

① Model of fishery: $\frac{dx}{dt} = x(1-x) - \frac{hx}{a+x}$; $h, a > 0$.

a) Show that the system can have 1, 2 or 3 fixed points, depending on the values of a, h . Classify their stability.

$$f(x) = x - x^2 - \frac{hx}{a+x}$$

Fixed point: $f(x) = 0 \Leftrightarrow x(1-x) - \frac{hx}{a+x} = 0$.

$$x(1-x - \frac{h}{a+x}) = 0$$

$$x \left(\frac{(1-x)(a+x) - h}{a+x} \right) = 0 \rightarrow \begin{cases} x_0 = 0 - \text{the first root} \\ a+x - ax - x^2 - h = 0 \end{cases}$$

$$x^2 + x(a-1) + h-a = 0$$

$$\Rightarrow D = (a-1)^2 - 4(h-a) = a^2 - 2a + 1 - 4h + 4a = (a+1)^2 - 4h$$

(5) • if $D > 0$, i.e. $|a+1| > 2\sqrt{h}$, i.e. $\begin{cases} a+1 > 2\sqrt{h} - \text{yes, can be} \\ a+1 < -2\sqrt{h} - \text{never can be, since } a > 0 \end{cases}$ and $x_0 = 0$.

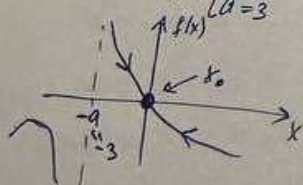
We have 3 roots: $x_{1,2} = \frac{1-a \pm \sqrt{(a+1)^2 - 4h}}{2}$.

(12) • if $D = 0$, i.e. $|a+1| = 2\sqrt{h} \Leftrightarrow a = 2\sqrt{h} - 1$, we have 2 roots: $x_{1,2} = \frac{1-a}{2}$ and $x_0 = 0$.

(11) • if $D < 0$, i.e. $|a+1| < 2\sqrt{h}$, we have 1 root: $x_0 = 0$.

Let's analyse stability of x_0, x_1, x_2 :

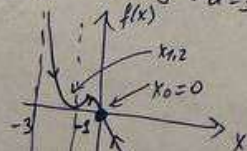
(1) $h > \frac{(a+1)^2}{4} \Rightarrow 1 \text{ root } (x_0 = 0) \text{ - stable}$ for example, $\begin{cases} h = 5 \\ a = 3 \end{cases}$



in this case x_0 - stable (from visual stability analysis, based on the sign of $f(x)$ for every x)

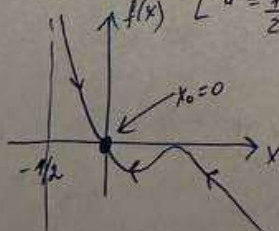
(2) $h = \frac{(a+1)^2}{4} \Rightarrow 2 \text{ roots}$
 $\begin{cases} x_0 = 0 - \text{stable} \\ x_{1,2} = \frac{1-a}{2} - \text{half-stable} \end{cases}$

2.1) $a > 1$, e.g. $\begin{cases} h = 4 \\ a = 3 \end{cases}$



$\Rightarrow x_0 = 0$ is stable
 $x_{1,2}$ - is half-stable

2.2) $a < 1$, e.g. $\begin{cases} h = \frac{9}{16} \\ a = \frac{1}{2} \end{cases}$



$\Rightarrow x_0 = 0$ is stable,
 $x_{1,2}$ is half-stable

(3) $h < \frac{(a+1)^2}{4} \Rightarrow 3 \text{ roots}$

$\begin{cases} x_0 = 0 \leftarrow \text{stable when } a < h, \text{ unstable when } a > h \\ x_1 = \frac{1-a - \sqrt{(a+1)^2 - 4h}}{2} \leftarrow \text{unstable when } a < h, \text{ stable when } a > h \\ x_2 = \frac{1-a + \sqrt{(a+1)^2 - 4h}}{2} \leftarrow \text{stable if } a, h > 0 \end{cases}$

How we analyse stability?

Let's understand how these x_0, x_1, x_2 are located with respect to zero.

3.1) when x_1 (and hence x_2) is greater than 0?

This happens when

$$\frac{1-a - \sqrt{(a+1)^2 - 4h}}{2} > 0$$

$$\frac{(a+1)^2 - 4h < (1-a)^2}{a < 1} \Leftrightarrow a < h$$

$\Rightarrow x_0$ and x_2 are stable
 x_1 is unstable

3.2) when $x_1 > 0$, and $x_2 < 0$?

This happens when

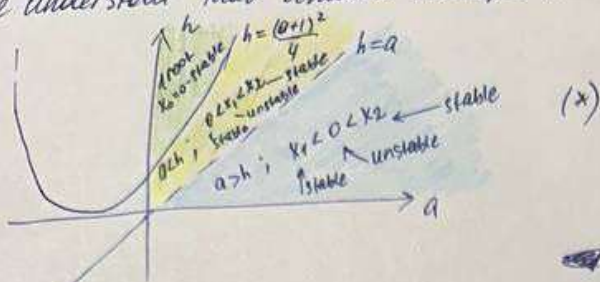
$$\frac{1-a + \sqrt{(a+1)^2 - 4h}}{2} > 0$$

$$(a+1)^2 - 4h > (1-a)^2 \Leftrightarrow a > h \Rightarrow x_1 \text{ and } x_2 - \text{stable}, x_0 \text{ is unstable}$$

3.3) when $a = h$, then $x_{1,2} = 0$

$\Rightarrow \begin{cases} 0 < a < 1 \\ a > 1 \end{cases}$
 $\begin{cases} x_3 - \text{stable} \\ x_0 - \text{half-stable} \end{cases}$ or $\begin{cases} x_3 - \text{stable} \\ x_0 - \text{half-stable} \end{cases}$

So we understood that behaviour changes when we reach curves $h=0$ and $h=\frac{(a+1)^2}{4}$



b) Analyse the dynamics near $x=0$ and show that a bifurcation occurs when $h=0$. What type of bifurcation is it?

To analyse stability at $x=0$, let's look at $f'(0)$.

$$f(x) = x - x^2 - \frac{hx}{a+x}$$

$$f'(x) = 1 - 2x - h \cdot \frac{1 + (a+x) - 1 \cdot x}{(a+x)^2} = 1 - 2x - \frac{ha}{(a+x)^2}$$

$$\Rightarrow f'(0) = 1 - \frac{ha}{a^2} = 1 - \frac{h}{a} = \frac{a-h}{a} \rightarrow \begin{cases} > 0, \text{ when } a > h \Rightarrow \text{unstable} \\ < 0, \text{ when } a < h \Rightarrow \text{stable} \end{cases}$$

When $h=a$, linear analysis is insufficient and we need larger order derivative.

$$f''(x) = -2 + \frac{2ha}{(a+x)^3} \xrightarrow[h=a]{x=0} -2 + \frac{2a^2}{a^3} = -2 + \frac{2}{a} = -2\left(1 - \frac{1}{a}\right) \text{ - may be } < 0 \text{ or } > 0, \text{ depending on } a.$$

$$f'''(x) = -\frac{6 \cdot ha}{(a+x)^4} \xrightarrow[h=a]{x=0} -\frac{6a^2}{a^4} = -\frac{6}{a^2} < 0, \forall a > 0.$$

