Name: Linh Tang

Date: 08/18/2020

Math Bootcamp

FINAL PROJECT- HOPF BIFURCATION

Problem 1:

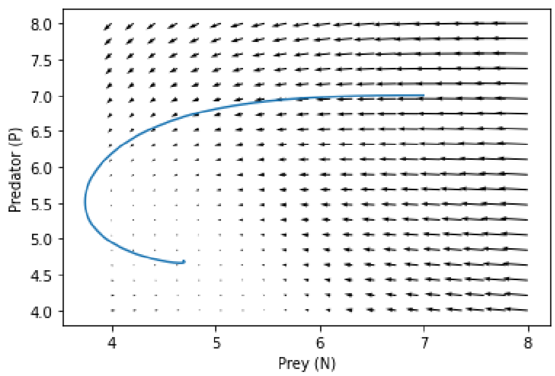
Simulate the vector field and show the trajectory.

From the plot, the equilibrium in the middle is a stable equilibrium point since everything approaches the point [Neq,Peq] = [4.69,4.69].

```
#Problem 1

#Parameters
r1 = 1
r2 = 0.1
k = 7
d = 1
j = 1
w = 0.4

# Make N and P symbolic variables
N = sym.Symbol ('N')
P = sym.Symbol ('P')

dt = 0.1
t = np.arange (0,200,dt) # set up the time step array

#The Holling Tanner prey-predator model
def HT(x,t):
  N = x[0]
  P = x[1]
  Nprime = r1*N*(1-N/k)-w*(N/(d+N))*P
  Pprime = r2*P*(1-j*P/N)
  return [Nprime, Pprime]

#solving ODE (Euler method)
sol = odeint(HT,[7,7],t)
sol
```

```
array([[7.        , 7.        ],
       [6.76711849, 6.99879301],
       [6.55607496, 6.99524329],
       ...,
       [4.69192497, 4.69192497],
       [4.69192497, 4.69192497],
       [4.69192497, 4.69192497]])
```

```
[27] # Plot the vector field and the trajectory
     N,P = np.meshgrid(np.linspace(4,8,20),np.linspace(4,8,20))
     Nprime = r1*N*(1-N/k)-w*(N/(d+N))*P
     Pprime = r2*P*(1-j*P/N)
     plt.quiver(N,P,Nprime, Pprime)

     plt.plot(sol[:,0],sol[:,1]) # get the first column, then second column -> plot it
     plt.xlabel('Prey (N)')
     plt.ylabel('Predator (P)')
     plt.show()
```



Problem 2:

The list of equilibrium points of the system:

```
#Problem 2

#Make symbolic variables
N = sym.Symbol ('N')
P = sym.Symbol ('P')
w = sym.Symbol ('w') # now w is a variable

#Parameter
r1 = 1
r2 = 0.1
k = 7
d = 1
j = 1

sym.init_printing()
Nprime = sym.Eq(r1*N*(1-N/k)-w*(N/(d+N))*P,0)
Pprime = sym.Eq (r2*P*(1-j*P/N),0)
sol = sym.solve([Nprime, Pprime],(N,P)) #  a list of several equilibria
sol
```

$$\left[(7.0,\quad 0.0),\quad \left(\frac{49.0w^2-7.0w\sqrt{49.0w^2-84.0w+64.0}-84.0w+6.0\sqrt{49.0w^2-84.0w+64.0}+50.0}{-7.0w+(49.0w^2-84.0w+64.0)^{0.5}+6.0},\quad -3.5w+0.5\sqrt{49.0w^2-84.0w+64.0}+3.0\right),\right.$$

$$\left.\left(\frac{1}{7.0w+(49.0w^2-84.0w+64.0)^{0.5}-6.0}\left(-49.0w^2-7.0w\sqrt{49.0w^2-84.0w+64.0}+84.0w+6.0\sqrt{49.0w^2-84.0w+64.0}-50.0\right),\quad -3.5w-0.5\sqrt{49.0w^2-84.0w+64.0}+3.0\right)\right]$$

```
[29] print ('Check second element of the list by plug in w = 0.4 and compare the result with what we got in problem 1')

     print ('Neq =',sol[1][0].subs(w,0.4))
     print ('Peq =',sol[1][1].subs(w,0.4))
```

```
Check second element of the list by plug in w = 0.4 and compare the result with what we got in problem 1
Neq = 4.69192496674806
Peq = 4.69192496674806
```

In the list of equilibria, the second element is biologically meaningful. By plug in w = 0.4, we got the same equilibrium with Problem 1.

