = a\rho_n(1-\rho_n^2)$$
$$\Leftrightarrow \rho_{n+1} = \rho_n + (a\Delta t)\rho_n(1-\rho_n^2)$$

Thus, the fixed points are:

$$\cancel{\rho_\infty} = \cancel{\rho_\infty} + (a\Delta t)\rho_\infty(1-\rho_\infty^2)$$

$$\Rightarrow \rho_\infty = 0 \quad or \quad \rho_\infty = 1$$

$$\Leftrightarrow r^* = 0 \quad or \quad r^* = \sqrt{a}$$

Now, recall that to have a stable point, we must have $|f'(r^*) < 1|$. Hence:

$$f'(\rho_n) = 1 + (a\Delta t)(1 - \rho_n^2 + \rho_n(-\rho_n^2)) = 1 + (a\Delta t)(1 - 3\rho_n^2)$$

So, for $\rho^* = 0$:

$$f'(0) = 1 + (a\Delta t) > 1$$

as both a and Δt are positive. Hence, $\rho^* = 0$ is an unstable point.

For $\rho^* = 1$:

$$f'(1) = 1 + (a\Delta t)(1-3) = 1 - 2(a\Delta t) < 1$$

Hence, $\rho^* = 1$ is a stable point.

$\therefore$ $r = 0$ is an unstable point, whereas $r = \sqrt{a}$ is a stable point.

Case $a < 0$

There are only changes in signs, as follows:

$$\Rightarrow \frac{\rho_{n+1} - \rho_n}{\Delta t} = -|a|\rho_n(1+\rho_n^2)$$
$$\Leftrightarrow \rho_{n+1} = \rho_n - (|a|\Delta t)\rho_n(1+\rho_n^2)$$

Thus, the fixed points are:

$$\cancel{\rho_\infty} = \cancel{\rho_\infty} + (|a|\Delta t)\rho_\infty(1+\rho_\infty^2)$$

$$\Rightarrow \rho_\infty = 0 \quad or \quad \rho_\infty = \pm i$$

But since we are interested only in real values, we can ignore the second solution.

$$\Leftrightarrow r^* = 0$$

Now, the stability analysis yields:

$$f'(\rho_n) = 1 - (|a|\Delta t)(1 + \rho_n^2 + \rho_n(\rho_n^2)) = 1 - (|a|\Delta t)(1 + 3\rho_n^2)$$

So, for $\rho^* = 0$:

$$f'(0) = 1 - (a\Delta t) < 1$$

which means $r = 0$ is a stable point in this case.

(b) **Compute the dynamics for the Hopf model using an adaptive runge-kutta technique (you are free to use $rk45$ in matlab or python, or use Garcia's code).**

Using the adaptive Runge-Kutta technique, we can easily solve the resulting equation:

$$\begin{cases} \dot{r} = r\left(a - r^2\right) \\ \dot{\theta} = -1 \end{cases}$$

which means we have a system of uncoupled differential equations.

In order to solve it, we must then define the parameters and initial conditions for our system:

```
A = 4;
ms=3;
tspan = [0 3.5];
yinit=[0.0001,0.5*pi;0.1,0.5*pi;0.5,0.5*pi;1.5,0.5*pi;4.0,0.5*pi];
```

where we define a 5×2 matrix to automatize the solutions for several initial conditions. In the same manner we automatize the marker selection:

```
chars=["-o","-x","-s","-.","-"];
```

Case a>0:

Then, we can use ode45 to solve the system, plot the solutions in the same figure and save it :

```
for ii=1:size(yinit,1)
    [t,y] = ode45(@(t,y) odefcn(t,y,A), tspan, yinit(ii,:));
    plot(t,y(:,1),chars(ii),'MarkerSize',ms)
    hold all
end
title('Hopf model theta solution for a=4');

legend('r0=0.0001','r0=0.1','r0=0.5','r0=1.5','r0=4');
box on

ax=gca;
ax.FontSize=12;

xlabel('t');
ylabel('r(t)');

%saveas(gcf,'HW4_1b_loop_a4','epsc');

hold off
```

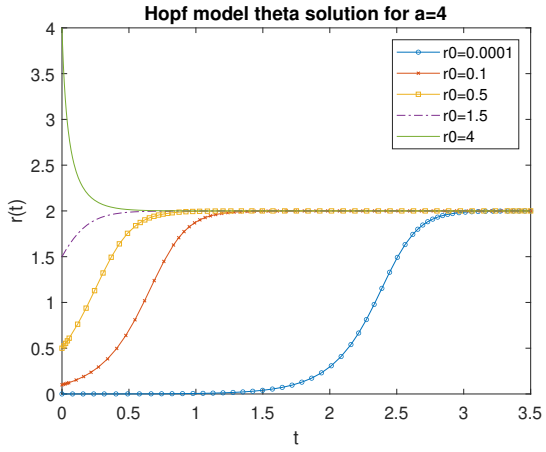

Figure 1: Hopf model for a=4. The stable point is $r = \sqrt{a}$ whereas the unstable point is $r = 0$.