When $a > 1$, $f''(0) < 0 \Rightarrow x_0$ - stable

When $a < 1$, $f''(0) > 0 \Rightarrow x_0$ - unstable

When $a=1$, then

$$f(x) = x(1-x) - \frac{x}{1+x} = \frac{x(1-x^2-1)}{1+x} = \frac{-x^3}{1+x}$$

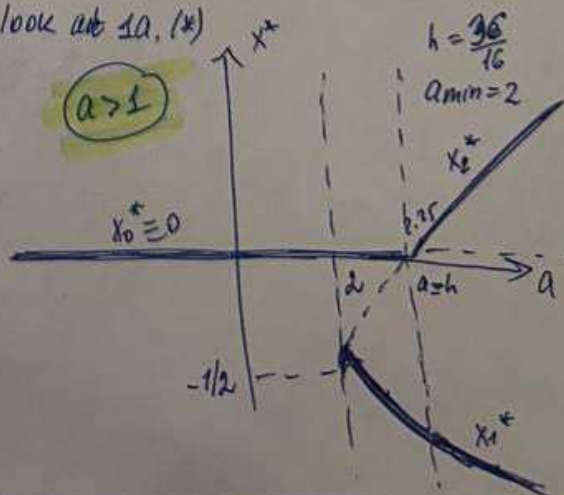
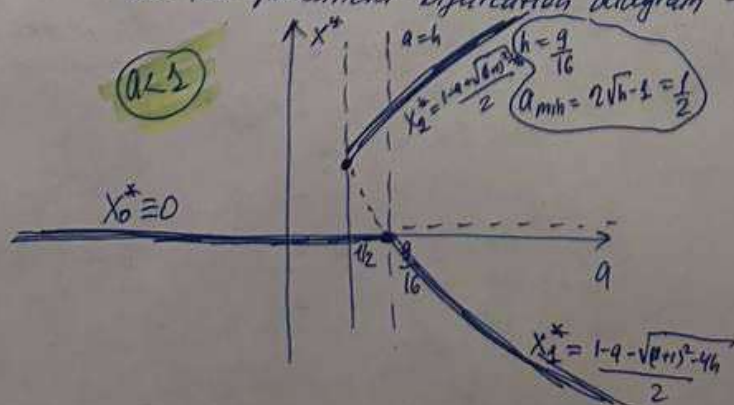
This bifurcation is transcritical - it will be seen at the graph from 1d, that there meet 4 curves.

c) Show that another bifurcation occurs when $h=\frac{(a+1)^2}{4}$. Classify this bifurcation.

We showed that at $h=\frac{(a+1)^2}{4}$ we start have 3 roots instead of 1 root or vice versa.

This is a saddle node bifurcation, because if we are by one side of the point, we have one point, and if we are by the other side of the curve, we have 3.

d) Sketch the two-parameter bifurcation diagram - look at 1d, (*)



(2) Consider $\ddot{\theta} + 2a\dot{\theta} + \omega^2 \sin \theta = 0$, $a > 0$
 ω^2

(2)

a) Rewrite this model as a 2-D system introducing x_1 and x_2 .

b) Find fixed points and classify their stability.

c) Sketch a phase portrait with nullclines for $\omega > a$.

d) Numerically solve and provide plot.

Introduce $\dot{x} = y$ (where y is $\dot{\theta}$)
 $\Rightarrow \dot{y} = -2ay - \omega^2 \sin x$

fixed points: $\begin{cases} y=0 \\ -2ay - \omega^2 \sin x = 0 \end{cases} \Rightarrow \begin{cases} y=0 \\ \sin x = 0 \end{cases} \Rightarrow \begin{cases} y=0 \\ x = \pi k, k \in \mathbb{N} \end{cases}$

Analyse stability: use Jacobian

$$J = \begin{pmatrix} \frac{\partial \dot{x}}{\partial x} & \frac{\partial \dot{x}}{\partial y} \\ \frac{\partial \dot{y}}{\partial x} & \frac{\partial \dot{y}}{\partial y} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\omega^2 \cos x & -2a \end{pmatrix} \bigg|_{x^* = \pi k} = \begin{pmatrix} 0 & 1 \\ -\omega^2 & -2a \end{pmatrix}$$

Find eigenvalues:

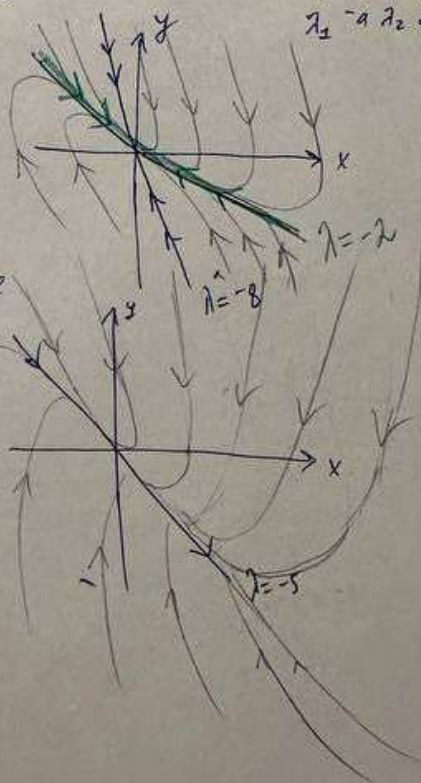
$$(A - \lambda E) = \begin{pmatrix} -\lambda & 1 \\ -\omega^2 & -2a - \lambda \end{pmatrix} \Rightarrow \lambda(2a + \lambda) + \omega^2 = 0$$

$$\lambda^2 + 2a\lambda + \omega^2 = 0$$

$$\Rightarrow D = 4a^2 - 4\omega^2 = 4(a^2 - \omega^2)$$

(1) • e.g. $a > \omega \Rightarrow \lambda_{1,2} = \frac{-2a \pm \sqrt{4a^2 - 4\omega^2}}{2} = -a \pm \sqrt{a^2 - \omega^2} < 0$
 \Rightarrow this is a stable node

e.g. $a = 5$, $\omega = 4$
 $\lambda_{1,2} = \frac{-10 \pm 6}{2} = -2$
 $\lambda_1 = (-1; 8)$
 $\lambda_2 = (-1; 2)$



(2) if $a = \omega$, then $\lambda_1 = \lambda_2 = -a$ - one eigenvalue

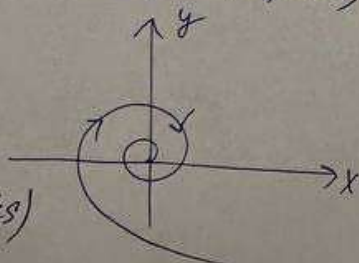
\Rightarrow this is a stable degenerate node

e.g. $a = 5$, $\omega = 5$
 $\lambda = -5$; $\lambda_1 = (1; -5)$

(3) if $a < \omega \Rightarrow \lambda_{1,2} = -a \pm i\sqrt{\omega^2 - a^2}$ - complex

\Rightarrow this is a stable spiral (focus)

e.g. $a = 4$, $\omega = 5$



(or $a = 4$, $\omega = 5$ also works)

! nullclines are $\begin{cases} y=0 \\ -\omega^2 x - 2ya = 0 \Leftrightarrow y = -\frac{\omega^2}{2a} x \end{cases}$

```
In [133]: import numpy as np
import matplotlib.pyplot as plt
from scipy import optimize
```

Q1

**Let's plot the bifurcation diagram (x^* vs a)
for two fixed values of $h = \frac{9}{16}, \frac{36}{16}$**

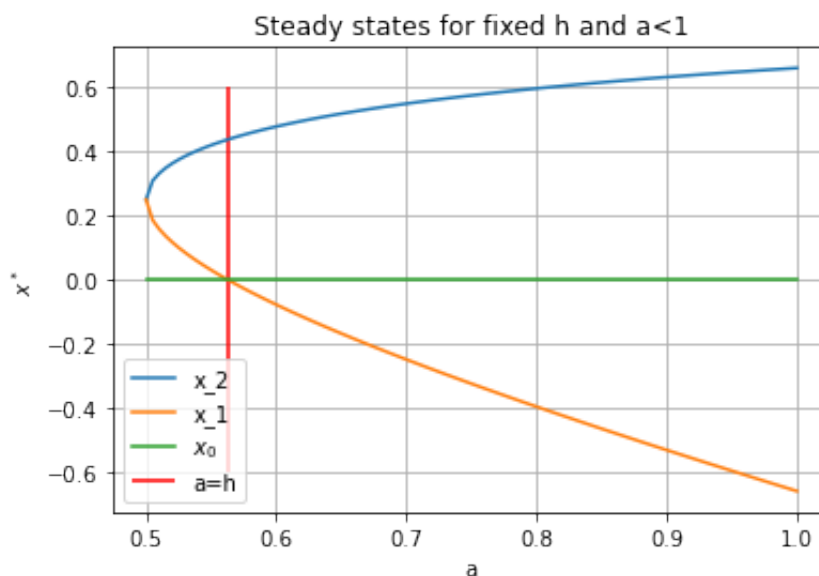
```
In [446]: def x_plus(a,h):
            return 0.5*(1-a+np.sqrt((a+1)**2-4*h))
def x_minus(a,h):
            return 0.5*(1-a-np.sqrt((a+1)**2-4*h))

h=9/16
a_min=2*np.sqrt(h)-1
print("a_min=",a_min)

mas_a=[a for a in np.linspace(a_min, 2*a_min,100)]
mas_x_plus=[x_plus(a,h) for a in mas_a]
mas_x_minus=[x_minus(a,h) for a in mas_a]
mas_x_zero=[0 for a in mas_a]

plt.plot(mas_a,mas_x_plus,label='x_2')
plt.plot(mas_a,mas_x_minus,label='x_1')
plt.plot(mas_a,mas_x_zero,label='$x_0$')
plt.xlabel('a')
plt.ylabel('$x^*$')
plt.title('Steady states for fixed h and a<1')
plt.vlines(h,ymin=-0.6, ymax=0.6,color='red',label='a=h')
plt.grid()
plt.legend()
plt.show()
```