Problem 3: Storing biologically meaningful equilibrium point to a variable.

```
[30] #Problem 3

     #Extract the second element(biological meaningful one) and store it
     Neq = sol[1][0]
     Peq = sol [1][1]
```

Problem 4:

Set up the Jacobian of the system and store it as variable J.

```
#Problem 4

#Make symbolic variables
N = sym.Symbol ('N')
P = sym.Symbol ('P')
w = sym.Symbol ('w')

#Parameter
r1 = 1
r2 = 0.1
k = 7
d = 1
j = 1

#Set up the Jacobian matrix based on two variables N and P
Nprime = r1*N*(1-N/k)-w*(N/(d+N))*P
Pprime = r2*P*(1-j*P/N)
sym.init_printing()
function = Matrix([Nprime, Pprime])
variable = Matrix ([N,P])
J = function.jacobian(variable)
J
```

$$\begin{bmatrix} \frac{NPw}{(N+1)^2} - \frac{2N}{7} - \frac{Pw}{N+1} + 1 & -\frac{Nw}{N+1} \\ \frac{0.1P^2}{N^2} & 0.1 - \frac{0.2P}{N} \end{bmatrix}$$

Problem 5:

```
[32] #Problem 5

     #subtitute Neq and Peq found in Problem 3
     Jeq = J.subs([(N,Neq),(P,Peq)])
     Jeq
```

$$\left[\begin{matrix}- \frac{w \left(- 3.5 w + 0.5 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 3.0\right)}{1 + \frac{49.0 w^{2} - 7.0 w \sqrt{49.0 w^{2} - 84.0 w + 64.0} - 84.0 w + 6.0 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 50.0}{- 7.0 w + \left(49.0 w^{2} - 84.0 w + 64.0\right)^{0.5} + 6.0}} + \frac{w \left(- 3.5 w + 0.5 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 3.0\right) \left(49.0 w^{2} - 7.0 w \sqrt{49.0 w^{2} - 84.0 w + 64.0} - 84.0 w + 6.0 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 50.0\right)}{\left(1 + \frac{49.0 w^{2} - 7.0 w \sqrt{49.0 w^{2} - 84.0 w + 64.0} - 84.0 w + 6.0 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 50.0}{- 7.0 w + \left(49.0 w^{2} - 84.0 w + 64.0\right)^{0.5} + 6.0}\right)^{2} \left(- 7.0 w + \left(49.0 w^{2} - 84.0 w + 64.0\right)^{0.5} + 6.0\right)} + 1 - \frac{2 \left(49.0 w^{2} - 7.0 w \sqrt{49.0 w^{2} - 84.0 w + 64.0} - 84.0 w + 6.0 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 50.0\right)}{7 \left(- 7.0 w + \left(49.0 w^{2} - 84.0 w + 64.0\right)^{0.5} + 6.0\right)} & - \frac{w \left(49.0 w^{2} - 7.0 w \sqrt{49.0 w^{2} - 84.0 w + 64.0} - 84.0 w + 6.0 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 50.0\right)}{\left(1 + \frac{49.0 w^{2} - 7.0 w \sqrt{49.0 w^{2} - 84.0 w + 64.0} - 84.0 w + 6.0 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 50.0}{- 7.0 w + \left(49.0 w^{2} - 84.0 w + 64.0\right)^{0.5} + 6.0}\right) \left(- 7.0 w + \left(49.0 w^{2} - 84.0 w + 64.0\right)^{0.5} + 6.0\right)}\\\frac{0.1 \left(- 7.0 w + \left(49.0 w^{2} - 84.0 w + 64.0\right)^{0.5} + 6.0\right)^{2} \left(- 3.5 w + 0.5 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 3.0\right)^{2}}{\left(49.0 w^{2} - 7.0 w \sqrt{49.0 w^{2} - 84.0 w + 64.0} - 84.0 w + 6.0 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 50.0\right)^{2}} & - \frac{0.2 \left(- 7.0 w + \left(49.0 w^{2} - 84.0 w + 64.0\right)^{0.5} + 6.0\right) \left(- 3.5 w + 0.5 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 3.0\right)}{49.0 w^{2} - 7.0 w \sqrt{49.0 w^{2} - 84.0 w + 64.0} - 84.0 w + 6.0 \sqrt{49.0 w^{2} - 84.0 w + 64.0} + 50.0} + 0.1\end{matrix}\right]$$

