MATHGM04: Computational And Simulation Method

Exercise 1: Investigating the behaviour of the Brusselator System for various choices of parameters A and B.

Pallu Thapa

Department Of Mathematics

UCL

Tutor

Dr Sergei Timoshin

Part A

Q1.

The Brusselator is a model of certain autocatalytic reactions described in terms of chemical concentrations *x(t)* and *y(t)* where *t* is time. The governing equations are given as:

$$\frac{dx}{dt} = A - Bx + x^2y - x, \qquad (1)$$

$$\frac{dy}{dt} = Bx - x^2y, \qquad (2)$$

with $A, B > 0$ and one equilibrium state at $\left(A, \frac{B}{A}\right)$. We can write the finite-difference fourth-order accurate Runge-Kutta (RK4) approximation of the Brusselator equations on a uniform grid $t_n = nh$, with the time step *h* for *n=0,1,2,....* . Here, We will use the Runge-Kutta algorithm for two dependant variable *x(t)* and *y(t)*. We write the above equations (1) and (2) as

$$\frac{dx}{dt} = A - Bx + x^2y - x = f(x_n, y_n, t_n), x(t_0) = x_0$$

$$\frac{dy}{dt} = Bx - x^2y = g(x_n, y_n, t_n), y(t_0) = y_0$$

Now, We get our approximate solution (x_n, y_n) at time t_n, *n=1,2,....* via the iteration of

$$x_{n+1} = x_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$y_{n+1} = y_n + \frac{1}{6}(l_1 + 2l_2 + 2l_3 + l_4)$$

Where the formulas for each of the *k's* and *l's* are

$$k_1 = hf(x_n, y_n, t_n) = h(A - Bx_n + x_n^2y_n - x_n)$$

$$l_1 = hg(x_n, y_n, t_n) = h(Bx_n - x_n^2y_n)$$

$$k_2 = hf\left(x_n + \frac{k_1}{2}, y_n + \frac{l_1}{2}, t_n + \frac{h}{2}\right) = h[A - B(x_n + \frac{k_1}{2}) + (x_n + \frac{k_1}{2})^2 (y_n + \frac{l_1}{2}) - (x_n + \frac{k_1}{2})]$$

$$l_2 = hg\left(x_n + \frac{k_1}{2}, y_n + \frac{l_1}{2}, t_n + \frac{h}{2}\right) = h[B(x_n + \frac{k_1}{2}) - (x_n + \frac{k_1}{2})^2 (y_n + \frac{l_1}{2})]$$

$$k_3 = hf\left(x_n + \frac{k_2}{2}, y_n + \frac{l_2}{2}, t_n + \frac{h}{2}\right) = h[A - B(x_n + \frac{k_2}{2}) + (x_n + \frac{k_2}{2})^2 (y_n + \frac{l_2}{2}) - (x_n + \frac{k_2}{2})]$$

$$l_3 = hg\left(x_n + \frac{k_2}{2}, y_n + \frac{l_2}{2}, t_n + \frac{h}{2}\right) = h[B(x_n + \frac{k_2}{2}) - (x_n + \frac{k_2}{2})^2 (y_n + \frac{l_2}{2})]$$

$$k_4 = hf(x_n + k_3, y_n + l_3, t_n + h) = h[A - B(x_n + k_3) + (x_n + k_3)^2 (y_n + l_3) - (x_n + k_3)]$$

$$l_4 = hg(x_n + k_3, y_n + l_3, t_n + h) = h[B(x_n + k_3) - (x_n + k_3)^2 (y_n + l_3)]$$

Q2.

We will use the following MATLAB code to implement fourth order Runge-Kutta (RK4) to solve the initial value problem with initial condition $x(0) = x_0$ and $y(0) = y_0$ for some constants $x_0$and y_0.

```
%Solve Brusselator using Runge-Kutta(RK4) Method
%dx/dt = A-Bx+x^2y-x
%dy/dt = Bx-x^2y
clear all
%Problem Setup
A=2;
B=6;
%Defining the functions
fx=@(x,y,t) A-B*x+x^2*y-x;
fy=@(x,y,t) B*x-x^2*y;
%Initial conditions and Step size
x(1)=0;
y(1)=1;
t(1)=0;
h=0.001;
tfinal=100;
N=ceil(tfinal/h);
%Solving using RK-4 Method
for i=1:N
    t(i+1)=t(i)+h;
    k1=h*fx(x(i),y(i),t(i));
    l1=h*fy(x(i),y(i),t(i));
    k2=h*fx(x(i)+k1*1/2,y(i)+l1*1/2,t(i)+h/2);
    l2=h*fy(x(i)+k1*1/2,y(i)+l1*1/2,t(i)+h/2);
    k3=h*fx(x(i)+k2*1/2,y(i)+l2*1/2,t(i)+h/2);
    l3=h*fy(x(i)+k2*1/2,y(i)+l2*1/2,t(i)+h/2);
    k4=h*fx(x(i)+k3,y(i)+l3,t(i)+h);
    l4=h*fy(x(i)+k3,y(i)+l3,t(i)+h);
    x(i+1)=x(i)+1/6*(k1+2*k2+2*k3+k4);
    y(i+1)=y(i)+1/6*(l1+2*l2+2*l3+l4);
end
x(N+1)
y(N+1)
```

This code can be run by typing brusselator.m on MATLAB.

Q3.

For this question, I will compute the solution for parameters *A=2,B=6* and initial condition *x(0)=0, y(0)=1*. I will investigate the solution for different time interval and time steps and decide upon the optimal values to use for this system.

Testing few values for *t* and *h*, I found that time interval between $50 \leq t \leq 100$ is reasonable to capture the behaviour of the system. I also discovered the system is stable for smaller values of h and a reasonable trade off between accuracy and computational effort is to use *h* as order of 10^{-3} . To make this system of choosing values of *t* and *h* somewhat rigorous, I will solve the equation for time interval *t=5,15,30,50,100,1000.* For *t=5*, the computation is too small to demonstrate any qualitative behaviour of the system. As we increase time the oscillatory behaviour of concentration with respect to time start to emerge and at *t=15* we see phase space turn into limit cycle. The phase space for higher values of time looks exactly the same as that for *t=15* and there is no topological change in the behaviour of the system. The time dependent graph of *x(t)* shows a oscillatory pattern which looks like multi-model distribution with peak close to *7*. The time dependent graph of *y(t)* looks like a slanted oscillatory *sine* graph. For the best result the graph of *x(t)* and *y(t)* are best observed between *t=50* and *t=100*. Hence, I have decide to use t=100 for this system. I have included plots of x(t),y(t) and Phase plot for few different values of time intervals.