a_min= 0.5



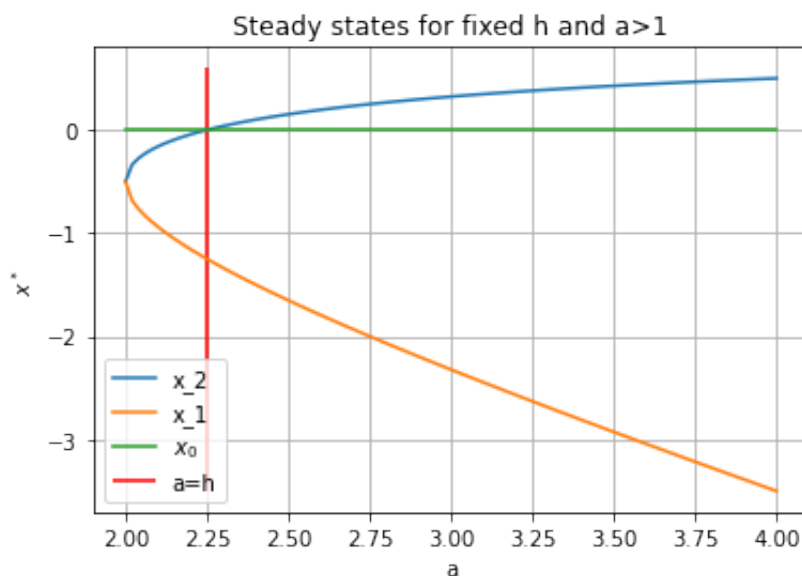
```
In [449]: def x_plus(a,h):
            return 0.5*(1-a+np.sqrt((a+1)**2-4*h))
def x_minus(a,h):
            return 0.5*(1-a-np.sqrt((a+1)**2-4*h))

h=36/16
a_min=2*np.sqrt(h)-1
print("a_min=",a_min)

mas_a=[a for a in np.linspace(a_min, 2*a_min,100)]
mas_x_plus=[x_plus(a,h) for a in mas_a]
mas_x_minus=[x_minus(a,h) for a in mas_a]
mas_x_zero=[0 for a in mas_a]

plt.plot(mas_a,mas_x_plus,label='x_2')
plt.plot(mas_a,mas_x_minus,label='x_1')
plt.plot(mas_a,mas_x_zero,label='$x_0$')
plt.xlabel('a')
plt.ylabel('$x^*$')
plt.title('Steady states for fixed h and a>1')
plt.vlines(h,ymin=-3.5, ymax=0.6,color='red',label='a=h')
plt.grid()
plt.legend()
plt.show()
```

a_min= 2.0



Q2

Let's draw sample phase portraits for cases (1),(2) and (3) from the handwritten solutions and validate that they really look like stable node, stable degenerate node and stable spiral.

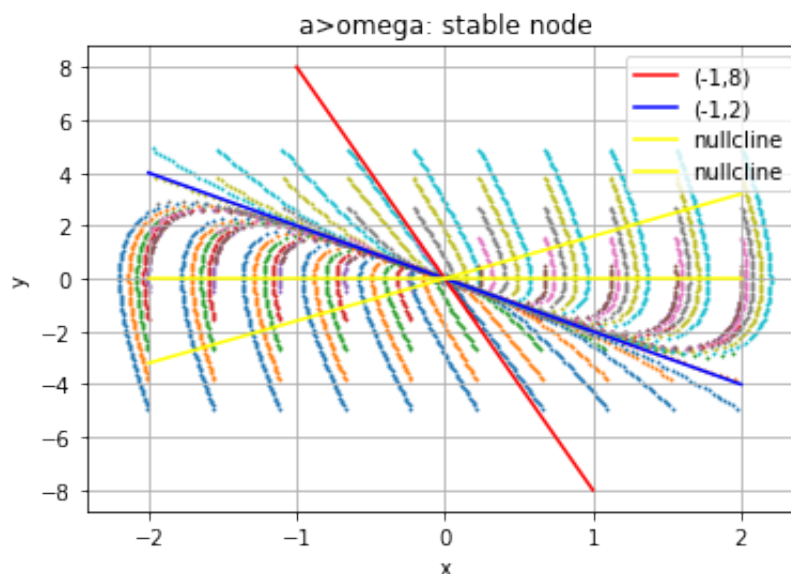
```

In [432]: # case 1:  $a > \omega \Rightarrow$  stable node - yes
a=5
omega=4
def f(x,y):
    return y
def g(x,y):
    return -omega**2*x -2*a*y

k=0.1

for x0 in np.linspace(-2,2,10):
    for y0 in np.linspace(-5,5,10):
        masx=[x0]
        masy=[y0]
        #plt.scatter(x0,y0,color='red')
        #print(x,y)
        for i in range(200):
            x,y=masx[i],masy[i]
            u,v=f(x,y),g(x,y)
            dx,dy=k*u/np.sqrt(u**2 + v**2),k*v/np.sqrt(u**2 + v**2)
            #print(dx,dy)
            masx.append(x+dx)
            masy.append(y+dy)
        #plt.arrow(x=x,y=y,dx=u/np.sqrt(u**2 + v**2),dy=v/(u**2 + v**2))
        plt.scatter(masx[1:],masy[1:],s=0.8)
plt.plot([-t for t in np.linspace(-1,1)], [8*t for t in np.linspace(-1,1)])
plt.plot([-t for t in np.linspace(-2,2)], [2*t for t in np.linspace(-2,2)])
plt.plot([-t for t in np.linspace(-2,2)], [0 for t in np.linspace(-2,2)])
plt.plot([-t for t in np.linspace(-2,2)], [-omega**2/(2*a)*t for t in np.linspace(-2,2)])
plt.xlabel('x')
plt.ylabel('y')
plt.title('a>omega: stable node')
plt.grid()
plt.legend()
plt.show()