Problem 6

Find the eigenvalue of the Jacobian that we found in problem 5, and then get the real part of the eigenvalue

```
[33] #Problem 6
     #Find the eigen value
     Jeq.eigenvals() #This is stored in dictionary data type
```

$$\Big\{ \frac{-16470860w^8+2352980w^7\sqrt{49w^2-84w+64}+109060623w^7-13563249w^6\sqrt{49w^2-84w+64}-333585336w^6+35357126w^5\sqrt{49w^2-84w+64}+609916426w^5-53525836w^4\sqrt{49w^2-84w+64}-725187064w^4+50540560w^3\sqrt{49w^2-84w+64}+572023991w^3-29645931w^2\sqrt{49w^2-84w}}{980\left(-16807w^7+2401w^6\sqrt{49w^2-84w+64}+105644w^6-13034w^5\sqrt{49w^2-84w+64}-301154w^5+31164w^4\sqrt{49w^2-84w+64}+500976w^4-41720w^3\sqrt{49w^2-84w+64}-523159w^3+32859w^2\sqrt{49w^2-84w+64}+342180w^2-14406w\sqrt{49w^2}\right.}$$

```
[34] #Extrac eigenvalues
     Jeq.eigenvals().keys()
```

```
dict_keys([(-16470860*w**8 + 2352980*w**7*sqrt(49*w**2 - 84*w + 64) + 109060623*w**7 - 13563249*w**6*sqrt(49*w**2 - 84*w + 64) - 333585336*w**6 + 35357126*w**5*sqrt(49*w**2 - 84*w + 6
```

```
[35] #Converting the data type: dict -> list
     Jeq_eval = list(Jeq.eigenvals().keys())
     Jeq_eval
```

$$\Big[\frac{-16470860w^8+2352980w^7\sqrt{49w^2-84w+64}+109060623w^7-13563249w^6\sqrt{49w^2-84w+64}-333585336w^6+35357126w^5\sqrt{49w^2-84w+64}+609916426w^5-53525836w^4\sqrt{49w^2-84w+64}-725187064w^4+50540560w^3\sqrt{49w^2-84w+64}+572023991w^3-29645931w^2\sqrt{49w^2-84w+}}{980\left(-16807w^7+2401w^6\sqrt{49w^2-84w+64}+105644w^6-13034w^5\sqrt{49w^2-84w+64}-301154w^5+31164w^4\sqrt{49w^2-84w+64}+500976w^4-41720w^3\sqrt{49w^2-84w+64}-523159w^3+32859w^2\sqrt{49w^2-84w+64}+342180w^2-14406w\sqrt{49w^2}\right.}$$

```
[36] #Get the real part

     eval = Jeq_eval[0]
     eval
```

$$\frac{-16470860w^8+2352980w^7\sqrt{49w^2-84w+64}+109060623w^7-13563249w^6\sqrt{49w^2-84w+64}-333585336w^6+35357126w^5\sqrt{49w^2-84w+64}+609916426w^5-53525836w^4\sqrt{49w^2-84w+64}-725187064w^4+50540560w^3\sqrt{49w^2-84w+64}+572023991w^3-29645931w^2\sqrt{49w^2-84w+6}}{980\left(-16807w^7+2401w^6\sqrt{49w^2-84w+64}+105644w^6-13034w^5\sqrt{49w^2-84w+64}-301154w^5+31164w^4\sqrt{49w^2-84w+64}+500976w^4-41720w^3\sqrt{49w^2-84w+64}-523159w^3+32859w^2\sqrt{49w^2-84w+64}+342180w^2-14406w\sqrt{49w^2-}\right.}$$

$$\frac{\sqrt{2}}{980\left(-16807w^7+2401w^6\sqrt{49w^2-84w+64}+105644w^6-13034w^5\sqrt{49w^2-84w+64}-301154w^5+31164w^4\sqrt{49w^2-84w+64}+500976w^4-41720w^3\sqrt{49w^2-84w+64}-523159w^3+32859w^2\sqrt{49w^2-84w+64}+342180w^2-14406w\sqrt{49w^2-84w+64}-129654w+2744\sqrt{49w^2-84w+64}+2\right.}$$