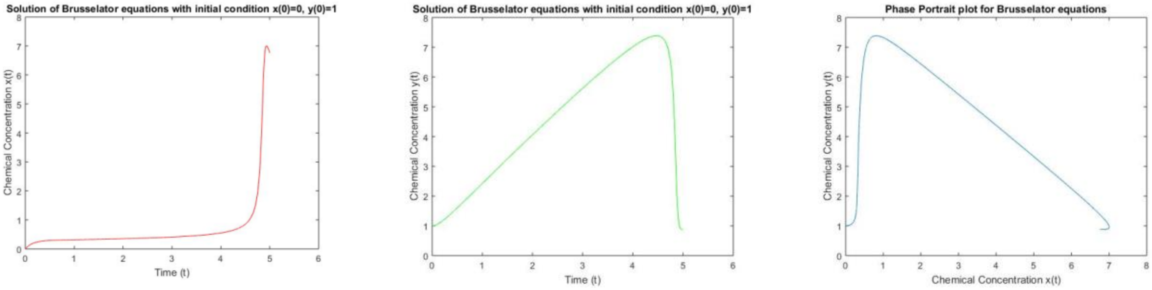


Fig 3.1: At t=5 there is no limit cycle

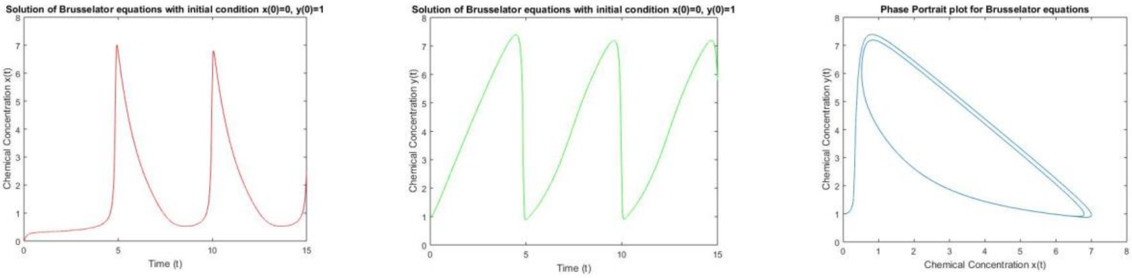


Fig 3.2: At t=15 oscillatory pattern appears and limit cycle also emerges

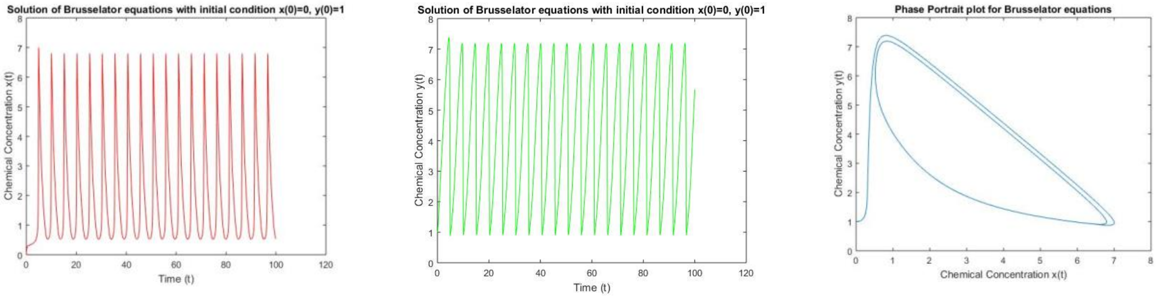


Fig 3.3: At t=100 clear oscillatory pattern appears but there no qualitative change in dynamics of the limit cycle

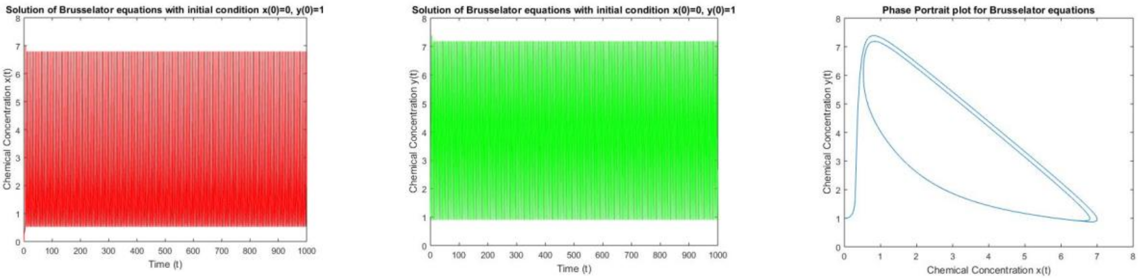


Fig 3.4: At t=1000 it becomes difficult to spot the oscillatory pattern

Picking values for time step h is very critical to stability and accuracy of Runge –Kutta scheme. Theory tells us that for RK4 the error approximately proportional to the step size. i.e. decreasing step size by factor of $\frac{1}{10}$ decreases the error by $\frac{1}{10^4}$ but significantly increases computational time. Since exact analytic solution is not known to measure error correctly and accuracy of our numerical scheme depends on step size, one may implement adaptive methods like runga-kutta Fehlberg. However, for this assignment, I will compute solution for different values of time step (in descending order of factor of 10) and compare the answers at data points. If the errors are sufficiently small and computational time is reasonable then I will pick that value of time step h. The table below shows the error and computational time for different values of *h*.

Time step (h)	$x_i(t)$	*Error* $[x_{i+1}(t) - x_i(t)]$	$y_i(t)$	*Error* $[y_{i+1}(t) - y_i(t)]$	*Runtime (s)*
0.01	*0.538037219*		*5.67572*		*1.5*
0.001	*0.538034875*	*2.34385E-06*	*5.67575*	*3.01502E-05*	*3.35*
0.0001	*0.538034875*	*2.37822E-10*	*5.67575*	*3.05939E-09*	*26.4*
0.00001	*0.538034875*	*2.9976E-15*	*5.67575*	*5.09814E-13*	*253.2*

Table 3.5: Errors and Runtime for different values of time-step h

The above table shows that for *h=0.001* the error was of the magnitude 10^{-6} and the runtime was *3.35* seconds. Anything below this value, the runtime is too long and does not give any qualitative change in the phase space. Also values greater than *0.01* the stability of the system deteriorates. Thus *h=0.001* is a suitable trade off between stability and runtime.