```




```

In [435]: # case 2: a =omega => stable degenerate node - yes
a=5
omega=5
def f(x,y):
    return y
def g(x,y):
    return -omega**2*x -2*a*y

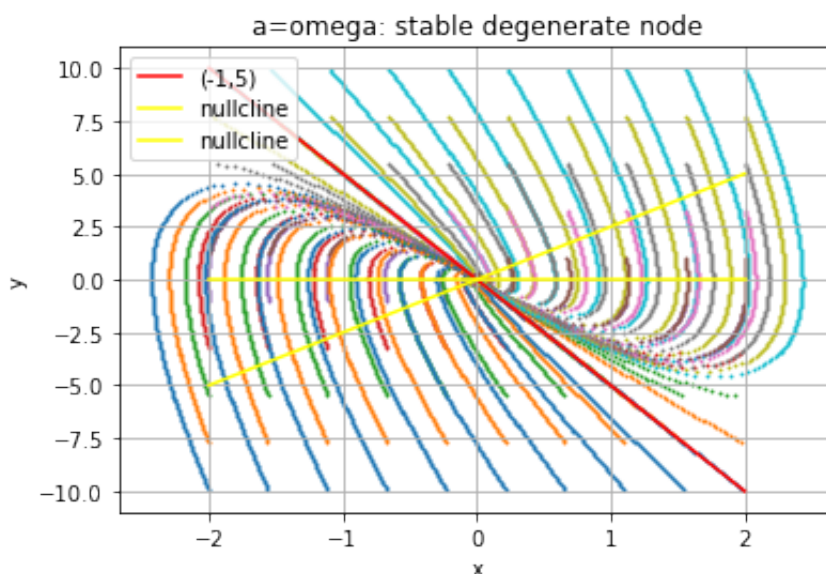
k=0.1

for x0 in np.linspace(-2,2,10):
    for y0 in np.linspace(-10,10,10):
        masx=[x0]
        masy=[y0]
        #plt.scatter(x0,y0,color='red')
        #print(x,y)
        for i in range(200):
            x,y=masx[i],masy[i]
            u,v=f(x,y),g(x,y)
            dx,dy=k*u/np.sqrt(u**2 + v**2),k*v/np.sqrt(u**2 + v**2)
            #print(dx,dy)
            masx.append(x+dx)
            masy.append(y+dy)
        #plt.arrow(x=x,y=y,dx=u/np.sqrt(u**2 + v**2),dy=v/(u**2 + v
        plt.scatter(masx[1:],masy[1:],s=0.8)

plt.plot([-t for t in np.linspace(-2,2)], [5*t for t in np.linspace(
plt.plot([-t for t in np.linspace(-2,2)], [0 for t in np.linspace(-2
plt.plot([-t for t in np.linspace(-2,2)], [-omega**2/(2*a)*t for t i

plt.xlabel('x')
plt.ylabel('y')
plt.title('a=omega: stable degenerate node')
plt.grid()
plt.legend()
plt.show()

```



```

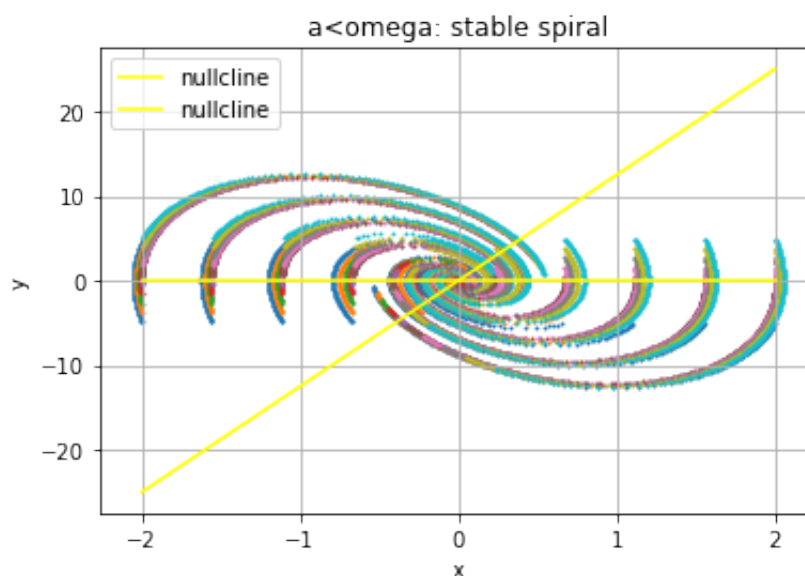
In [438]: # case 3:  $a < \omega \Rightarrow$  stable spiral
a=4
omega=10
def f(x,y):
    return y
def g(x,y):
    return -omega**2*x -2*a*y

k=0.1

for x0 in np.linspace(-2,2,10):
    for y0 in np.linspace(-5,5,10):
        masx=[x0]
        masy=[y0]
        #plt.scatter(x0,y0,color='red')
        #print(x,y)
        for i in range(200):
            x,y=masx[i],masy[i]
            u,v=f(x,y),g(x,y)
            dx,dy=k*u/np.sqrt(u**2 + v**2),k*v/np.sqrt(u**2 + v**2)
            #print(dx,dy)
            masx.append(x+dx)
            masy.append(y+dy)
            #plt.arrow(x=x,y=y,dx=u/np.sqrt(u**2 + v**2),dy=v/(u**2 + v**2))
            plt.scatter(masx[1:],masy[1:],s=0.8)
plt.plot([-t for t in np.linspace(-2,2)], [0 for t in np.linspace(-2,2)])
plt.plot([-t for t in np.linspace(-2,2)], [-omega**2/(2*a)*t for t in np.linspace(-2,2)])

plt.xlabel('x')
plt.ylabel('y')
plt.title('a<omega: stable spiral')
plt.grid()
plt.legend()
plt.show()

```



And we see that yes, for case (3) point $(0, 0)$ really looks like a stable spiral.

Q3

3a) Plot phase portraits for $\mu = 0.5, 0.8, 1.4$

Let's denote $x := c$, $y := h$, plug in the needed variables (and make sure that the dimensionalities for both sides really coincide - I checked this); we will get the following system: $\dot{x} = f(x, y, \mu)$, $\dot{y} = g(x, y, \mu)$, where

$$f(x, y, \mu) = \mu 8.1 y \frac{0.111 + 0.889x}{0.7 + x} - \frac{2x}{0.1 + x} = 0.02, \quad g(x, y, \mu) = 0.5 \frac{1}{1 + \left(\frac{x}{0.7}\right)^2}. \text{ So}$$

nullclines are given by equations $f(x, y, \mu) = 0$, $g(x, y, \mu) = 0$, where in both cases we reorganize the identity to express y as a function of x to draw the curve in xOy plane.

Now let's plot the nullclines and numerically integrated trajectories for 1000 steps (using Euler's scheme) to identify the type of fixed point obtained in the point of intersection of nullclines for three different parameters of μ , where big red dots denote the points where trajectories started.