Problem 7

The values of eigen value at w = 0.4, w = 0.7, w = 1.0.

```
[37] #Problem 7

     #find eigenvalue at w = 0.4
     eval.subs(w,0.4)
```

$$-0.249239274177901 - 0.0731441710538034\sqrt{2}i$$

```
[38] #find eigenvalue at w = 0.7
     eval.subs(w,0.7)
```

$$-0.0775683098551748 - 0.162842244186297\sqrt{2}i$$

```
[39] #find eigenvalue at w = 1.0
     eval.subs(w,1.0)
```

$$0.0292161951255947 - 0.161208514813185\sqrt{2}i$$

Problem 8:

Find ½ the trace and evaluate it with w = 0.4, w = 0.7, w = 1.0. We got the same real part with problem 7, but in a shorter way

```
[64] #Problem 8
     # An algebraic shortcut to fin the real part of eigenvalue

     real = 1/2 *(np.trace(Jeq)) # 1/2 (a+d) = real part of eigen value
     real
```

$$-\frac{0.5w\left(-3.5w+0.5\sqrt{49.0w^2-84.0w+64.0}+3.0\right)}{1+\frac{49.0w^2-7.0w\sqrt{49.0w^2-84.0w+64.0}-84.0w+6.0\sqrt{49.0w^2-84.0w+64.0}+50.0}{-7.0w+\left(49.0w^2-84.0w+64.0\right)^{0.5}+6.0}} + \frac{0.5w\left(-3.5w+0.5\sqrt{49.0w^2-84.0w+64.0}+3.0\right)\left(49.0w^2-7.0w\sqrt{49.0w^2-84.0w+64.0}-84.0w+6.0\sqrt{49.0w^2-84.0w+64.0}+50.0\right)}{\left(1+\frac{49.0w^2-7.0w\sqrt{49.0w^2-84.0w+64.0}-84.0w+6.0\sqrt{49.0w^2-84.0w+64.0}+50.0}{-7.0w+\left(49.0w^2-84.0w+64.0\right)^{0.5}+6.0}\right)^2\left(-7.0w+(49.0w^2-84.0w+64.0)^{0.5}+6.0\right)} - \frac{0.1\left(-7.0w+\left(49.0w^2-84.0w+64.0\right)^{0.5}+6.0\right)\left(-3.5w+0.5\sqrt{49.0w^2-84.0w+64.0}+3.0\right)}{49.0w^2-7.0w\sqrt{49.0w^2-84.0w+64.0}-84.0w+6.0\sqrt{49.0w^2-84.0w+64.0}+50.0} + 0.55 - \frac{0.142857142857143}{-7.0w+(49.0w^2-84.0w+64.0)^{0.5}+6.0}\left(49.0w^2-7.0w\sqrt{49.0w^2-84.0w+64.0}-84.0w+6.0\sqrt{49.0w^2-84.0w+64.0}+50.0\right)$$

```
[65] real.subs(w,0.4) #real part of eigenvalue at w=0.4
```

$$-0.249239274177899$$

```
[66] real.subs(w,0.7) #real part of eigenvalue at w=0.7
```

$$-0.0775683098551474$$

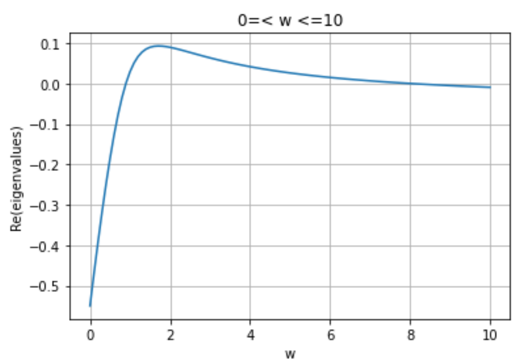
```
[67] real.subs(w,1.0) #real part of eigenvalue at w=1.0
```

$$0.0292161951175557$$

Problem 9:

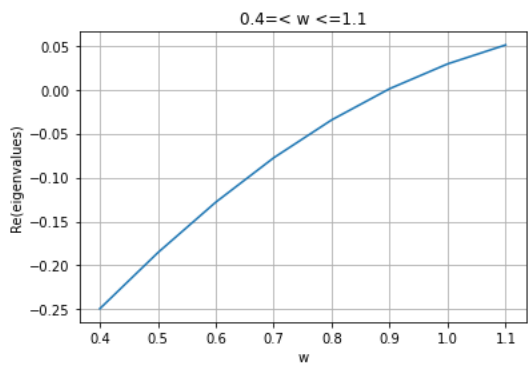
Plot for w in between 0 and 10

```python
#Problem 9
w = sym.Symbol ('w')

def frange(start, stop, step):
    #range() like function which accept float type
    i = start
    while i < stop:
        yield i
        i += step

w_arr = list(frange(0, 10, 0.1)) #create a list of w values
real_v =[]
#Calculate the real eigenvalue with corresponding w value
for i in range (len(w_arr)):
  new = real.subs(w,w_arr[i])
  real_v.append(new)

# Plot for w between 0 anad 10
plt.plot(w_arr,real_v)
plt.xlabel('w')
plt.ylabel('Re(eigenvalues)')
plt.title ('0=< w <=10')
plt.grid()
plt.show()
```



- Plot for w from 0.4-1.1

```
[69] w_arr = list(frange(0.4, 1.1, 0.1)) #create a list of w values 0.4 ->1.1

     real_v =[]
     for i in range (len(w_arr)):
       new = real.subs(w,w_arr[i])
       real_v.append(new)
     real_v

     # Plot for w between 0.4 anad 1.1
     plt.plot(w_arr,real_v)
     plt.xlabel('w')
     plt.ylabel('Re(eigenvalues)')
     plt.title ('0.4=< w <=1.1')
     plt.grid()
     plt.show()
```



Problem 10:

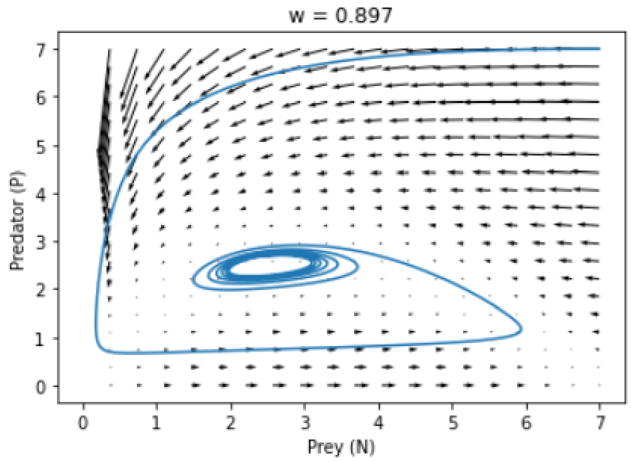
By judging the graph with w in between 0.4 and 1.1, the real part of eigen value is 0 with the value of w about 0.89. For the graph with w in between 0 -10, we got another point that made the real part of eigenvalue 0, w is about 8.

```
[70] # Problem 10

     #Find w such that the real part of eigenvalue is 0
     w = sym.Symbol ('w')
     eq = sym.Eq(real,0)
     points = sym.solve(eq,w)
     points
```