As we can see in this figure below, the solutions approach $\sqrt{4}$ for any initial set of values.

Now, lets try with a couple different values for a:

```
%For a=2
A = 2;
ms=3;
tspan = [0 6];
yinit=[0.0001,0.5*pi;0.1,0.5*pi;0.5,0.5*pi;1.5,0.5*pi;4.0,0.5*pi];
for ii=1:size(yinit,1)
    [t,y] = ode45(@(t,y) odefcn(t,y,A), tspan, yinit(ii,:));
    plot(t,y(:,1),chars(ii),'MarkerSize',ms)
```

```
    hold all
end
title('Hopf model theta solution for a=2');

legend('r0=0.0001','r0=0.1','r0=0.5','r0=1.5','r0=4');
box on

ax=gca;
ax.FontSize=12;

xlabel('t');
ylabel('r(t)');

%saveas(gcf,'HW4_1b_loop_a2','epsc');

hold off
```

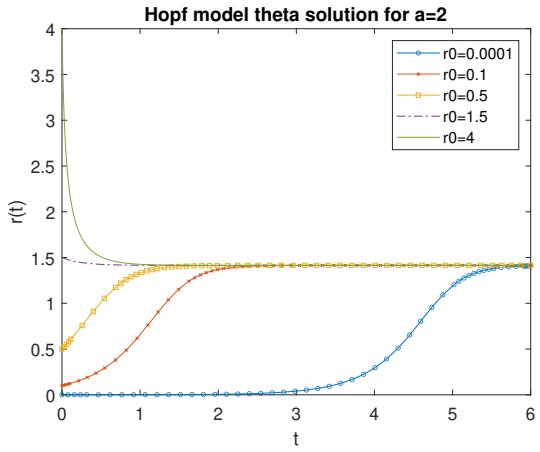

Figure 2: Hopf model for a=2. The stable point is $r = \sqrt{a}$ whereas the unstable point is $r = 0$.

Again, the function approaches $r = \sqrt{2}$.

Finally, lets try $a = 0.25$:

```
%For a=0.25
A = 0.25;
ms=3;
tspan = [0 40];
yinit=[0.0001,0.5*pi;0.1,0.5*pi;0.5,0.5*pi;1.5,0.5*pi;4.0,0.5*pi];
for ii=1:size(yinit,1)
    [t,y] = ode45(@(t,y) odefcn(t,y,A), tspan, yinit(ii,:));
    plot(t,y(:,1),chars(ii),'MarkerSize',ms)
    hold all
end
title('Hopf model theta solution for a=1/4');

legend('r0=0.0001','r0=0.1','r0=0.5','r0=1.5','r0=4');
box on

ax=gca;
ax.FontSize=12;

```

```
19  xlabel('t');
20  ylabel('r(t)');
21
22  %saveas(gcf,'HW4_1b_loop_a025','epsc');
23
24  hold off
```

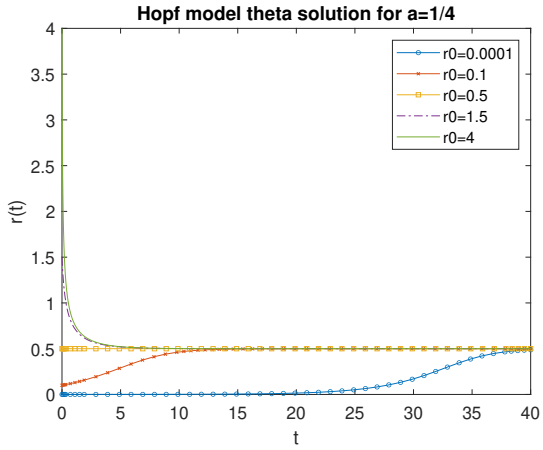

Figure 3: Hopf model for $a = \frac{1}{4}$. The stable point is $r = \sqrt{a}$ whereas the unstable point is $r = 0$.