In [227]:




```

mu=0.5

def f(x,y,mu):
    return mu*8.1*y*(0.111+0.889*x/(0.7+x))-2*x/(0.1+x) + 0.02
def g(x,y,mu):
    return 1/(1+(x/0.7)**2)/2 - y/2

def nullcline1(x,mu):
    return (2*x/(0.1+x)-0.02)/(8.1*mu*(0.111+0.889*x/(0.7+x)))
def nullcline2(x,mu):
    return 1/(1+(x/0.7)**2)

k=0.01
xrange=np.linspace(0,1,10)
yrange=np.linspace(0,1.5,10)
nullclinerange=np.linspace(0,3.5,100)
plt.figure(figsize=(12, 8))

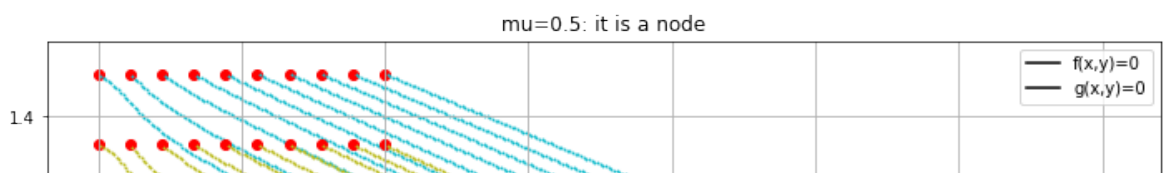
for x0 in xrange:
    for y0 in yrange:
        masx=[x0]
        masy=[y0]
        plt.scatter(x0,y0,color='red')
        #print(x,y)
        for i in range(1000):
            x,y=masx[i],masy[i]
            u,v=f(x,y,mu),g(x,y,mu)
            dx,dy=k*u/np.sqrt(u**2 + v**2),k*v/np.sqrt(u**2 + v**2)
            #print(dx,dy)
            masx.append(x+dx)
            masy.append(y+dy)
            #plt.arrow(x=x,y=y,dx=u/np.sqrt(u**2 + v**2),dy=v/(u**2 + v**2))
        plt.scatter(masx[1:],masy[1:],s=0.8)

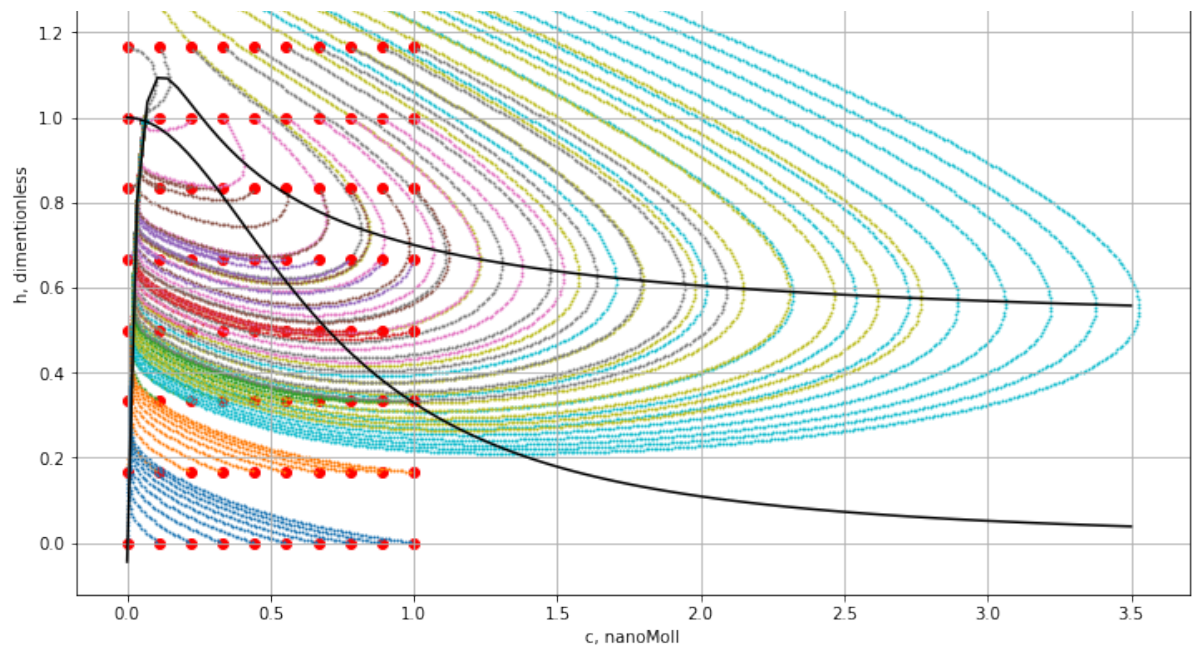
plt.plot([x for x in nullclinerange], [nullcline1(x,mu) for x in nullclinerange])
plt.plot([x for x in nullclinerange], [nullcline2(x,mu) for x in nullclinerange])

# opt_func = lambda x: nullcline1(x,mu)-nullcline2(x,mu)
# fp=optimize.root_scalar(opt_func, bracket=[0, 5]).root
# print("fp=",fp)

plt.grid()
plt.xlabel('c, nanoMol')
plt.ylabel('h, dimensionless')
plt.title('mu='+str(mu)+' : it is a node')
plt.legend()
plt.show()

```





We see that for $\mu = 0.5$, we have a stable node close to $(1, 0)$ on (c, h) plane. All trajectories tend to this point for fixed μ , so there will be the only fixed point for μ from regime 1.

In [226]:

```

mu=0.8

def f(x,y,mu):
    return mu*8.1*y*(0.111+0.889*x/(0.7+x))-2*x/(0.1+x) + 0.02
def g(x,y,mu):
    return 1/(1+(x/0.7)**2)/2 - y/2

def nullcline1(x,mu):
    return (2*x/(0.1+x)-0.02)/(8.1*mu*(0.111+0.889*x/(0.7+x)))
def nullcline2(x,mu):
    return 1/(1+(x/0.7)**2)

k=0.01
xrange=np.linspace(0,1,10)
yrange=np.linspace(0,1,10)
nullclinerange=np.linspace(0,4,100)
plt.figure(figsize=(12, 8))

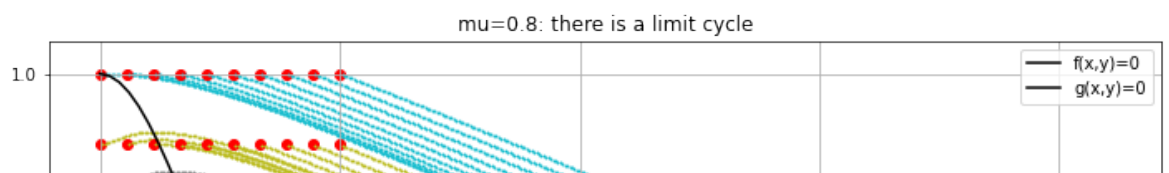
for x0 in xrange:
    for y0 in yrange:
        masx=[x0]
        masy=[y0]
        plt.scatter(x0,y0,color='red')
        #print(x,y)
        for i in range(1000):
            x,y=masx[i],masy[i]
            u,v=f(x,y,mu),g(x,y,mu)
            dx,dy=k*u/np.sqrt(u**2 + v**2),k*v/np.sqrt(u**2 + v**2)
            #print(dx,dy)
            masx.append(x+dx)
            masy.append(y+dy)
            #plt.arrow(x=x,y=y,dx=u/np.sqrt(u**2 + v**2),dy=v/(u**2 + v**2))
        plt.scatter(masx[1:],masy[1:],s=0.8)

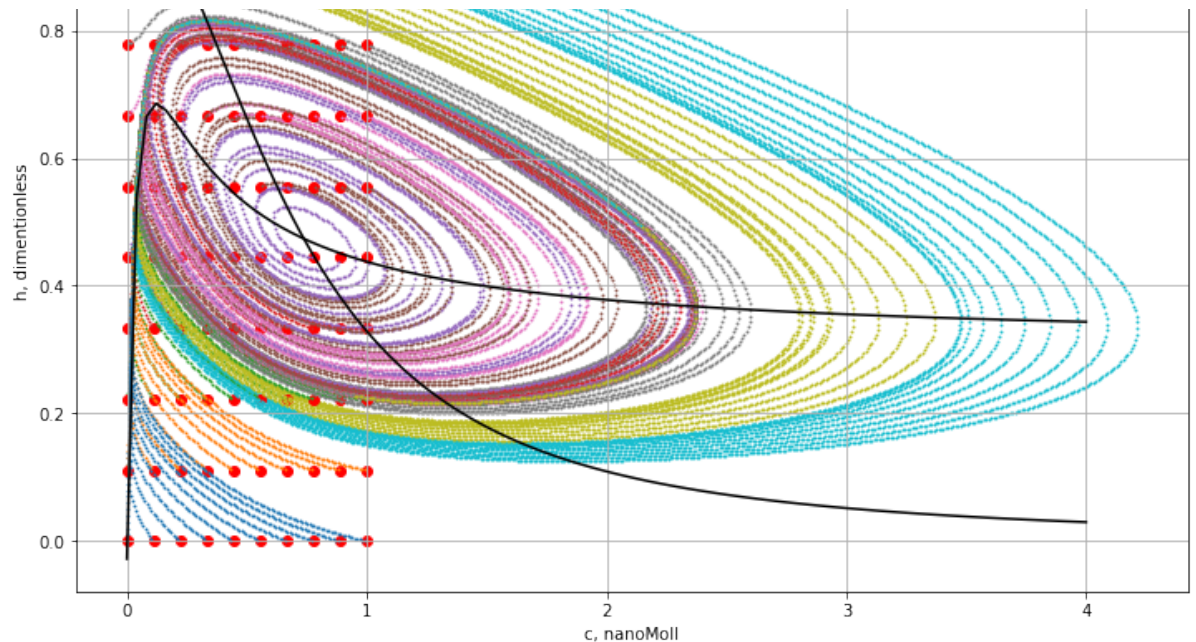
plt.plot([x for x in nullclinerange], [nullcline1(x,mu) for x in nullclinerange])
plt.plot([x for x in nullclinerange], [nullcline2(x,mu) for x in nullclinerange])

# opt_func = lambda x: nullcline1(x,mu)-nullcline2(x,mu)
# fp=optimize.root_scalar(opt_func, bracket=[0, 5]).root
# print("fp=",fp)

plt.grid()
plt.xlabel('c, nanoMol')
plt.ylabel('h, dimensionless')
plt.title('mu='+str(mu)+' : there is a limit cycle')
plt.legend()
plt.show()

```





We see that for $\mu = 0.8$, we have a stable limit cycle close to $(0.8, 0.5)$ on (c, h) plane.