$[0.896799718157616, \quad 8.01034313898524]$

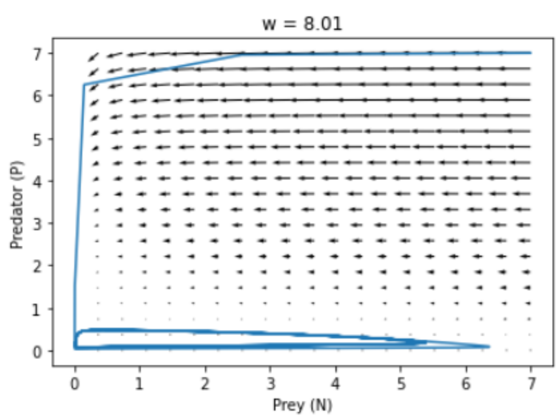
- Plot with w = 0.897: the Hopf bifurcation occurs.

```
[71] #Plot Vector field for w = 0.897

     #Parameters
     r1 = 1
     r2 = 0.1
     k = 7
     d = 1
     j = 1
     w = round(points[0],3) #w = 0.897

     # Make N and P symbolic variables
     N = sym.Symbol ('N')
     P = sym.Symbol ('P')

     dt = 0.1
     t = np.arange (0,200,dt)

     def HT(x,t):
       N = x[0]
       P = x[1]
       Nprime = r1*N*(1-N/k)-w*(N/(d+N))*P
       Pprime = r2*P*(1-j*P/N)
       return [Nprime, Pprime]

     #solving ODE

     sol = odeint(HT,[7,7],t)
     N,P = np.meshgrid(np.linspace(0,7,20),np.linspace(0,7,20))

     Nprime = r1*N*(1-N/k)-w*(N/(d+N))*P
     Pprime = r2*P*(1-j*P/N)
     plt.quiver(N,P,Nprime, Pprime)
     plt.plot(sol[:,0],sol[:,1])
     plt.xlabel('Prey (N)')
     plt.ylabel('Predator (P)')
     plt.title('w = 0.897')
     plt.show()
```



Extra credit:

The equilibrium is a saddle point at [Neq,Peq] = [0.139,0.139] with w = 8.01. This is not a healthy cycle. When the trajectory tries to reach the equilibrium point, we have the asymptotic behavior; that means some variables approach the infinity that we do not want it happens in the nature.



```
#Extra credit

#Make symbolic variables
N = sym.Symbol ('N')
P = sym.Symbol ('P')


#Parameter
r1 = 1
r2 = 0.1
k = 7
d = 1
j = 1
w = 8.01

#Set up the Jacobian matrix based on two variables N and P
Nprime = r1*N*(1-N/k)-w*(N/(d+N))*P
Pprime = r2*P*(1-j*P/N)
sym.init_printing()
function = Matrix([Nprime, Pprime])
variable = Matrix ([N,P])
J_10 = function.jacobian(variable)
J_10
```

$$\begin{bmatrix} \frac{8.01NP}{(N+1)^2} - \frac{2N}{7} - \frac{8.01P}{N+1} + 1 & -\frac{8.01N}{N+1} \\ \frac{0.1P^2}{N^2} & 0.1 - \frac{0.2P}{N} \end{bmatrix}$$

```
Neq.subs(w,8.01)
Peq.subs(w,8.01)
```

$$0.139416080616446$$

```
[26] N_10 = 0.1394 # value of Neq at w = 8.01
     P_10 = 0.1394 # value of Peq at w = 8.01
     J10_eq = J_10.subs([(N,N_10),(P,P_10)])
     J10_eq
```

$$\begin{bmatrix} 0.100083485608737 & -0.97998420221169 \\ 0.1 & -0.1 \end{bmatrix}$$

```
[27] J10_eq.eigenvals()
```

$$\left\{ \frac{83485608737}{2000000000000000} - \frac{\sqrt{35196027967133433381254926483 1}\,i}{2000000000000000} : 1, \quad \frac{83485608737}{2000000000000000} + \frac{\sqrt{351960279671334333812549264831}\,i}{2000000000000000} : 1 \right\}$$

```
[28] J10_eq.eigenvals().keys()
```

```
dict_keys([83485608737/2000000000000000 - sqrt(351960279671334333812549264831)*I/2000000000000000, 83485608737/2000000000000000
```

```
J_10_eval = list(J10_eq.eigenvals().keys()) #convert to the eigenvalue list
print ('Eigen values: \n')
J_10_eval
```

```
Eigen values:
```

$$\left[\frac{83485608737}{2000000000000000} - \frac{\sqrt{351960279671334333812549264831}\,i}{2000000000000000}, \quad \frac{83485608737}{2000000000000000} + \frac{\sqrt{351960279671334333812549264831}\,i}{2000000000000000} \right]$$