Once again, the function approaches $r = \sqrt{0.25}$.

We can clearly see that whenever the initial values of r are close to zero, the time that it takes to converge to $\sqrt{a}$ is longer. Still, it always ends up converging (unless we start exactly with $r = 0$).

Case a<0:

```
%For a=-4
A = -4.0;
ms=3;
tspan = [0 1.5];
yinit=[0.0001,0.5*pi;0.1,0.5*pi;0.5,0.5*pi;1.5,0.5*pi;4.0,0.5*pi];
for ii=1:size(yinit,1)
    [t,y] = ode45(@(t,y) odefcn(t,y,A), tspan, yinit(ii,:));
    plot(t,y(:,1),chars(ii),'MarkerSize',ms)
    hold all
end
title('Hopf model theta solution for a=-4');

legend('r0=0.0001','r0=0.1','r0=0.5','r0=1.5','r0=4');
box on

ax=gca;
ax.FontSize=12;

xlabel('t');
ylabel('r(t)');

%saveas(gcf,'HW4_1b_loop_an4','epsc');

hold off
```

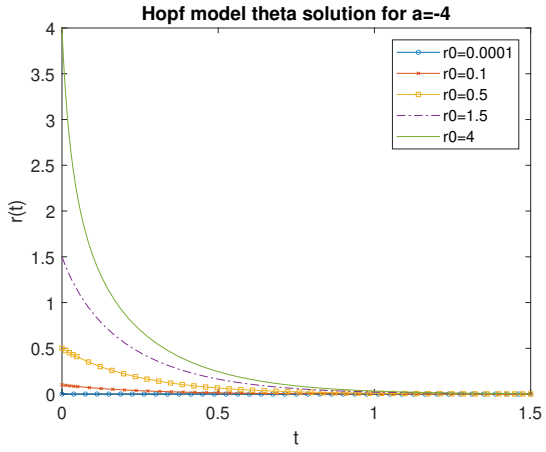

Figure 4: Hopf model for $a = -4$. The only fixed and stable point is $r = 0$.

In the case $a < 0$, we see a completely different behavior, as we showed in part a. In this case, the system always converges to $r = 0$. The only difference the starting parameters have is on how fast the approach to zero is.

In summary, whenever $a > 0$, the stable point is $r = \sqrt{a}$ and the time it takes to converge to it is dependent on the initial parameters. If we choose to start too close to zero, the time it takes to converge to $\sqrt{a}$ becomes larger. If we were to start right at zero, then it would take infinite time in order to converge at the stable point $\sqrt{a}$ as $r = 0$ is an unstable point. In contrast, when $a < 0$, the only stable point is $r = 0$ and the time it takes for convergence depends only on how far from zero we start.

Theta equation

The simpler to analyze is the θ equation as it clearly represents a constant decrease over time.

```
A = 2.0;
ms=3;
tspan = [0 40];
yinit=[0.0001,0.5*pi;0.1,0.5*pi;0.5,0.5*pi;1.5,0.5*pi;4.0,0.5*pi];
for ii=1:size(yinit,1)
    plot(t,y(:,2),chars(ii),'MarkerSize',ms)
    hold all
end
title('Hopf model theta solution for a=2');

legend('r0=0.0001','r0=0.1','r0=0.5','r0=1.5','r0=4');
box on

ax=gca;
ax.FontSize=12;

xlabel('t');
ylabel('theta(t)');

%saveas(gcf,'HW4_1b_loop_theta_a2','epsc');
hold off
```

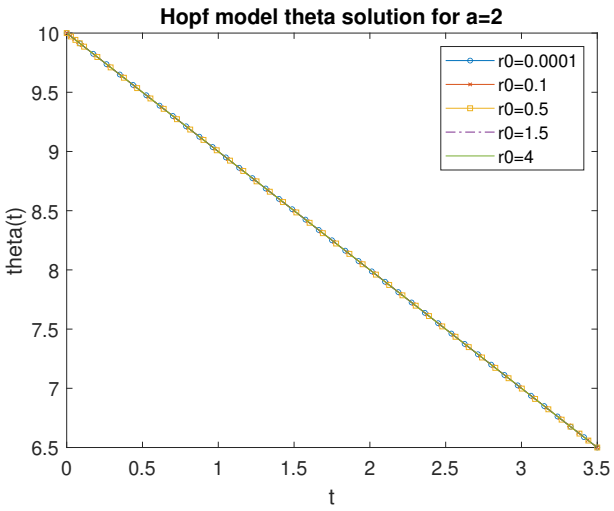

Figure 5: Hopf model theta result for a=2

Clearly, no matter what parameters we choose, we will always end up with the same behavior. The initial point depends on the initial angle that we choose, but the slope will always be the same.