We see that for this regime there will be a limit cycle and hence oscillations, so we will need to compute max and min amplitudes of trajectories.

In [269]:

```

mu=1.4

def f(x,y,mu):
    return mu*8.1*y*(0.111+0.889*x/(0.7+x))-2*x/(0.1+x) + 0.02
def g(x,y,mu):
    return 1/(1+(x/0.7)**2)/2 - y/2

def nullcline1(x,mu):
    return (2*x/(0.1+x)-0.02)/(8.1*mu*(0.111+0.889*x/(0.7+x)))
def nullcline2(x,mu):
    return 1/(1+(x/0.7)**2)

k=0.01
xrange=np.linspace(0,1,10)
yrange=np.linspace(0,0.6,10)
nullclinerange=np.linspace(0,6,100)
plt.figure(figsize=(12, 8))

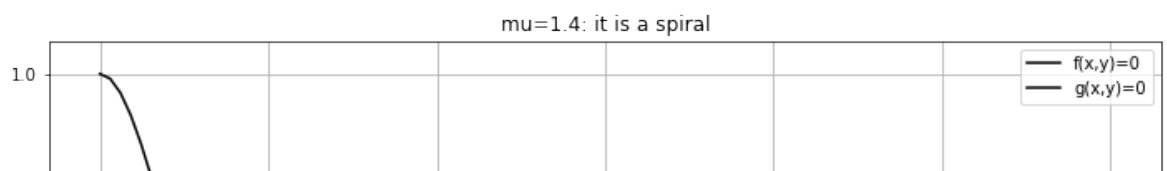
for x0 in xrange:
    for y0 in yrange:
        masx=[x0]
        masy=[y0]
        plt.scatter(x0,y0,color='red')
        #print(x,y)
        for i in range(1000):
            x,y=masx[i],masy[i]
            u,v=f(x,y,mu),g(x,y,mu)
            dx,dy=k*u/np.sqrt(u**2 + v**2),k*v/np.sqrt(u**2 + v**2)
            #print(dx,dy)
            masx.append(x+dx)
            masy.append(y+dy)
            #plt.arrow(x=x,y=y,dx=u/np.sqrt(u**2 + v**2),dy=v/(u**2 + v**2))
        plt.scatter(masx[1:],masy[1:],s=0.8)

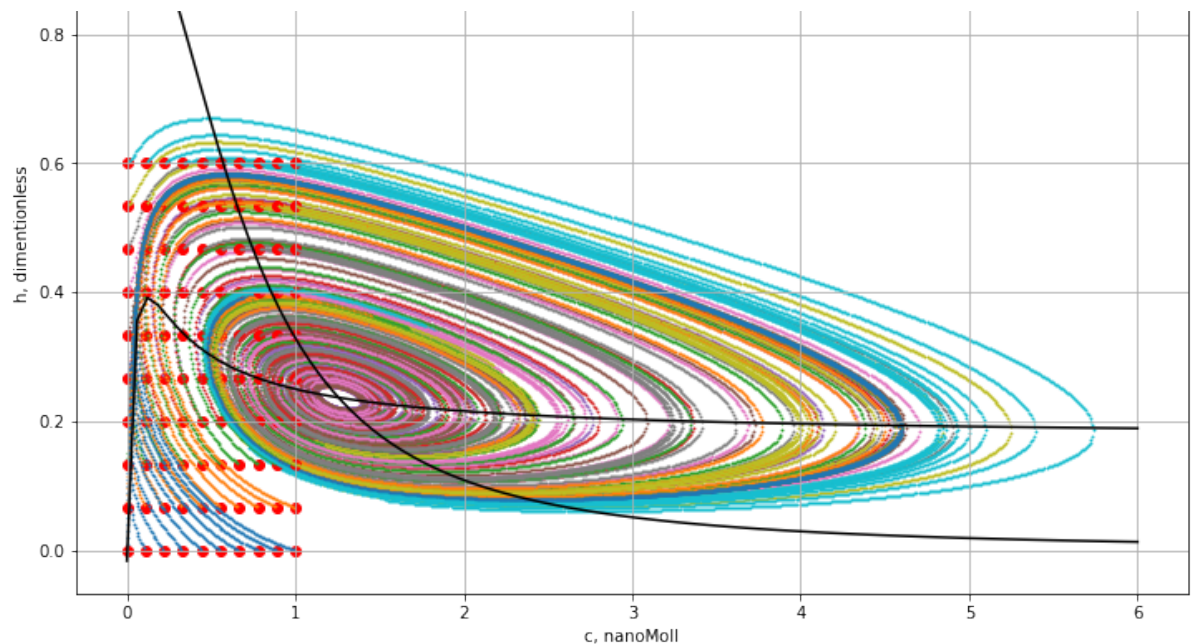
plt.plot([x for x in nullclinerange], [nullcline1(x,mu) for x in nullclinerange])
plt.plot([x for x in nullclinerange], [nullcline2(x,mu) for x in nullclinerange])

# opt_func = lambda x: nullcline1(x,mu)-nullcline2(x,mu)
# fp=optimize.root_scalar(opt_func, bracket=[0, 5]).root
# print("fp=",fp)

plt.grid()
plt.xlabel('c, nanoMol')
plt.ylabel('h, dimentionless')
plt.title('mu='+str(mu)+' : it is a spiral')
plt.legend()
plt.show()

```





We see that for $\mu = 1.4$, we have a stable spiral close to $(1.2, 0.25)$ on (c, h) plane. All trajectories tend to this point for fixed μ , so there will be the only fixed point for μ from regime 1.

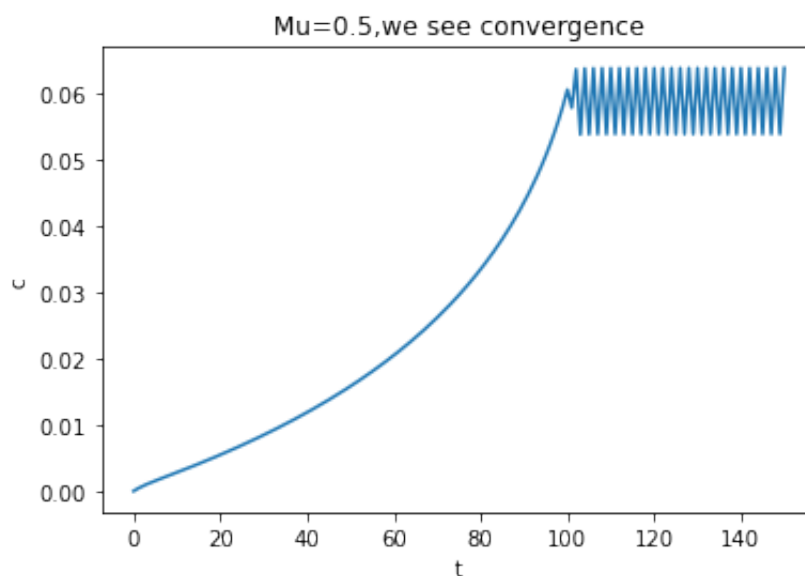
3b) find out in which of three regimes we see significant oscillations


```

In [455]: mu=0.5 # we see convergence
k=0.01
masx=[0.0]
masy=[0.0]
t_range=range(150)
for t in t_range:
    x,y=masx[t],masy[t]
    u,v=f(x,y,mu),g(x,y,mu)
    dx,dy=k*u/np.sqrt(u**2 + v**2),k*v/np.sqrt(u**2 + v**2)
    masx.append(x+dx)
    masy.append(y+dy)
print("c_min=",np.min(masx[-50:-1]),"c_max=",np.max(masx[-50:-1]),"
plt.plot(np.arange(len(masx)),masx)
plt.xlabel('t')
plt.ylabel('c')
plt.title('Mu=0.5,we see convergence')
plt.show()