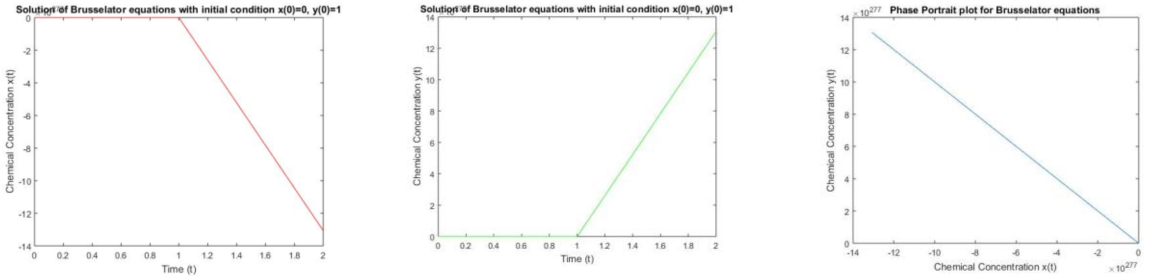


Fig 3.6: At h=1, the solution blows to infinity and we observe instability

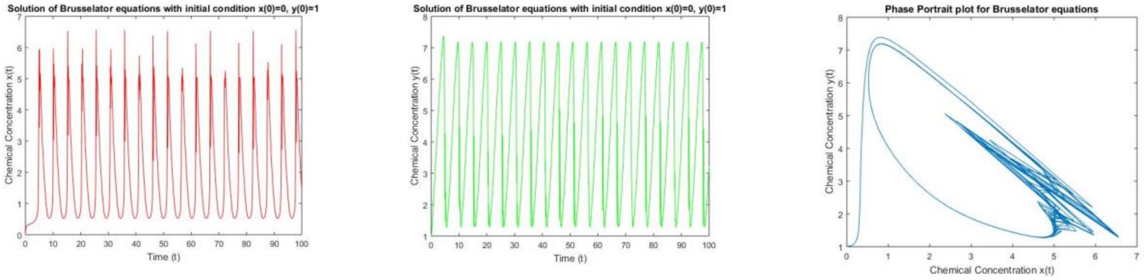


Fig 3.7: At h=0.1, we still observe instability

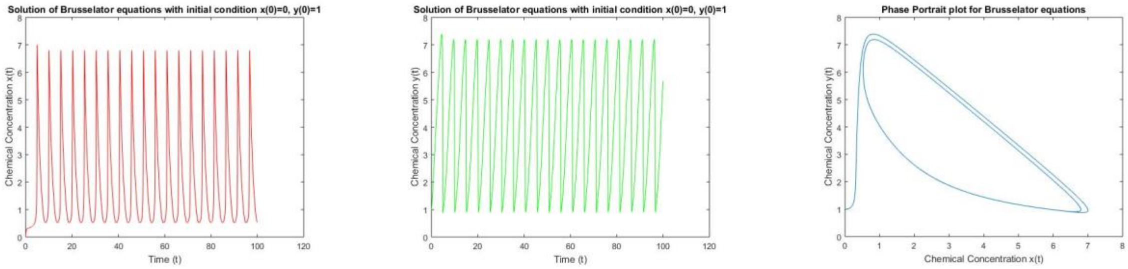


Fig 3.8: At h=0.01, the solution stabilises

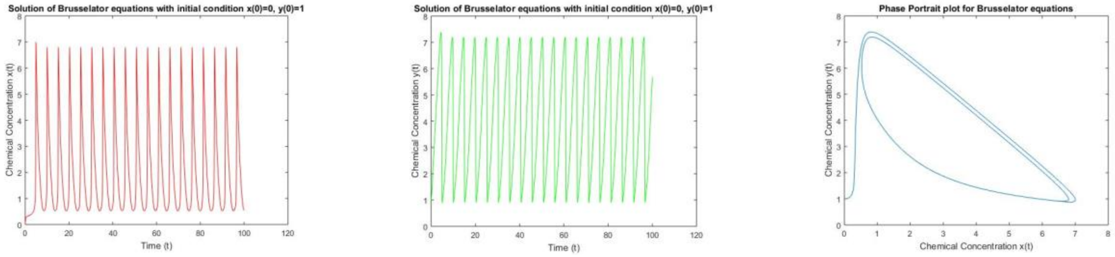


Fig 3.9: At h=0.0001, although we get greater stability there is no topological change in phase space in comparison with h=0.001but with a very long runtime in computation

I have decided to go with *h=0.001* and time interval *t=100* to compute the solution to the brusselator equation. The solution to the IVP gives us *x=0.5380* and *y=5.6758.* I have plotted the three figures below.