In order to implement any of the code parts above, we must define the following function:

```
function dydt = odefcn(t,y,A)
    dydt = zeros(2,1);
    dydt(1) = y(1)*(A-y(1)^2);
    dydt(2) = -1;
end
```

2 Problem 2

Suppose a $n \times n$ matrix M has n orthogonal eigenvectors $\vec{v_i}$, such that $M\vec{v_i} = \lambda_i \vec{v_i}$ with all λ_i distinct.

(a) **Show analytically that $M = VLV^{-1}$, where L is a diagonal matrix of eigenvalues and V is the matrix formed of columns of eigenvectors (that is, $V_{ij} = (v_j)_i$).**

(b) **Use (a) to analytically show that M^k has eigenvalues λ_i^k, with the same eigenvectors. Use that to show that $M^k\vec{b} = c_1\lambda_1^n\vec{v_1} + c_2\lambda_2^k\vec{v_2} + ...$ for some c_i you should determine. Argue that this means $lim_{k\to\infty} M^k b \propto \vec{v}_{max}$(for a constant of proportionality you should determine), with $\vec{v}_{max}$ the eigenvector associated with the largest eigenvalue of M.**

We can rewrite M^k as follows:

$$M^k = (VLV^{-1})^k = \underbrace{VL\overbrace{V^{-1}V}^{I}L\overbrace{V^{-1}}^{I}..\overbrace{.V}^{I}LV^{-1}}_{k\ \ times} = VL^kV^{-1}$$

$$\Rightarrow M^k = VL^kV^{-1}$$

Since we know that L is a diagonal matrix containing the eigenvalues of M, it follows that L^k is a diagonal matrix containing the eigenvalues of M^k. Finally, since for a diagonal matrix the kth power of a matrix is equivalent to taking the kth power of each entry, the eigenvalues of M^k must be precisely λ_i^k where λ_i are the eigenvalues of M. From the form of the boxed equation, we can also tell that the eigenvectors did not change, the matrix V still appears in the same form as $M = VLV^{-1}$.

Now, we can write any vector $\vec{a}$ in terms of the orthonormal set of eigenvectors of M. Recall that since all are independent, they span $\mathbb{R}^k$. Hence:

$$\vec{a} = c_1\vec{v}_1 + c_2\vec{v}_2 + ... + c_n\vec{v}_n$$

$$\Rightarrow X\vec{a} = c_1X\vec{v}_1 + c_2X\vec{v}_2 + ... + c_nX\vec{v}_n = c_1\lambda_1\vec{v}_1 + c_2\lambda_2\vec{v}_2 + ... + c_n\lambda_n\vec{v}_n$$

Now, we can take the dot product with the ith eigenvector, and use the orthonormality condition:

$$\Rightarrow X\vec{a}\cdot\vec{v}_i = c_1\lambda_1\vec{v}_1\cdot\vec{v}_i + c_2\lambda_2\vec{v}_2\cdot\vec{v}_i + ... + c_i\lambda_i\vec{v}_i\cdot\vec{v}_i + ... + c_n\lambda_n\vec{v}_n\cdot\vec{v}_i = \lambda_i\vec{v}_i\cdot\vec{v}_i$$

$$\Rightarrow X\vec{a}\cdot\vec{v}_i = c_i\lambda_i|\vec{v}_i|^2$$

$$\Rightarrow c_i = \frac{X\vec{a}\cdot\vec{v}_i}{\lambda_i|\vec{v}_i|^2} = \frac{\cancel{\lambda_i}\vec{a}\cdot\vec{v}_i}{\cancel{\lambda_i}|\vec{v}_i|^2}$$

$$\Rightarrow c_i = \frac{\vec{a}\cdot\vec{v}_i}{|\vec{v}_i|^2}$$

Thus, we can write $\vec{a}$ as:

$$\vec{a} = \frac{\vec{a}\cdot\vec{v}_1}{|\vec{v}_i|^2}\vec{v}_1 + \frac{\vec{a}\cdot\vec{v}_2}{|\vec{v}_i|^2}\vec{v}_2 + ... + \frac{\vec{a}\cdot\vec{v}_n}{|\vec{v}_i|^2}\vec{v}_n$$

And $X\vec{a}$ as:

$$X\vec{a} = \frac{X\vec{a}\cdot\vec{v}_1}{|\vec{v}_i|^2}\vec{v}_1 + \frac{X\vec{a}\cdot\vec{v}_2}{|\vec{v}_i|^2}\vec{v}_2 + ... + \frac{X\vec{a}\cdot\vec{v}_n}{|\vec{v}_i|^2}\vec{v}_n = \frac{\vec{v}_1\cdot\vec{a}}{|\vec{v}_i|^2}X\vec{v}_1 + \frac{\vec{v}_2\cdot\vec{a}}{|\vec{v}_i|^2}X\vec{v}_2 + ... + \frac{\vec{v}_n\cdot\vec{a}}{|\vec{v}_i|^2}X\vec{v}_n$$

where in the last step, we used the fact that the dot product is just a scalar Hence:

$$X\vec{a} = \frac{\vec{v}_1\cdot\vec{a}}{|\vec{v}_i|^2}\lambda_1\vec{v}_1 + \frac{\vec{v}_2\cdot\vec{a}}{|\vec{v}_i|^2}\lambda_2\vec{v}_2 + ... + \frac{\vec{v}_n\cdot\vec{a}}{|\vec{v}_i|^2}\lambda_n\vec{v}_n$$

Finally, we can apply this to our specific case by choosing $X = M^k$ and $\vec{a} = \vec{b}$:

$$M^k\vec{b} = \frac{\lambda_1^k\vec{v}_1\cdot\vec{b}}{|\vec{v}_i|^2}\vec{v}_1 + \frac{\lambda_2^k\vec{v}_2\cdot\vec{b}}{|\vec{v}_i|^2}\vec{v}_2 + ... + \frac{\lambda_n^k\vec{v}_n\cdot\vec{b}}{|\vec{v}_i|^2}\vec{v}_n$$

With this last expression, it becomes clear that as k increases, the weight given increases for the components corresponding to the largest eigenvalues. In particular, in the case $k \to \infty$:

$$\lim_{k\to\infty} M^k\vec{b} = \frac{\lambda_{max}^k\vec{v}_{max}\cdot\vec{b}}{|\vec{v}_{max}|^2}\vec{v}_{max} = \lambda_{max}^k(\vec{b}\cdot\vec{e}_{max})\vec{e}_{max}$$

where in the last step we used the denominator to write the expression in terms of a unit vector $\vec{e}_{max}$. Hence, if we multiply a large amount of times M to $\vec{b}$, we will end up with the maximum eigenvector, scaled by the maximum eigenvalue of M.

(c) **Define a 50×50 matrix $M_{ij} = 1$ if $|i - j| < 3$ and 0 otherwise, and the vector $(\vec{b_0})i = 1$. Compute the largest eigenvalue, λ_{max}, and eigenvector, $\vec{v}_{max}$, of this matrix (you may use any numerical method)**

The code for finding the eigenvectors on Matlab is the following:

```
[V,D] = eig(M);
%Because D is a diagonal matrix, we only need one index to call it
[max_e, imax_e]=max(max(D));
max_V=V(:,imax_e);
```

The output is the following:

```
max_V =

    0.0166
    0.0268
    0.0393
    0.0507
    0.0623
    0.0735
    0.0845
    0.0952
    0.1055
    0.1154
    0.1249
    0.1340
    0.1425
    0.1505
    0.1580
    0.1649
    0.1711
```

```
    0.1767
    0.1817
    0.1860
    0.1896
    0.1925
    0.1947
    0.1961
    0.1969
    0.1969
    0.1961
    0.1947
    0.1925
    0.1896
    0.1860
    0.1817
    0.1767
    0.1711
    0.1649
    0.1580
    0.1505
    0.1425
    0.1340
    0.1249
    0.1154
    0.1055
    0.0952
    0.0845
    0.0735
    0.0623
    0.0507
    0.0393
    0.0268
    0.0166
```

(d) **Iteratively compute $\vec{b}_k = M\vec{b}_{k-1}/|\vec{b}_{k-1}|$, iterating until $|\vec{b}_k - \vec{b}_{k-1}| < 10^{-6}|\vec{b}_k|$ (with the total number of iterations k_{max}). This is called a power iteration of M. Show that $\vec{b}_{kmax} \propto \vec{v}_{max}$, and that $M\vec{b}_{kmax} \approx \lambda_{max}\vec{b}_{kmax}$.**

The specific code to iterate through this multiplication is:

```
b_old=b;
while 1
    kmax++;
    b_new=M*b_old/norm(b_old);
    if (norm(b_new - b_old) <1.0e-6*norm(b_new))
        break;
    end
    b_old=b_new;
end
```

The output is summarized in figure 6. As it can be seen, the agreement is pretty good with the maximum error around 0.01% after $k_{max} = 307$ iterations.

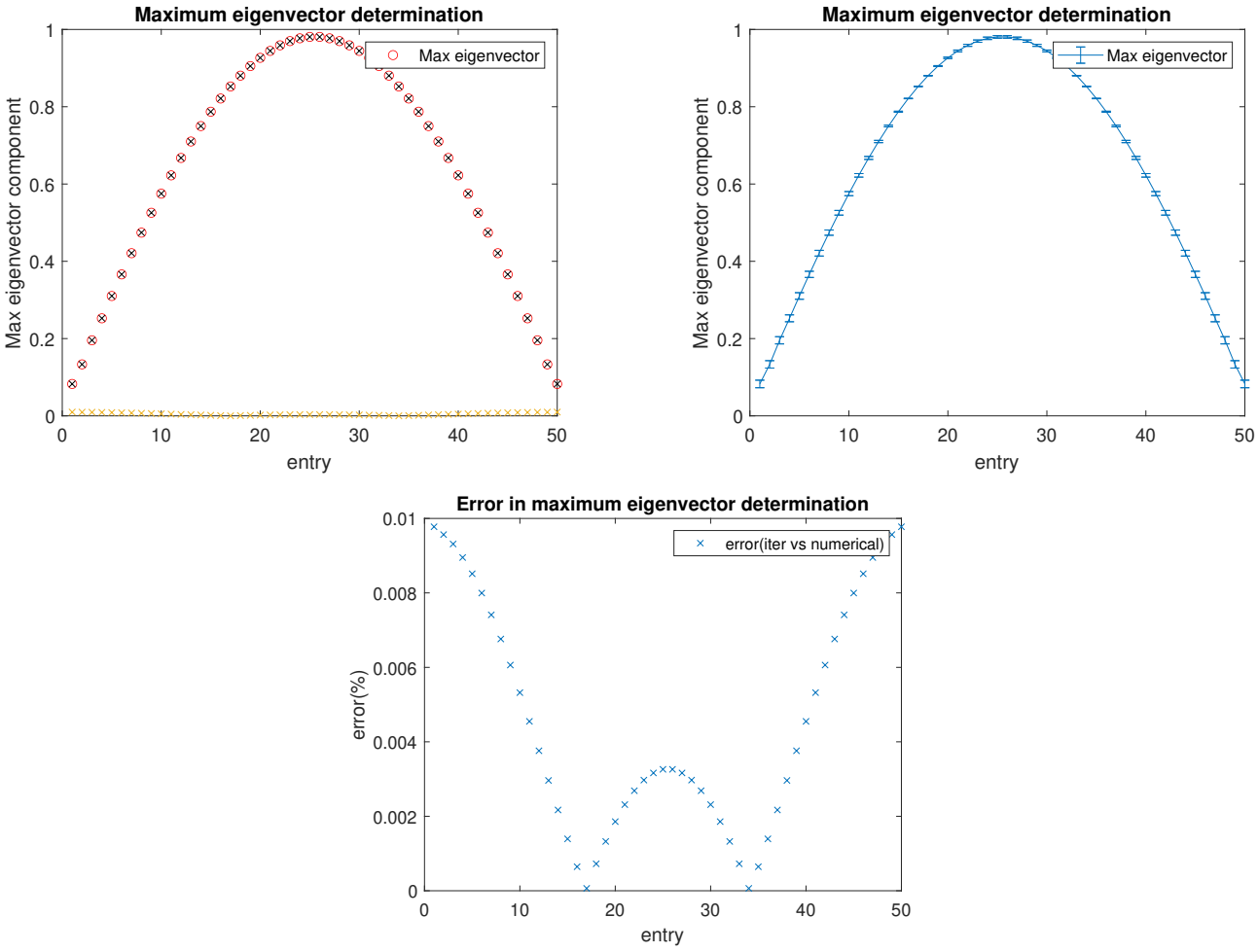

Figure 6: Top Left: Numerical and iterative solution for the largest eigenvector. Top right: Iterative solutions including error bars representing the error between the numerical and iterative solutions. Bottom center: Absolute error between the numerical and iterative solutions.