```

c_min= 0.0538990442318773 c_max= 0.0639576723369066 diff= 0.010058628105029305



```

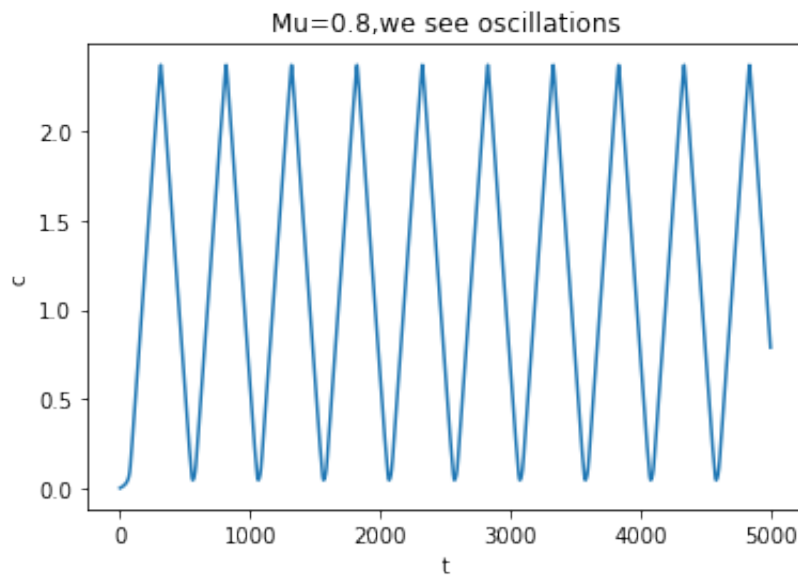
In [456]: mu=0.8 # we see big amplitude oscillations
k=0.01
masx=[0.0]
masy=[0.0]
t_range=range(5000)
for t in t_range:
    x,y=masx[t],masy[t]
    u,v=f(x,y,mu),g(x,y,mu)
    dx,dy=k*u/np.sqrt(u**2 + v**2),k*v/np.sqrt(u**2 + v**2)
    masx.append(x+dx)
    masy.append(y+dy)

print("c_min=",np.min(masx[-50:-1]),"c_max=",np.max(masx[-50:-1]),"

plt.plot(np.arange(len(masx)),masx)
plt.xlabel('t')
plt.ylabel('c')
plt.title('Mu=0.8,we see oscillations')
plt.show()

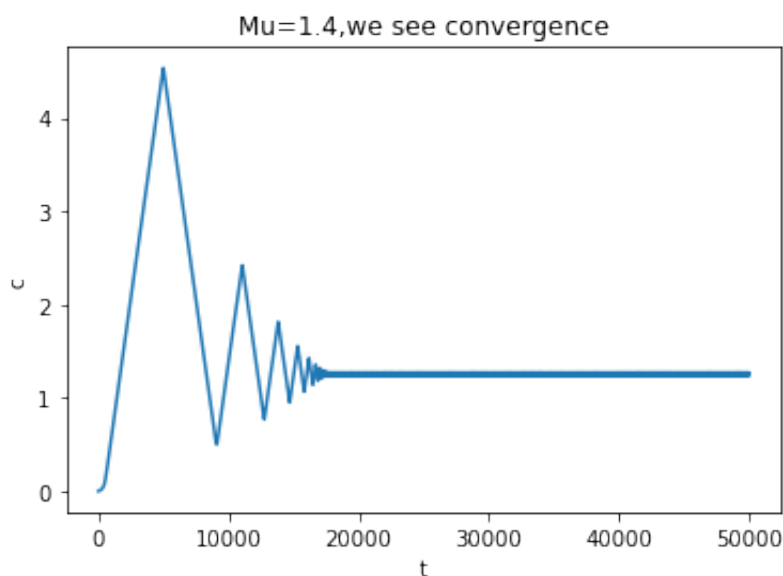
```

c_min= 0.7985847927750621 c_max= 1.2776893828268887 diff= 0.4791045900518266



```
In [457]: mu=1.4 #we see convergence
k=0.001
masx=[0]
masy=[0]
t_range=range(50000)
for t in t_range:
    x,y=masx[t],masy[t]
    u,v=f(x,y,mu),g(x,y,mu)
    dx,dy=k*u/np.sqrt(u**2 + v**2),k*v/np.sqrt(u**2 + v**2)
    masx.append(x+dx)
    masy.append(y+dy)
print("c_min=",np.min(masx[-50:-1]),"c_ma=x",np.max(masx[-2000:-1]))
plt.plot(np.arange(len(masx)),masx)
plt.xlabel('t')
plt.ylabel('c')
plt.title('Mu=1.4,we see convergence')
plt.show()
```

c_min= 1.2438772658077775 c_ma=x 1.2701942653251108 diff= 0.02631699951733335



3c) Plot bifurcation diagram for μ

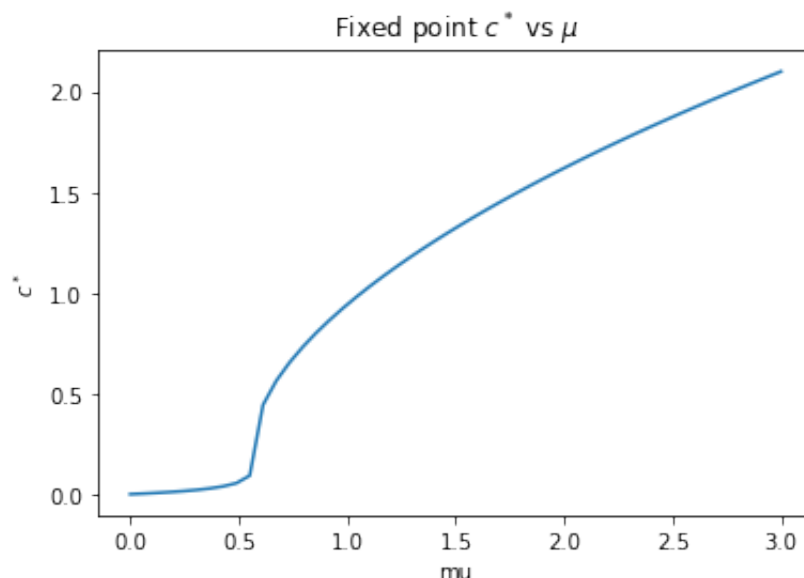
First print the line that is given by the intersection of nullclines.


```
In [458]: #fixed point
mas_mu=[]
mas_fp=[]

for mu in np.linspace(0,3,50):
    opt_func = lambda x: nullcline1(x,mu)-nullcline2(x,mu)
    fp=optimize.root_scalar(opt_func, bracket=[0, 5]).root
    mas_mu.append(mu)
    mas_fp.append(fp)
plt.plot(mas_mu,mas_fp)
plt.xlabel('mu')
plt.ylabel('$c^*$')
plt.title('Fixed point $c^*$ vs $\mu$')
plt.show()
```

<ipython-input-269-a13c8b766be4>:9: RuntimeWarning: divide by zero encountered in double_scalars

```
return (2*x/(0.1+x)-0.02)/(8.1*mu*(0.111+0.889*x/(0.7+x)))
```



Now plot the bifurcation diagram: to do this, we take different μ , and for each μ we run many simulations of trajectories of the system and measure the min and max amplitude of them in a long-time runned (converged) mode. Note that for μ in regime 2 green line (that is given) is unstable limit cycle (we can check this by reversing the time and running simulations -

now all trajectories with μ from regime 2 will converge to green curve).

While blue and orange lines are stable limit cycles.