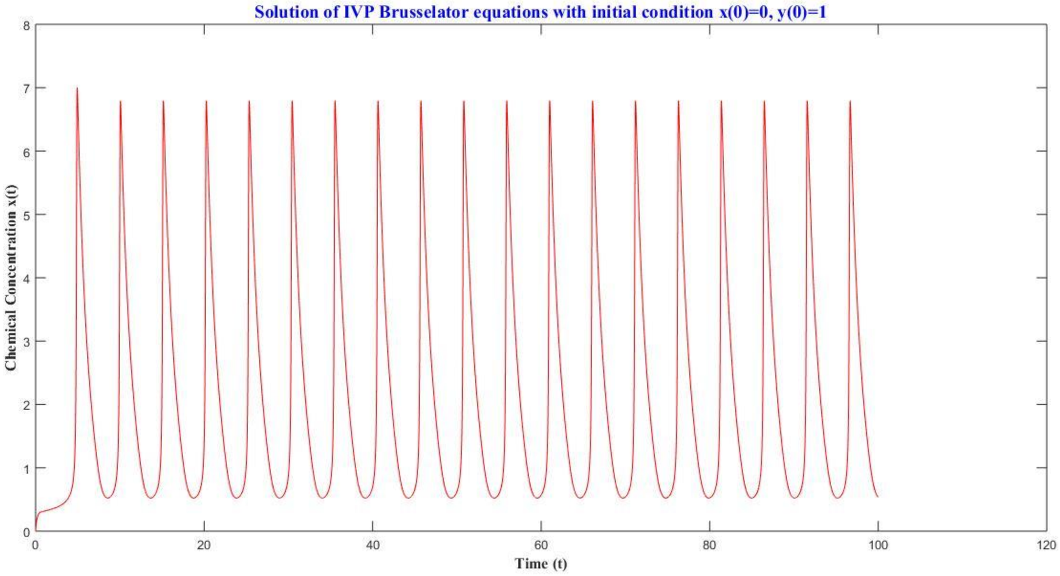


Fig 3.10: x(t) Vs t graph for choice of h=0.001 and t=100

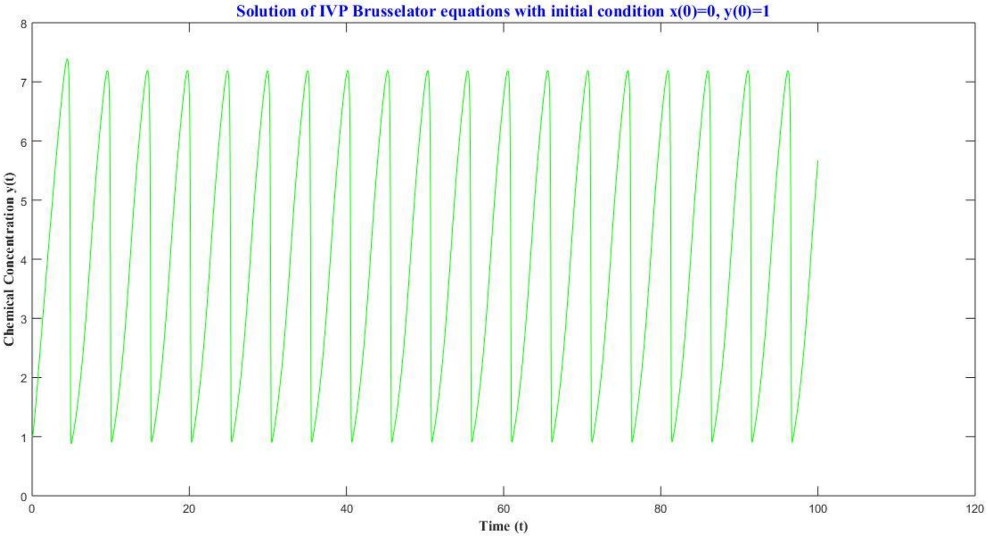


Fig 3.11: y(t) Vs t graph for choice of h=0.001 and t=100

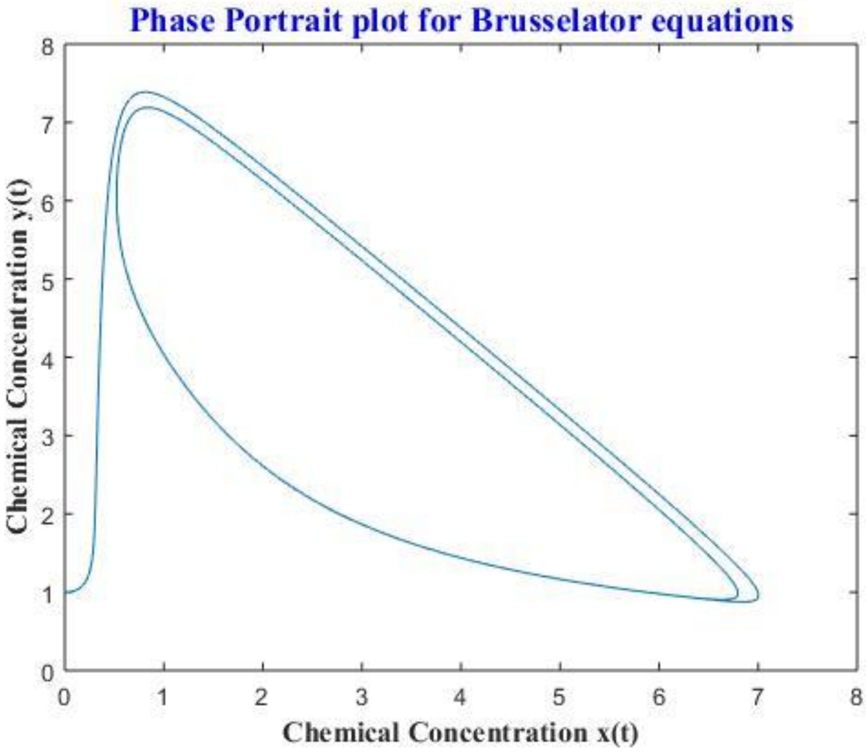


Fig 3.12: Phase Portrait showing limit cycle for h=0.001 and t=100

Q4.

I have tested various initial conditions which either start inside or outside the limit cycle to check whether the limit cycle is a Global Attractor. Some of the points that I have tested are listed below.

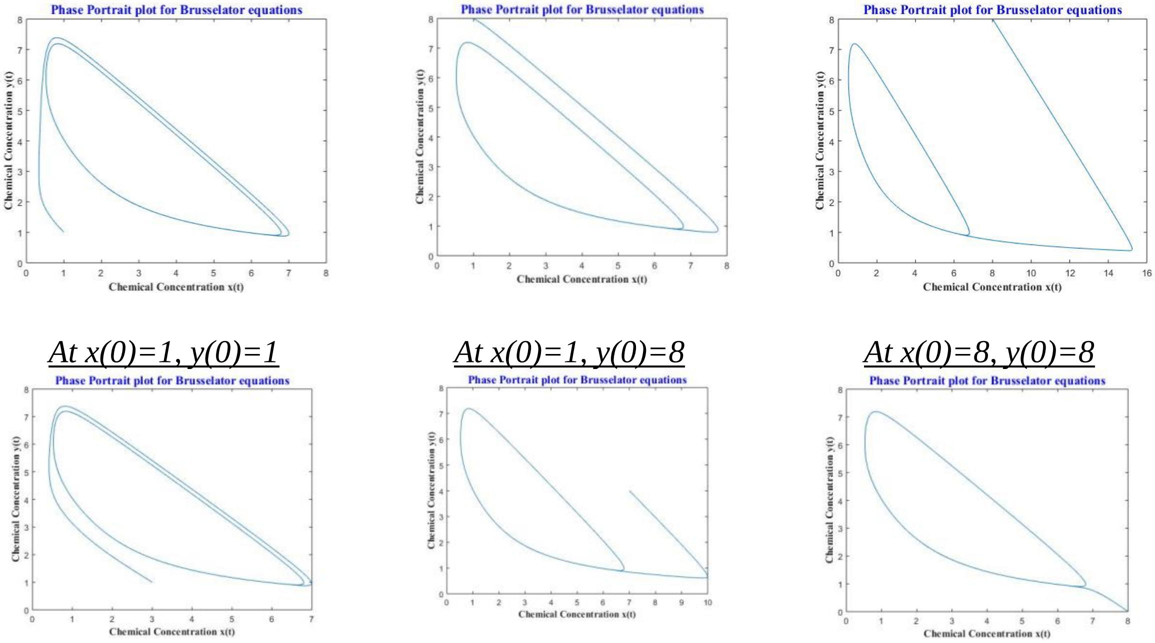
x(0)=1	x(0)=3	x(0)=6	x(0)=7	x(0)=7	x(0)=8	x(0)=8	x(0)=5	x(0)=4	x(0)=2
y(0)=1	y(0)=1	y(0)=0.5	y(0)=3	y(0)=4	y(0)=0	y(0)=8	y(0)=7	y(0)=7	y(0)=8

Table 4.1: Various initial conditions which lie outside the limit cycle

x(0)=1	x(0)=2	x(0)=3	x(0)=4	x(0)=5	x(0)=5	x(0)=6	x(0)=5
y(0)=6	y(0)=5	y(0)= 5	y(0)=4	y(0)=2	y(0)=1	y(0)=2	y(0)=3

Table 4.2: Various initial conditions which lie inside the limit cycle

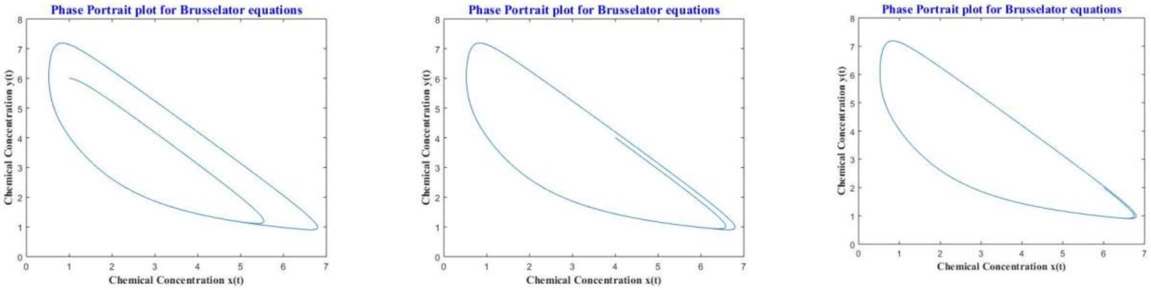
Running the program for various initial conditions listed in Figure 4.1, we observe that as $t \to \infty$, the trajectories becomes asymptotically close to the limit cycle.



At x(0)=1, y(0)=1 At x(0)=1, y(0)=8 At x(0)=8, y(0)=8

At x(0)=3, y(0)=1 At x(0)=7, y(0)=4 At x(0)=8, y(0)=0

Fig 4.3: Points starting outside the limit cycle end up in limit cycle as $t \to \infty$



At x(0)=1, y(0)=6 At x(0)=4, y(0)=4 At x(0)=6, y(0)=2

Fig 4.4: Points starting outside the limit cycle end up in limit cycle as $t \to \infty$

We notice that running the program for the initial values that lie inside the limit cycle we see similar pattern repeating as that for points outside the limit cycle .i.e. the trajectories starting inside the limit cycle become asymptotically close to the cycle and does not leave the limit cycle as $t \to \infty$. An interesting result happens at *x(0)=2, y(0)=3* where the phase space is completely blank. This is the equilibrium point of the system.

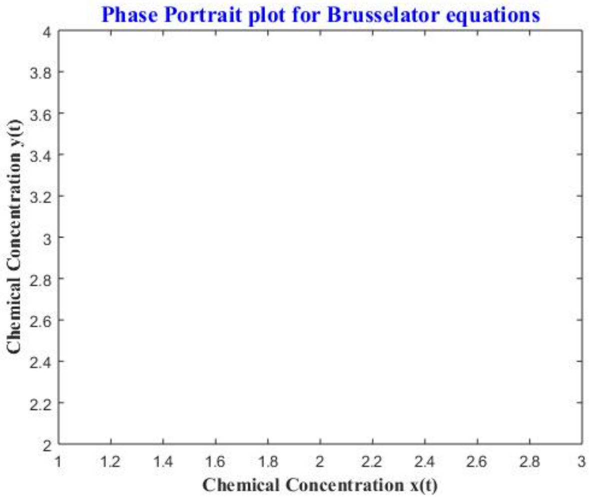


Fig 4.5: At *x(0)=2, y(0)=3 we get a blank phase space*

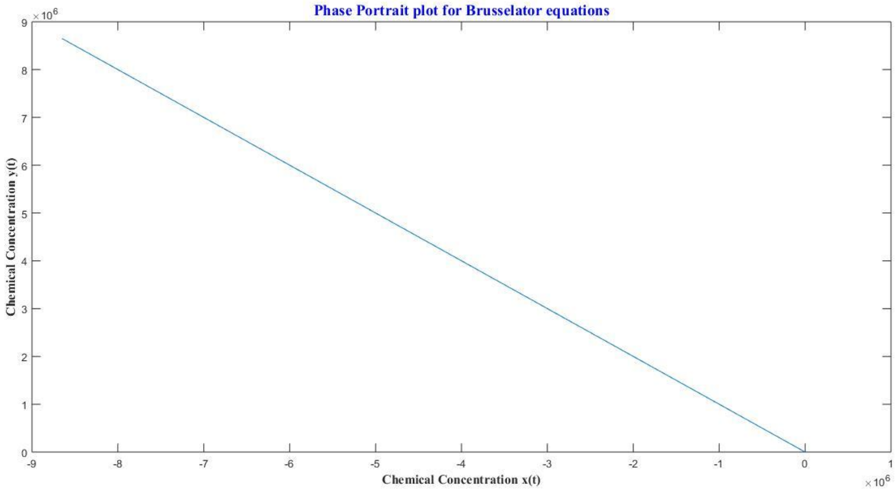


Fig 4.6: At *x(0)=37, y(0)=37, the solution blows up and we get instability. Any values above this tend to blow up*

Experimenting with different initial conditions, we notice that values inside the limit cycle and values that lie outside which are close to the limit cycle end up in limit cycle as time increases. For larger value of initial condition for example *x(0)=37, y(0)=37* we get instability and there is no limit cycle as $t \to \infty$. Hence, We can conclude that for initial conditions which are close to the limit cycle including points that lie inside the cycle (except for the point *x(0)=2, y(0)=3*), we observe that the limit cycle is a global attractor.

Q5.

The accuracy and stability of the numerical scheme is directly linked to the choice of step size *h*. While using RK4 local errors are committed in current step of the algorithm and these tend to accumulate as we continue the iterations. The smaller the step size *h* the smaller the error at a fixed time interval but the algorithm is computationally intensive. So there is a trade off between Error and Runtime of the algorithm which was shown in question 2. In previous question(2), I have tested the system for various values of the time step(h) and found that for values of *h* that are strictly greater than 10^{-2}, the numerical scheme is unstable. For values of h that are smaller than 10^{-2} , the numerical schemes stability and accuracy increases. As shown previously, *h=0.001* gives a balanced choice between stability and runtime. Hence, we can conclude that the stability of the method is not stable for every choice of time step *h*.

We also know from literature that for RK4 to be stable $z_i = \lambda_i h$, (where λ=eigen value) must lie in the methods stability region which is $|1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24}| \leq 1$.

Q6.

Let a=exact value and a_h^*=numerical approximate value using RK4. If the exact value was know (i.e. the exact analytical solution of IVP) then verifying order of accuracy is just a matter of plotting a loglog plot of | a - a_h^*| Vs *h* and finding the slope. Since, we do not have the exact analytic solution of the brusselator equation we need to devise another method.

The numerical method is of order *p* if there is a number *M* such that

$$|a_h^* - a| \leq Mh^p, \qquad (3)$$

where *M* is independent of h and h is sufficiently small. *M* is generally dependent on the exact solution of the differential equation and h^p is also called convergence rate.

I will compute a_h^* for different values of *h* where *h* is halved successively in each iteration. Now, the approach is to look at the ratio of differences between a_h^*. Hence, we get

$$\frac{a_h^* - a_{\frac{h}{2}}^*}{a_{\frac{h}{2}}^* - a_{\frac{h}{4}}^*} = \frac{Mh^p - M(h/2)^p + O(h^{p+1})}{M(h/2)^p - M(h/4)^p + O(h^{p+1})} = \frac{1 - 2^p + O(h)}{2^{-p} - 2^{-2p} + O(h)} = 2^p + O(h),$$

Hence,

$$log_2 \left| \frac{a_h^* - a_{\frac{h}{2}}^*}{a_{\frac{h}{2}}^* - a_{\frac{h}{4}}^*} \right| \longrightarrow 4 \text{ as } h \rightarrow 0, \qquad (4)$$

Below is the table showing the above method in action.

h	a_h^*	$a_h^* - a_{\frac{h}{2}}^*$	$\frac{a_h^* - a_{\frac{h}{2}}^*}{a_{\frac{h}{2}}^* - a_{\frac{h}{4}}^*}$	$log_2\left\|\frac{a_h^* - a_{\frac{h}{2}}^*}{a_{\frac{h}{2}}^* - a_{\frac{h}{4}}^*}\right\|$
0.01	0.538037219	2.20E-06	1.59E+01	3.989879
0.005	0.538035022	1.38E-07	1.59E+01	3.993128
0.0025	0.538034884	8.68E-09	1.60E+01	3.996263
0.00125	0.538034875	5.44E-10	1.60E+01	3.998437
0.000625	0.538034875	3.40E-11	6.33E-11	
0.0003125	0.538034875	5.38E-01		

Table 6.1: Table of values for the solution of Brusselator Equation using RK4. The last column is the final approximation of the order of accuracy p.

The last column in the above Table 6.1 clearly shows the order of accuracy converging to 4 as *h* tends to zero. This verifies that the Runge-Kutta fourth order method we implemented in solving the Brusselator equation is indeed of 4th-order accurate.

Part B

For the parameter values of *A=2 and B=6*, we get an asymptotically stable periodic solution. What this means is that, points starting either inside or outside the limit cycle spiral towards the limit cycle. This implies the system exhibit self-sustained oscillation even in the absence of external periodic forcing. Hence, the solution curve we obtain for concentration with respect to time is a sinusoidal oscillation of constant amplitude.

If we write our equations (1) and (2) as $\dot{x}$=f(x, y); and $\dot{y}$=g(x, y) then the divergence can be calculated as

$$div\,\boldsymbol{f} = \nabla.\boldsymbol{f} = \frac{\partial \mathrm{f}}{\partial \mathrm{x}} + \frac{\partial \mathrm{g}}{\partial \mathrm{y}} \qquad (5)$$

And the Jacobian Matrix of the system can be calculated as

$$J = \begin{pmatrix} \frac{\partial \mathrm{f}}{\partial \mathrm{x}} & \frac{\partial \mathrm{f}}{\partial \mathrm{y}} \\ \frac{\partial \mathrm{g}}{\partial \mathrm{x}} & \frac{\partial \mathrm{g}}{\partial \mathrm{y}} \end{pmatrix} \qquad (6)$$

We are given the equilibrium point of this system at (2,3). So, $\nabla.\boldsymbol{f} = 2xy - x^2 - 7$= Trace=1. This implies the system is dissipative. The Jacobian Matrix for the system is

$$J = \begin{pmatrix} 2xy - 7 & x^2 \\ 6 - 2xy & -x^2 \end{pmatrix}$$

$$J_{(2,3)} = \begin{pmatrix} 5 & 4 \\ -6 & -4 \end{pmatrix}$$

We also get Trace = 1 and Determinant = 4. Looking at the Trace-Determinant diagram below this show the equilibrium is unstable.

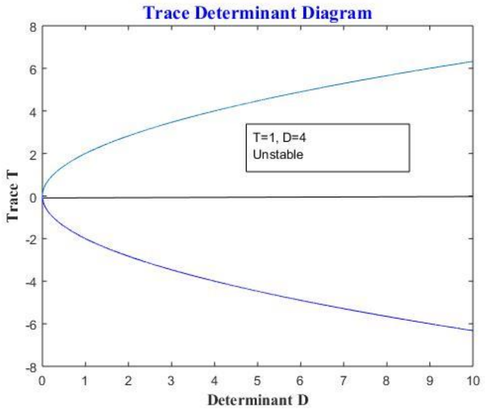


Fig 7.1: Trace-Determinant showing unstable equilibrium

Calculating $T^2 - 4D = -15 < 0 \Longrightarrow$ Equilibrium point is a focus. We can perform further analysis by calculating eigenvalue of the Jacobian matrix to understand the dynamics of the system.

Eigenvalue $(\lambda) = \frac{T \pm\sqrt{T^2-4D}}{2} = \frac{1\pm i\sqrt{15}}{2}$. We know that in order for system to have stable equilibrium, the Real part of eigenvalue must be negative. In this case the real part is positive, so the result matches with the conclusion we got above when performing Trace-Determinant analysis. Since, we have complex conjugate pair of eigenvalue, we know the equilibrium is repelling spiral.

All the analysis above suggest that we get a Asymptotically periodic solution which agree with the results we got in part A of this project.

<u>For the General Case of Parameter Values A and B</u>

The divergence of equation (1) and (2) is $\nabla.\boldsymbol{f} = 2xy - x^2 - B - 1$= Trace. Since, the equilibrium is at *(A,B/A)*, we get the Jacobian matrix of this system as $J_{(A,B/A)} = \begin{pmatrix} B-1 & A^2 \\ -B & -A^2 \end{pmatrix}$. Now, Trace(T)= $B - A^2 - 1$, and Determinant (D)=A^2. Also, $\nabla.\boldsymbol{f} = B - A^2 - 1$.

We know from the literature that if the equilibrium point is a repeller, then we can be assured of existence of a closed orbit by considering a punctured hole inside a region. The repeller drives all the neighbouring trajectories into the region and since the fixed point lie in the hole, the region is free of fixed point and **Poincare-Bendixson Theorem** applies which states that the trajectories must approach a periodic orbit.

From the reasoning given above, it suffices to find conditions under which the equilibrium is a repeller meaning either an unstable node or spiral. Thus, the fixed point is stable for $T < 0$ and unstable for $T > 0$. The dividing line $T = 0$ occurs when $B = A^2 + 1$.

<u>CASE: $B > A^2 + 1$</u>

In this situation, $\nabla.\boldsymbol{f} > 0$ which implies that the system is expanding. Now picking few values of *A* and *B* satisfying $B > A^2 + 1$, I have run the previously used numerical scheme in MATLAB and solve the brusselator equation. The table below shows few values that satisfies the above condition. For some values I get a node and for some I get focus but all are unstable. This suggests equilibrium is a source. I have experimented with different initial points and time scale. I have decided to keep the value of h same as before to maintain numerical stability of the Runge-Kutta scheme in solving the differential equation. I will also keep the Time range same as before. I have change the Initial condition to *x(0)=1* and *y(0)=1*.

A	**B**	**T**	**D**	$\mathbf{T^2 - 4D}$	**Stability**
0.1	2	0.99	0.01	0.9401	Unstable Node
1	3	1	1	-3	Unstable Focus
3	12	2	9	-32	Unstable Focus

Table 7.2: Various values of A and B satisfying $B > A^2 + 1$

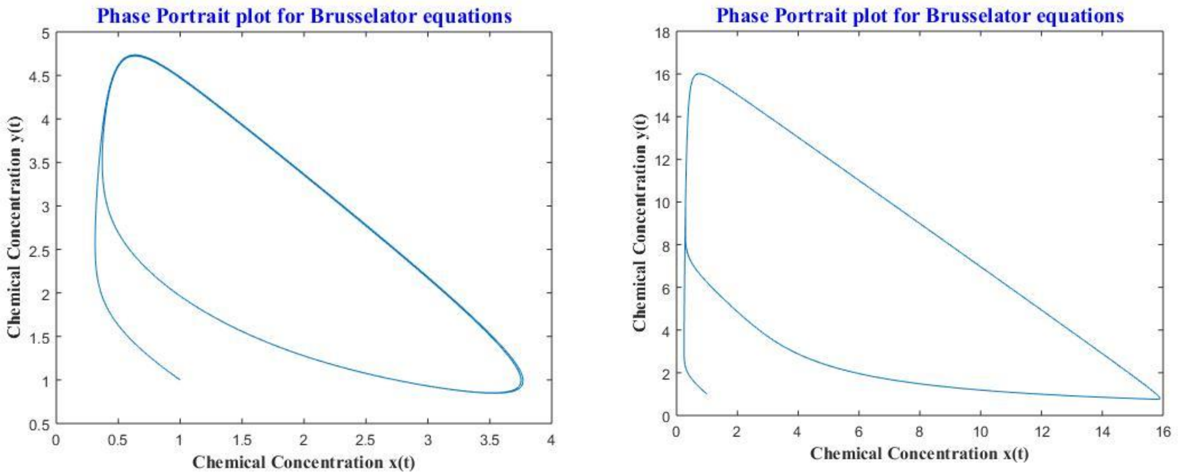


Fig7.3: At *A=1* and *B=3 and A=3 and B=12, we still have a limit cycle*

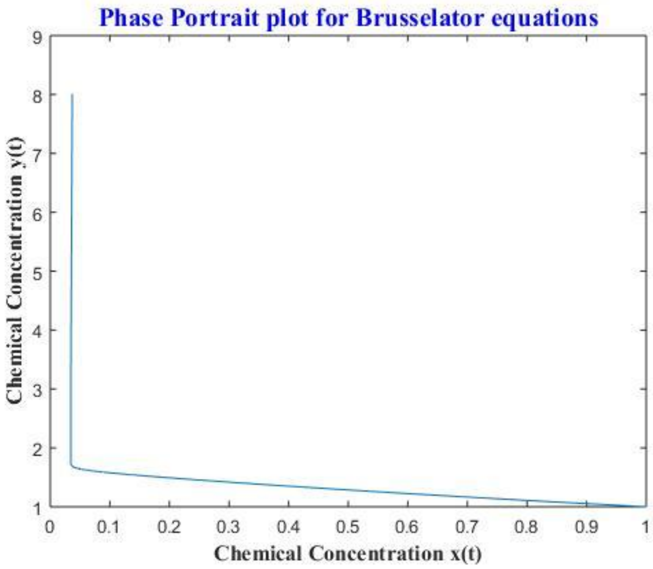


Fig 7.4: At *A=0.1* and *B=2 we don't get a limit cycle*

What we notice here is only when we have a unstable Focus, We obtain a limit cycle. For a node we do not get a limit cycle.

CASE: $B < A^2 + 1$

Here, the divergence $\nabla.\boldsymbol{f} < 0$ which implies that the system is dissipative meaning constant contraction of phase volumes. Here are few values satisfying the condition $B < A^2 + 1$ listed in the table below.

A	B	T	D	$T^2 - 4D$	Stability
1	0.5	-1.5	1	-1.75	Stable Focus
0.5	1	-0.25	0.25	-0.9375	Stable Focus
3	7	-3	9	-27	Stable Focus

Table 7.5: Various values of *A* and *B* satisfying $B < A^2 + 1$

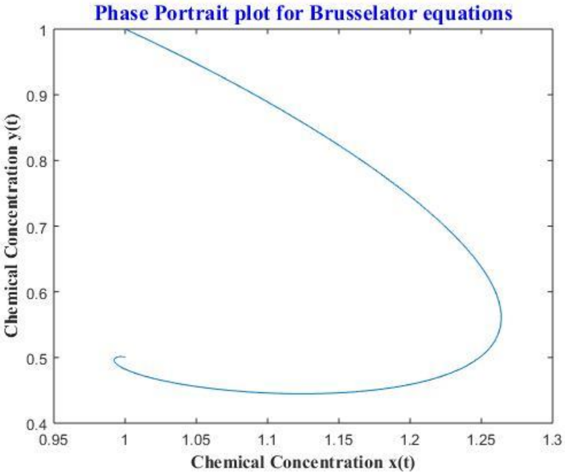


Fig 7.6: At *A=1* and *B=0.5 we get a stable focus/spiral*

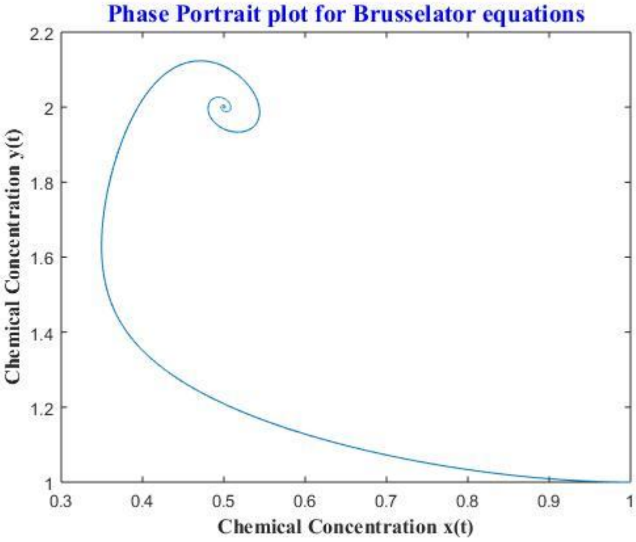


Fig7.7: At *A=0.5*and *B=1 we get a stable focus/spiral*

On Fig 7.7, we observe as time ⟶ ∞ the spiral converges clockwise to the equilibrium point. In this case all the eigenvalues of the Jacobian comes as complex conjugate pairs and thus we do not get any nodes.

CASE: $B = A^2 + 1$

In this case *T=0*, so all the points lie in the Determinant axis on the Trace-Determinant Diagram. These correspond to centre.

A	B	T	D	Stability
1	2	0	1	Centre
3	10	0	9	Centre

Table 7.8: Few values of *A* and *B* satisfying $B = A^2 + 1$

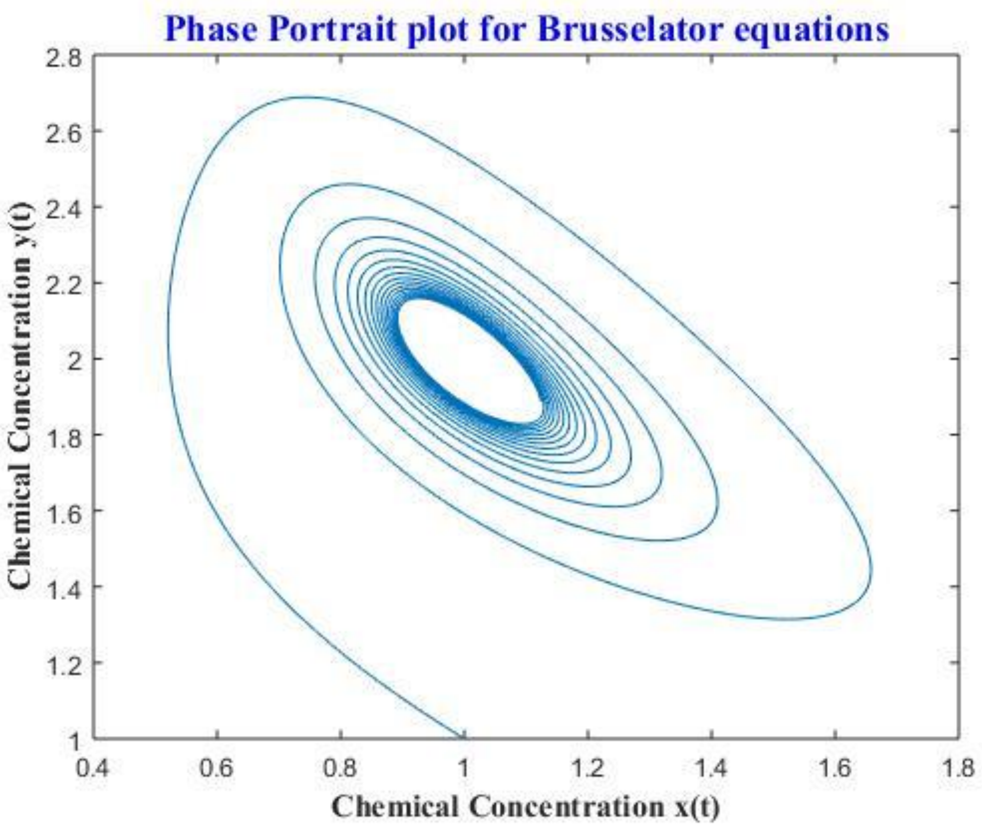


Fig7.9: At *A=1* and *B=2 we get a centre*

When $B = A^2 + 1$, the system becomes conservative and we only get centres. In this case the eigenvalues of the Jacobian Matrix is purely imaginary so we get rotation.

Conclusion:

We observe that for $B > A^2 + 1$ limit cycle exist and solution converges to the equilibrium as time increases. But for $B \leq A^2 + 1$ limit cycle does not exist as time increases.

Appendix

1. The MATLAB code that I have used for the question 3 to plot graphs are listed below:

```
%Solve Brusselator using Runge-Kutta(RK4) Method
%dx/dt = A-Bx+x^2y-x
%dy/dt = Bx-x^2y
clear all
%Problem Setup
A=3;
B=10;
%Defining the functions
fx=@(x,y,t) A-B*x+x^2*y-x;
fy=@(x,y,t) B*x-x^2*y;
%Initial conditions and Step size
x(1)=1;
y(1)=1;
t(1)=0;
h=0.001;
tfinal=100;
N=ceil(tfinal/h);
%Solving using RK-4 Method
for i=1:N
    t(i+1)=t(i)+h;
    k1=h*fx(x(i),y(i),t(i));
    l1=h*fy(x(i),y(i),t(i));
    k2=h*fx(x(i)+k1*1/2,y(i)+l1*1/2,t(i)+h/2);
    l2=h*fy(x(i)+k1*1/2,y(i)+l1*1/2,t(i)+h/2);
    k3=h*fx(x(i)+k2*1/2,y(i)+l2*1/2,t(i)+h/2);
    l3=h*fy(x(i)+k2*1/2,y(i)+l2*1/2,t(i)+h/2);
    k4=h*fx(x(i)+k3,y(i)+l3,t(i)+h);
    l4=h*fy(x(i)+k3,y(i)+l3,t(i)+h);
    x(i+1)=x(i)+1/6*(k1+2*k2+2*k3+k4);
    y(i+1)=y(i)+1/6*(l1+2*l2+2*l3+l4);
end
x(N+1)
y(N+1)
%plotting the solution
figure(1); clf(1)
plot(t,x,'-r')           %plotting time dependent graph for x(t)
xlabel('Time (t)','FontName','Times New
Roman','FontSize',12,'FontWeight','bold')
ylabel('Chemical Concentration x(t)','FontName','Times New
Roman','FontSize',12,'FontWeight','bold')
title('Solution of IVP Brusselator equations with initial condition x(0)=0,
y(0)=1','FontName','Times New
Roman','FontSize',14,'FontWeight','bold','Color','b')
hold on
figure(2); clf(2)
plot(t,y,'-g')           %plotting time dependent graph for y(t)
xlabel('Time (t)','FontName','Times New
Roman','FontSize',12,'FontWeight','bold')
ylabel('Chemical Concentration y(t)','FontName','Times New
Roman','FontSize',12,'FontWeight','bold')
title('Solution of IVP Brusselator equations with initial condition x(0)=0,
y(0)=1','FontName','Times New
Roman','FontSize',14,'FontWeight','bold','Color','b')
```

```
figure(3); clf(3)
plot(x,y)            %plotting the phase portrait in phase plane
xlabel('Chemical Concentration x(t)','FontName','Times New
Roman','FontSize',12,'FontWeight','bold')
ylabel('Chemical Concentration y(t)','FontName','Times New
Roman','FontSize',12,'FontWeight','bold')
title('Phase Portrait plot for Brusselator equations','FontName','Times New
Roman','FontSize',14,'FontWeight','bold','Color','b')
hold off
```

The MATLAB code that I have for plotting the Trace-Determinant Diagram is given below:

```
D=[0:0.001:10];
plot(D,sqrt(4*D),D,-sqrt(4*D),'-b')
xlabel('Determinant D','FontName','Times New
Roman','FontSize',12,'FontWeight','bold')
ylabel('Trace T','FontName','Times New
Roman','FontSize',12,'FontWeight','bold')
title('Trace Determinant Diagram','FontName','Times New
Roman','FontSize',14,'FontWeight','bold','Color','b')
```