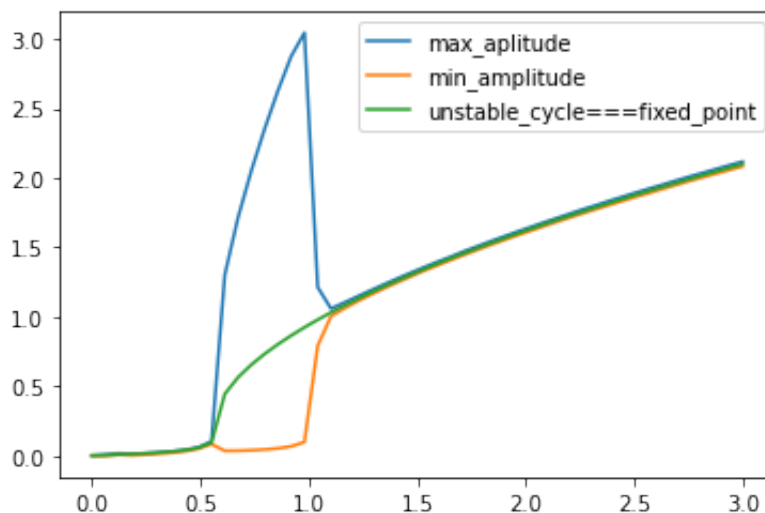
```

In [287]: from tqdm import tqdm
mas_mu=[]
mas_max_amplitude=[]
mas_min_amplitude=[]
for mu in tqdm(np.linspace(0,3,50)):
    mas_mu.append(mu)
    masx=[0.0]
    masy=[0.0]
    k=0.01
    t_range=range(5000)
    if (mu >1):
        k=0.001
        t_range=range(50000)
    for t in t_range:
        x,y=masx[t],masy[t]
        u,v=f(x,y,mu),g(x,y,mu)
        dx,dy=k*u/np.sqrt(u**2 + v**2),k*v/np.sqrt(u**2 + v**2)
        masx.append(x+dx)
        masy.append(y+dy)
    mas_max_amplitude.append(np.max(masx[-2000:-1]))
    mas_min_amplitude.append(np.min(masx[-2000:-1]))

plt.plot(mas_mu,mas_max_amplitude,label='max_aplitude')
plt.plot(mas_mu,mas_min_amplitude,label='min_amplitude')
plt.plot(mas_mu,mas_fp,label='unstable_cycle===fixed_point')
plt.legend()
plt.show()

```

100%|██████████| 50/50 [00:11<00:00, 4.20it/s]



Q4

4a) Fixed points

Fixed points are such x that $f(x) = x$, this means $rx(1 - x^\theta) = 0$ so $x = 0$ or $x = 1$;

To analyse stability use $f'(x) = 1 + r(1 - x^\theta) - r\theta x^\theta$;

$f'(0) = 1 + r > 1$ so $x = 0$ is unsatable;

$f'(1) = 1 - r\theta$ is < 1 in modulo when $\theta \in (0, \frac{2}{r})$, so for these θ $x = 1$ is stable;
Stable 2-cycle occurs for the firs time, when the fixed point loses stability for the first time, and it happens at $\frac{2}{r}$.

4b) Plot bifurcation diagram

The algorithm is the follows: for all theta run N-lengs simulations each starting at a different x_0 , take the steady state of trajectory by throwing away the beginning of each tragectory, and plot the (θ, x_i) pairs on the plane.

In [365]:

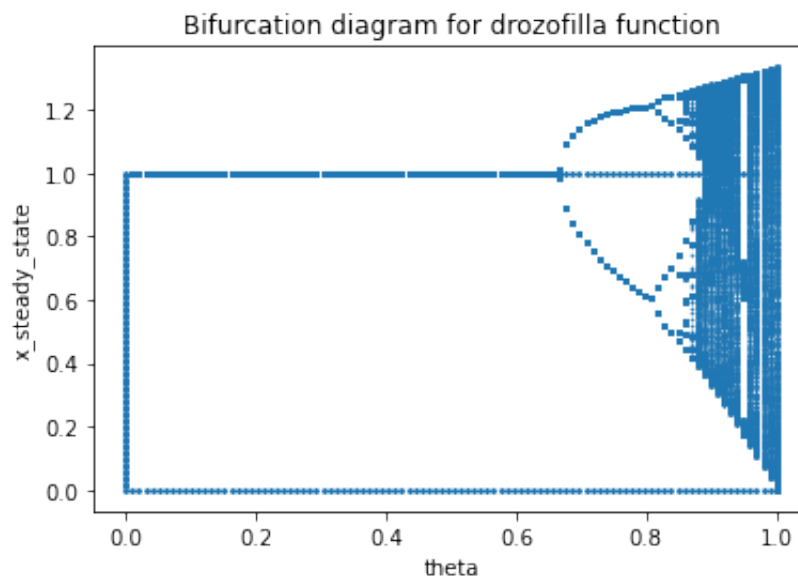
```

r=3

def fdroz(x,r,theta):
    if np.abs(x) < 1e-15:
        return 0.0
    if np.abs(1-x) < 1e-15:
        return 1.0
    return x + r*x*(1-x**theta)
r=3

otvtheta=[]
otvx=[]
for theta in np.linspace(0,3/3,100):
    for x0 in np.linspace(0,1,100):
        masx=[x0]
        N=1000
        for i in range(N):
            x=masx[i]
            masx.append(fdroz(x,3,theta))
        masx=masx[-20:-1]
        for el in masx:
            otvtheta.append(theta)
            otvx.append(el)
plt.scatter(otvtheta,otvx,s=0.5)
plt.xlabel('theta')
plt.ylabel('x_steady_state')
plt.title('Bifurcation diagram for drozofilla function')
plt.show()

```



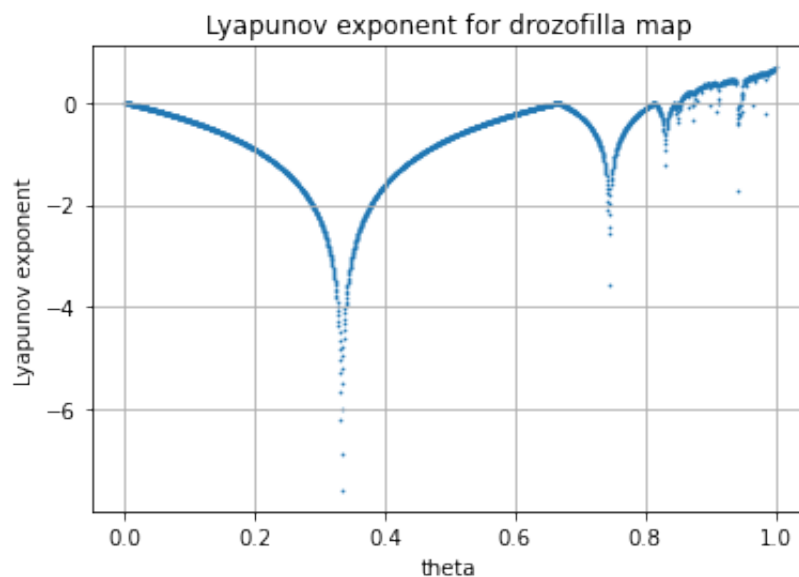
We see that the first point when $x = 1$ loses stability and transforms into two points is exactly at point $\theta = \frac{2}{r} = \frac{2}{3}$ that we found analytically earlier.

4c calculate Lyapunov exponent for different thetas

We just take the formula from the lectures that say we need to take any starting point x_0 , calculate $x_n := f^n(x_0)$, take logarithms of the absolute value of each $\frac{df(x_i)}{dx}$, where $\frac{df}{dx}$ is the derivative of drozofilla function, and take the limit of mean values of these logarithms. This limit is called the Lyapunov exponent. And we just plot these limit values for different θ s.

```
In [411]: def fdroz_prime(x,r,theta):
            return 1 + r*(1-x**theta) - r*theta*x**theta
r=3
mas_lyapunovs=[]
mas_theta=np.linspace(0,3/3,2000)
for theta in tqdm(mas_theta):
    N=1000
    x0=np.pi/5 #take any initial condition
    s=0
    masx=[x0]
    for i in range(N):
        xi=masx[-1]
        s+=np.log(np.abs(fdroz_prime(xi,r,theta)))
        masx.append(fdroz(xi,r,theta))
    mas_lyapunovs.append(s/N)
plt.scatter(mas_theta,mas_lyapunovs,s=0.5)
plt.xlabel('theta')
plt.ylabel('Lyapunov exponent')
plt.grid()
plt.title('Lyapunov exponent for drozofilla map')
plt.show()
```

100%|██████████| 2000/2000 [00:11<00:00, 174.70it/s]



Note that for negative Lyapunov exponents the regime is stable.

We also see that both the bifurcation diagram and Lyapunov exponent graphs look similar to what we had in the lecture with logistic map.

In []:

