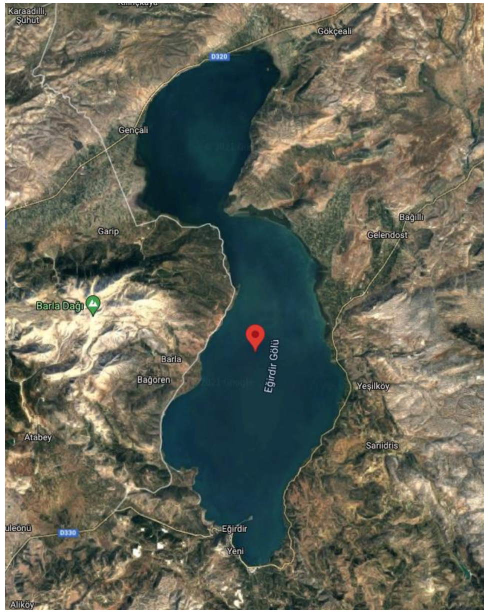
Group: Amalia Bogri, Christian Berrig & Jonas Bolduan

Tragedy of the commons

General model parameters

Parameter	Calculation	Unit
Resource density, N		$[kg/km^2]$
Carrying capacity, K		$[kg/km^2]$
Longevity, λ		$[\text{years}]$
Death rate, d	$\text{d} = 1/\lambda$	$[1/\text{year}]$
Offspring, f		$[1/\text{years}]$
Birth rate, br	$\text{br} = f/\lambda$	$[1/\text{year}]$
Growth rate, r	$\text{r} = br - d$	$[1/\text{year}]$
Effort, E		$[1/\text{year}]$
Agent density, A		$[boats/km^2]$
Clearance rate, b		$[km^2/boat/year]$
Profit, P	$\text{P} = pbN - C$	$[\$/year/boat]$
Cost of exploitation, C		$[\$/year/boat]$
Effort increase rate, w		$[boats/km^2/year]$
Price, p		$[\$/kg]$

Crayfish
(Astacus leptodactylus)



Lake Eğirdir
(180km north
of Antalya, Turkey)
size (area): 482 km²

Exploitation with constant effort model

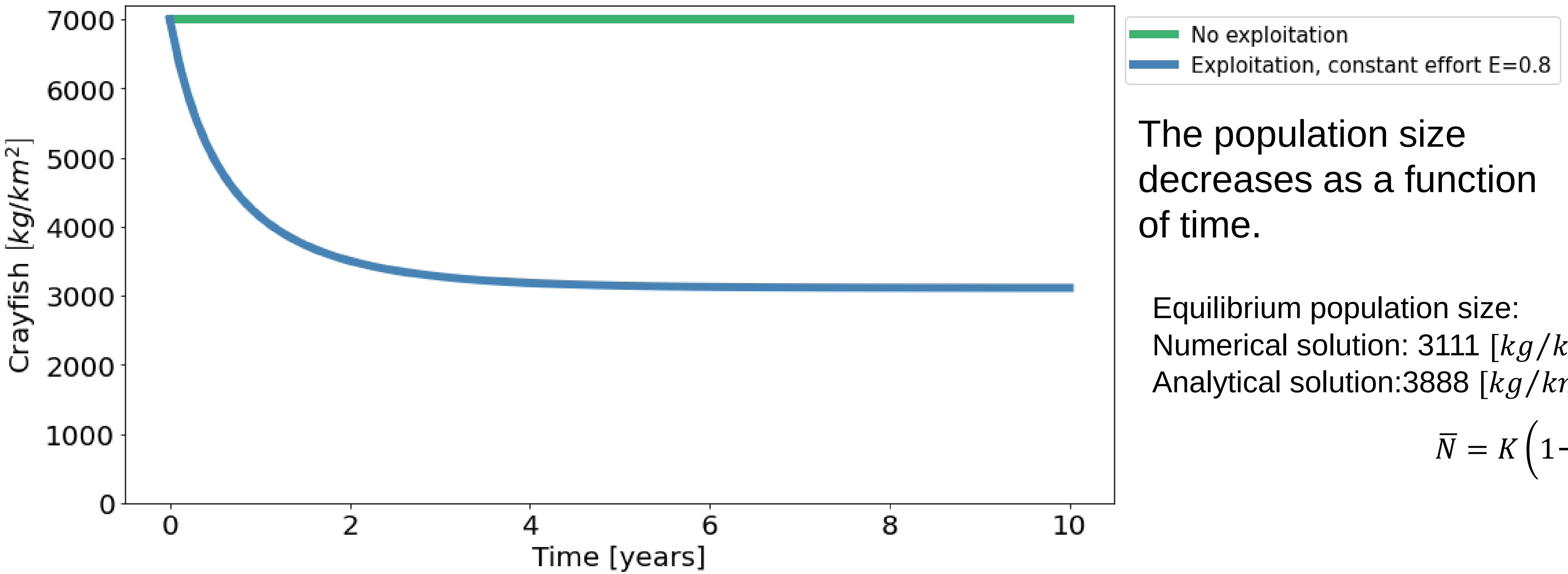
Crayfish: $\boldsymbol{N}$

$$\frac{dN}{dt} = r\left(1 - \frac{N}{K}\right)N - EN$$

Parameter	Calculation	Unit	Value
Resource density, N		$[kg/km^2]$	
Carrying capacity, K		$[kg/km^2]$	7000
Longevity, λ		[years]	5
Death rate, d	$\mathrm{d} = 1/\lambda$	[1/year]	1/5
Offspring, f		[1/years]	10
Birth rate, br	$\mathrm{br} = f/\lambda$	[1/year]	10/5
Growth rate, r	$\mathrm{r} = br - d$	[1/year]	1.8
Effort, E		[1/year]	0.8

Exploitation with constant effort model

$$\frac{dN}{dt} = r\left(1-\frac{N}{K}\right)N - EN$$



The population size decreases as a function of time.

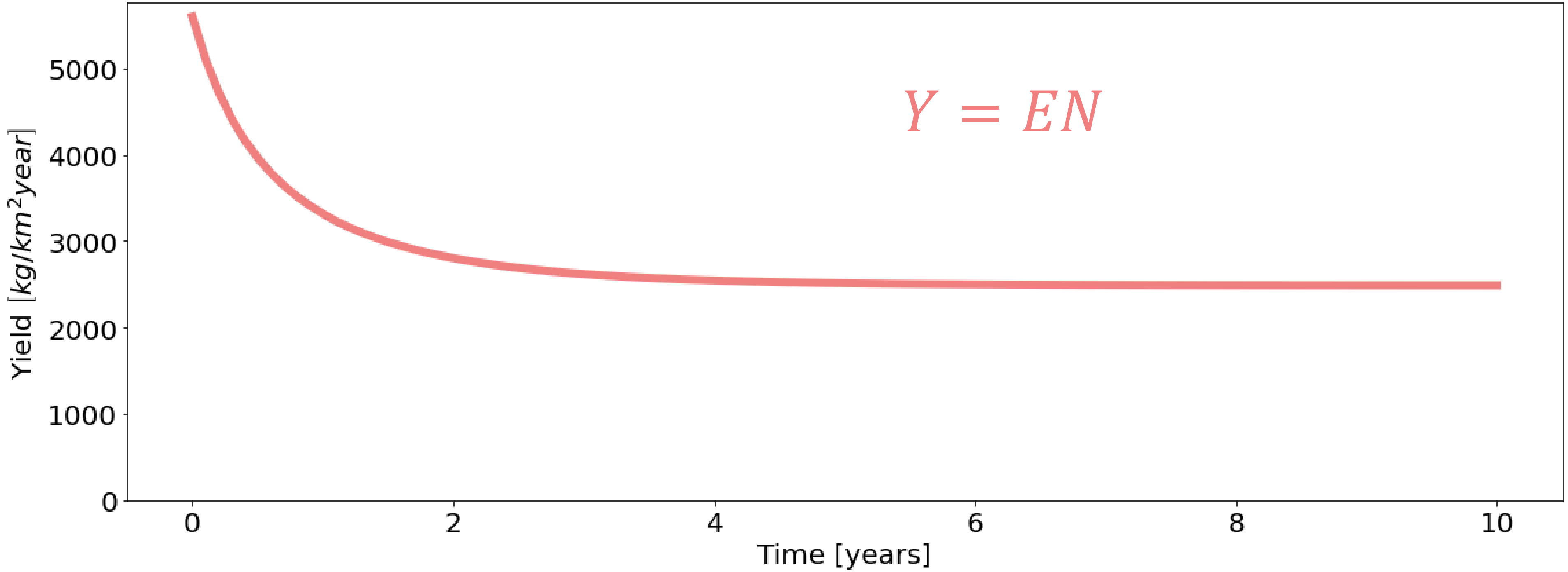
Equilibrium population size:
Numerical solution: 3111 $[kg/km^2]$
Analytical solution:3888 $[kg/km^2]$

$$\bar{N} = K\left(1-\frac{E}{r}\right)$$

Exploitation with constant effort model

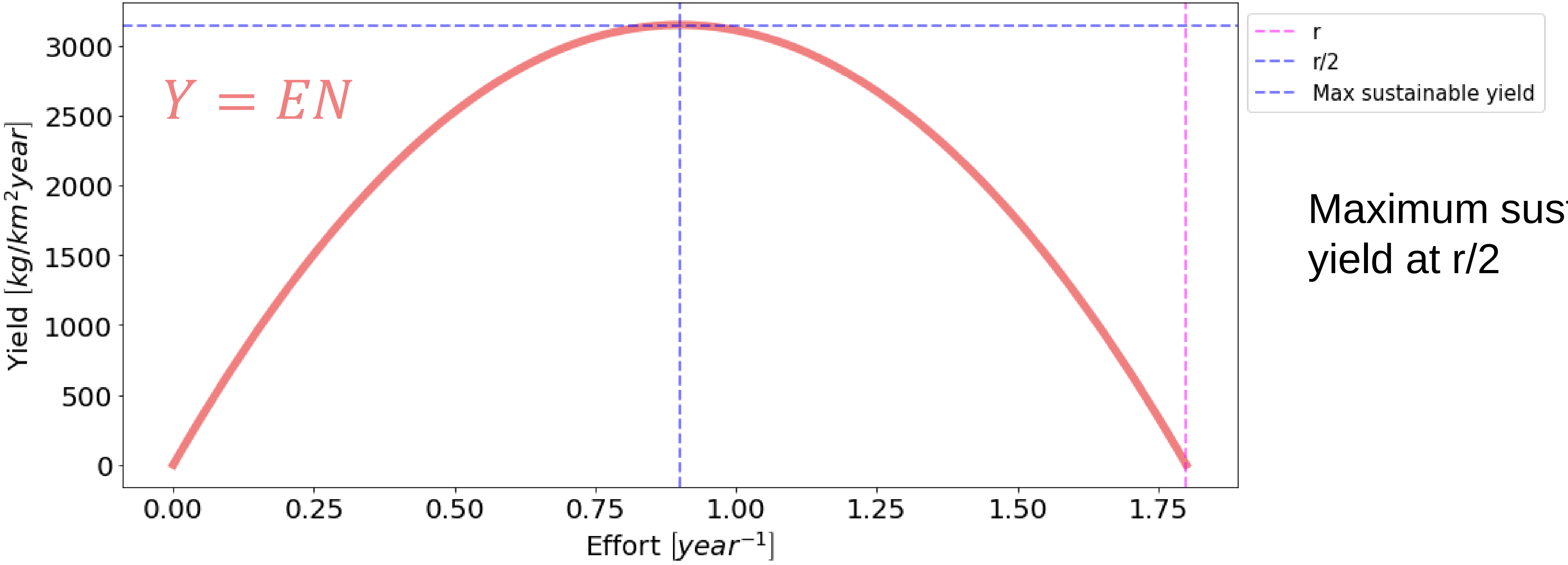
$$\frac{dN}{dt} = r\left(1 - \frac{N}{K}\right)N - EN$$

The yield decreases as a function of time.



Exploitation with constant effort model

$$\frac{dN}{dt} = r\left(1-\frac{N}{K}\right)N - EN$$



Maximum sustainable yield at r/2

Exploitation with constant effort model, python code

```
# # -------- Logistic equation with exploitation factor---------
def deriv_e(N, t, r, K, E):
    return r*N*(1-(N/K))-E*N

# Question 2: Consider N0 = K, and constant effort E. How does population size vary as function of time?

# Parameters:
K = 7000 # Carrying capacity # Unit: [kg crayfish/ km^2]
b = 10/5 # Birth rate # Unit: [1 / year]
d = 1/5 # Death rate # Unit: [1 / year]
r = b-d # Growth rate # Unit: [1 / year] # r = 1.8
N0 = K # Initial population equal to Carrying capacity  # Unit: [kg crayfish/ km^2]
t = np.linspace(0, 10, 100) # in years
E = 0.8 # Effort # Exploitation  # Unit: [1/year]
par = (r, K, E)

E = 0
par = (r, K, E)
ns_e0 = odeint(deriv_e, N0, t, args=par) # numerical solution for exploitation

E = 1
par = (r, K, E)
ns_e1 = odeint(deriv_e, N0, t, args=par) # numerical solution for exploitation

fig, ax = plt.subplots(1,1, figsize=(16, 6), tight_layout=True)
ax.plot(t, ns_e0, label = 'No exploitation', color='mediumseagreen')
ax.plot(t, ns_e1,  label = 'Exploitation, constant effort E=0.8', color='steelblue')
ax.set_ylim(bottom=0)
ax.set_ylabel('Crayfish $\\left[kg/km^{2}\\right]$')
ax.set_xlabel('Time [years]')
ax.legend(bbox_to_anchor=(1.0, 1), loc=2, borderaxespad=0.5, fontsize=15)
plt.show()
print(f'Steady state at {min(ns_e1)} kg crayfish/sqkm')

# Question 3: How does yield vary with time?
E=0.8
fig, ax = plt.subplots(1,1, figsize=(16, 6), tight_layout=True)
ax.plot(t, ns_e1*E, color = 'lightcoral')
ax.set_ylim(bottom=0)
ax.set_ylabel('Yield $\\left[kg/km^{2} year\\right]$')
ax.set_xlabel('Time [years]')
plt.show()
print(f'Steady state at yield {min(ns_e1*E)}')
```

```
Es = np.linspace(0, r, 100)
Y = Es*(-K*(Es - r)/r)
fig, ax = plt.subplots(1,1, figsize=(16, 6), tight_layout=True)
ax.plot(Es, Es*(-K*(Es - r)/r), color = 'lightcoral')
ax.axvline(r, alpha=0.5, linestyle="--", color="magenta", linewidth=2, label='r')
ax.axvline(r/2, alpha=0.5, linestyle="--", color="blue", linewidth=2, label='r/2')
ax.axhline(max(Es*(-K*(Es - r)/r)), alpha=0.5, linestyle="--", color="blue", linewidth=2,
label='Max sustainable yield')

ax.set_ylabel('Yield $\\left[kg/km^{2} year\\right]$')
ax.set_xlabel('Effort $\\left[year^{-1}\\right]$')
ax.legend(bbox_to_anchor=(1.0, 1), loc=2, borderaxespad=0.5, fontsize=15)
plt.show()
print(f'Max sustainable yield at {max(Es*(-K*(Es - r)/r))}')


# Question 4. Does equilibrium population size and yeld converge to the exact analytical
equilibrium?

# Analytical solutions for steady states:
# Solve for the equilibrium abundances of prey and predator
Ns = Symbol('Ns')
Nstar = solve([r*Ns*(1-(Ns/K))-E*Ns])
print(Nstar)


# Let's try with the full analytical solution, to be sure:
Ns = Symbol('Ns')
rs = Symbol('rs')
Ks = Symbol('Ks')
Es = Symbol('Es')
Nstar_symb = solve([(rs*(1-(Ns/Ks))-Es)*Ns], Ns)
print(Nstar_symb)
print(-K*(E - r)/r)
```

Exploitation by agents

Crayfish: ***N***

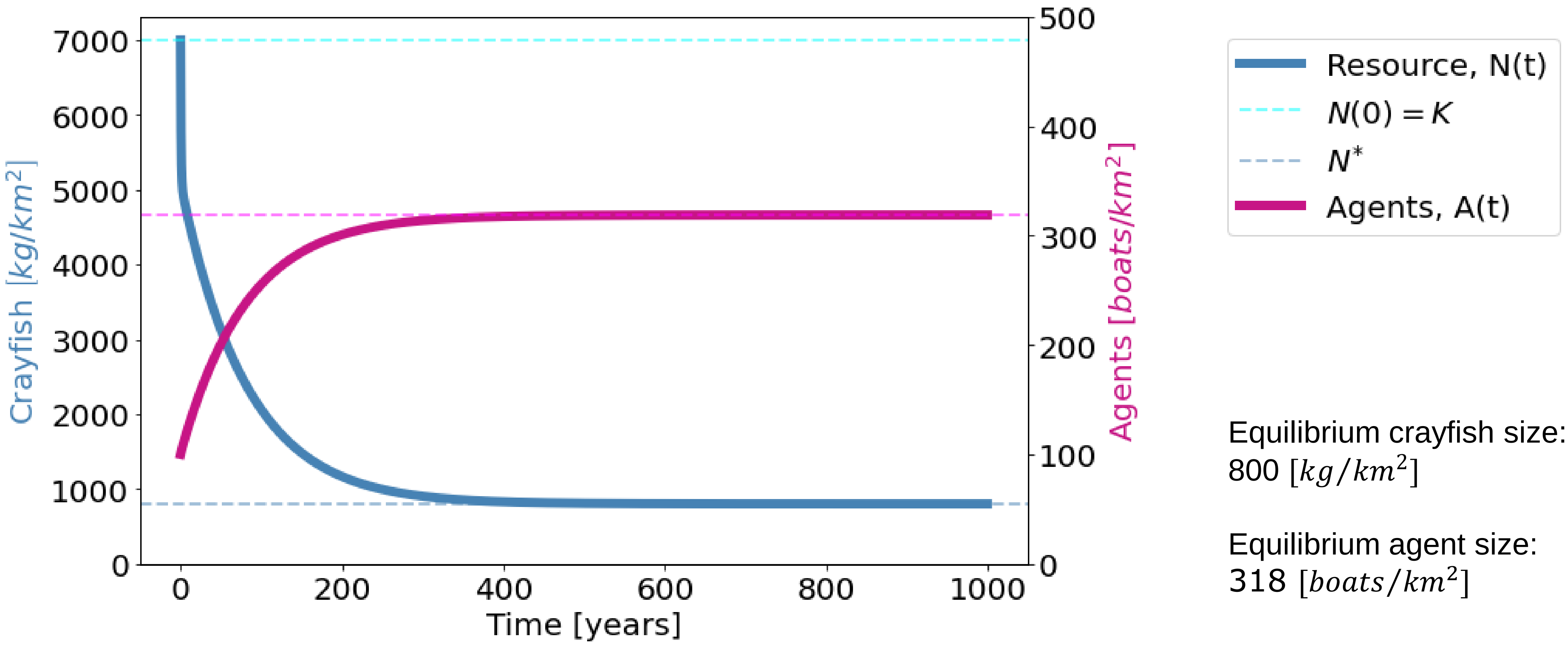
$$\frac{dN}{dt} = r\left(1 - \frac{N}{K}\right)N - bAN$$

AGENTS – Fishers: ***A***

$$\frac{dA}{dt} = w(\frac{PbN}{c} - 1)$$

Parameter	Calculation	Unit	Values
Resource density, N		$[kg/km^2]$	
Carrying capacity, K		$[kg/km^2]$	7000
Growth rate, r	$\mathrm{r} = br - d$	$[1/\mathrm{year}]$	1.8
Agent density, A		$[boats/km^2]$	
Clearance rate, b		$[km^2/boat/year]$	
Profit, P	$\mathrm{P} = pbN - C$	$[\$/year/boat]$	
Cost of exploitation, C		$[\$/year/boat]$	
Effort increase rate, w		$[boats/km^2/year]$	
Price, p		$[\$/kg]$	

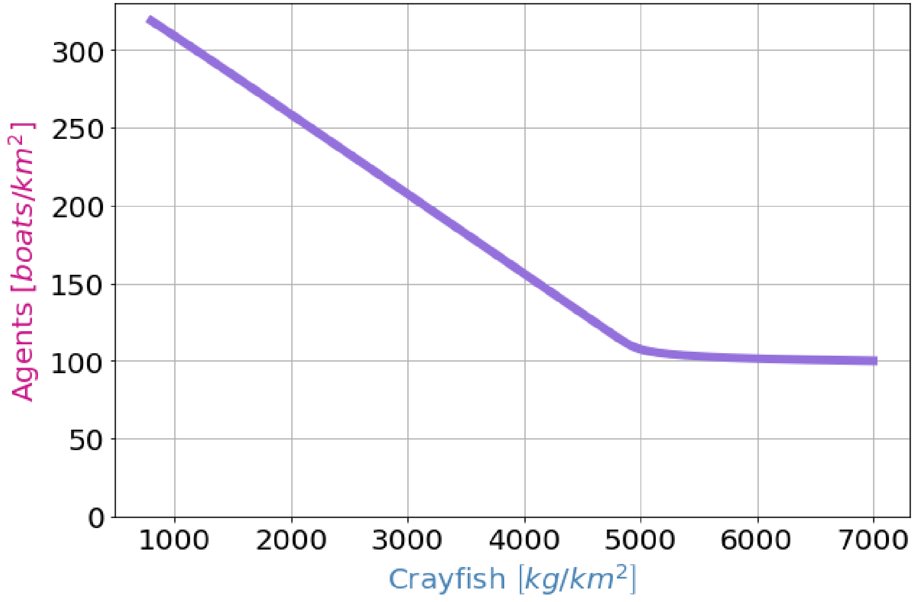
Exploitation by agents

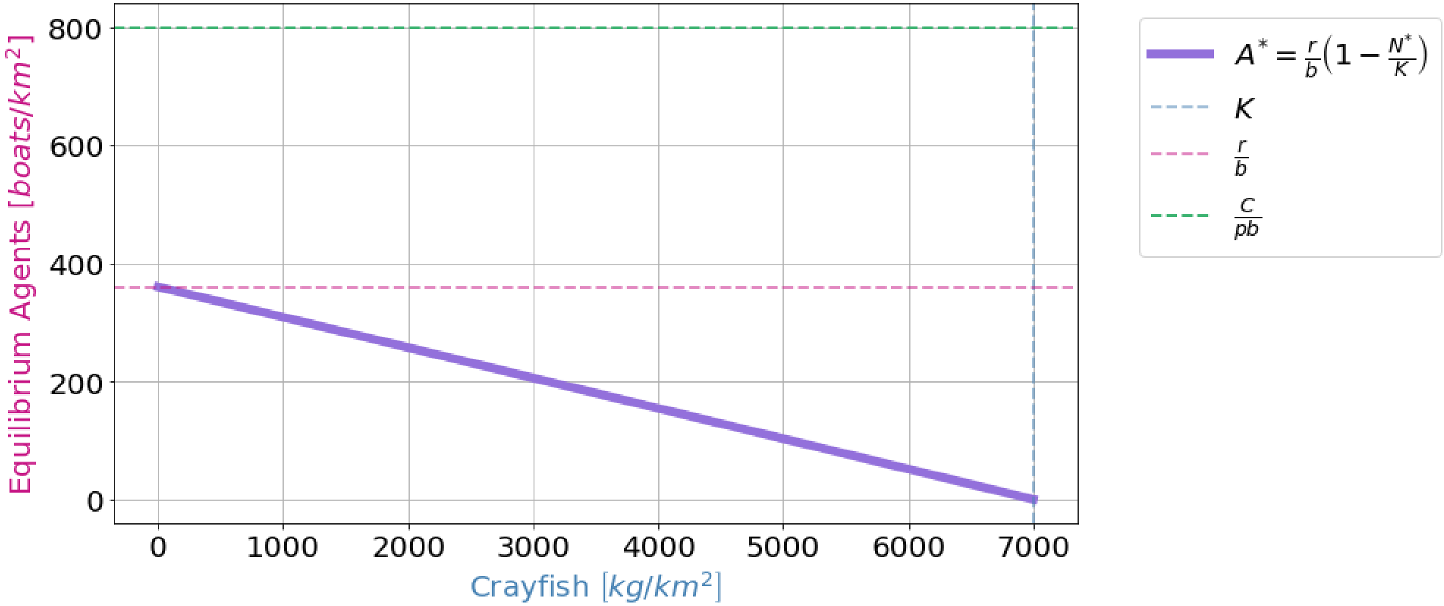


Equilibrium crayfish size:
800 $[kg/km^2]$

Equilibrium agent size:
318 $[boats/km^2]$

Exploitation by agents

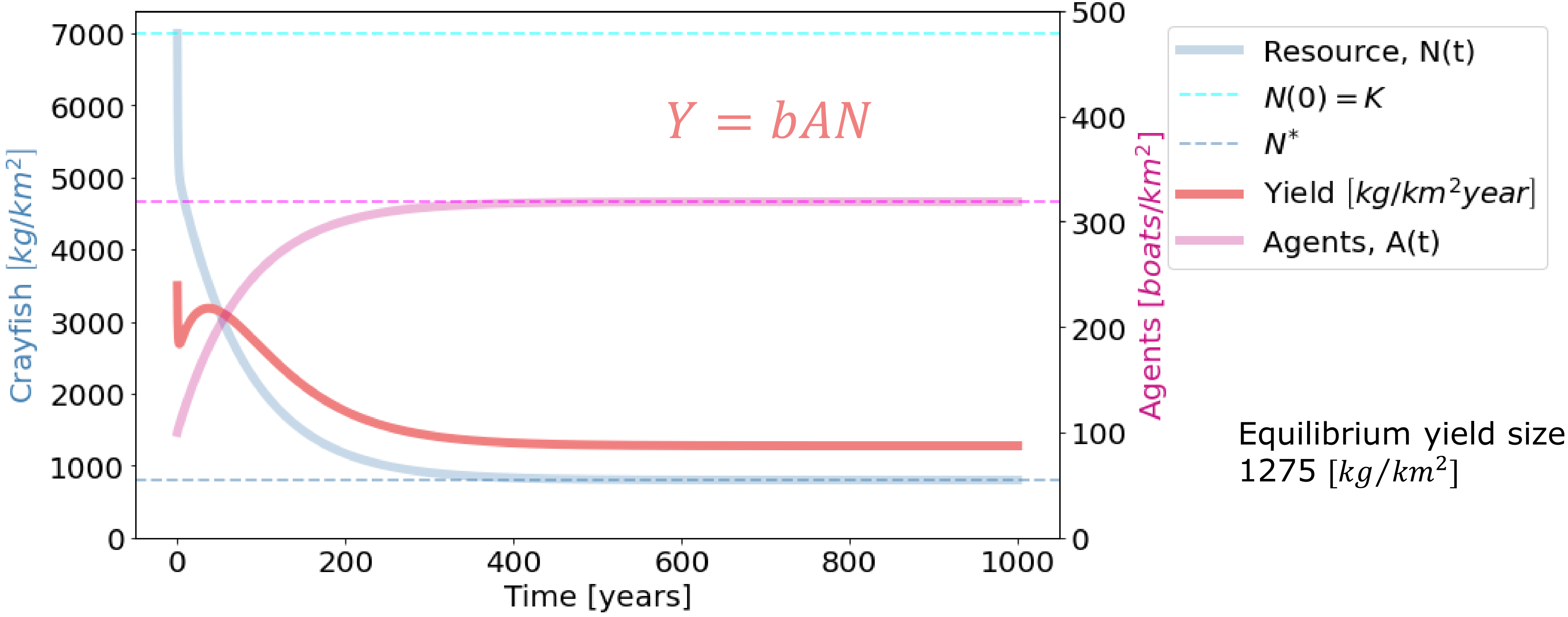




Equilibrium crayfish size:
800 $[kg/km^2]$
(negative agent size)

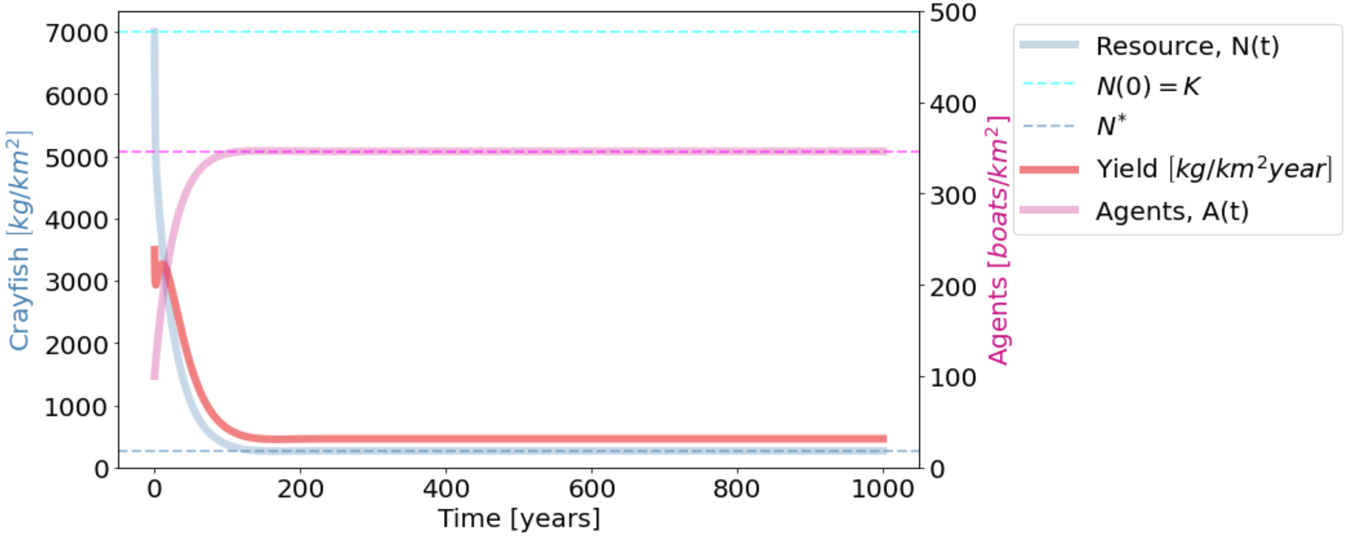
Equilibrium agent size:
318 $[boats/km^2]$ (zero resourse size)

Exploitation by agents

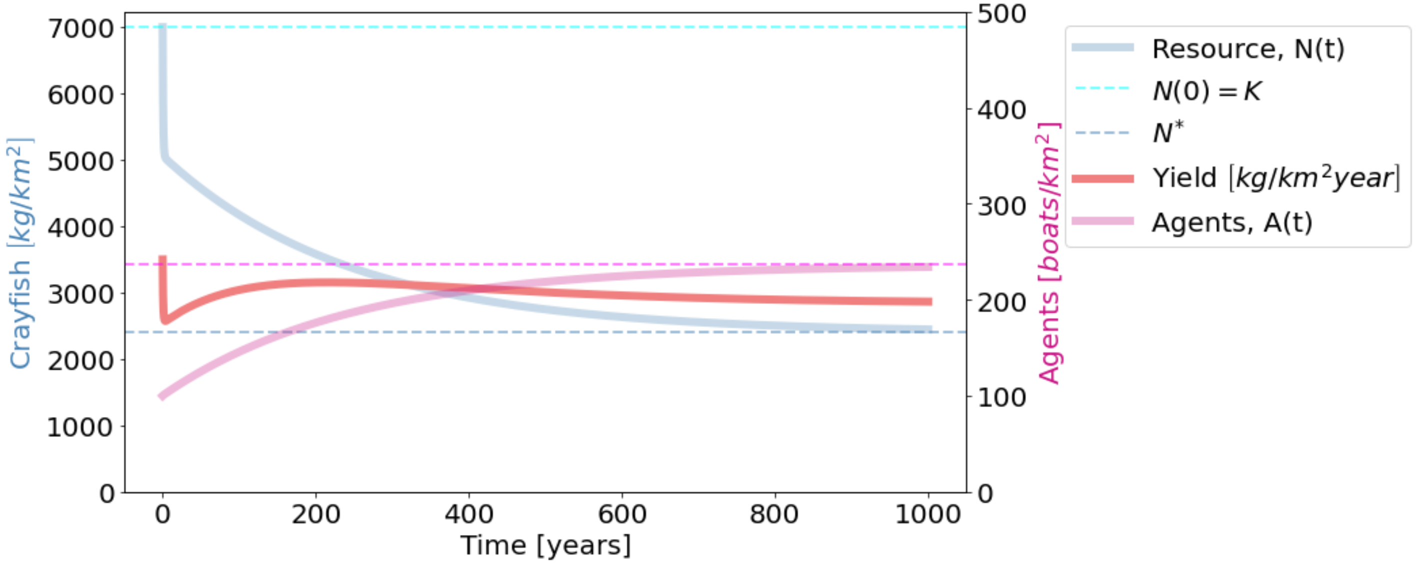


Equilibrium yield size:
1275 $[kg/km^2]$

Exploitation by agents

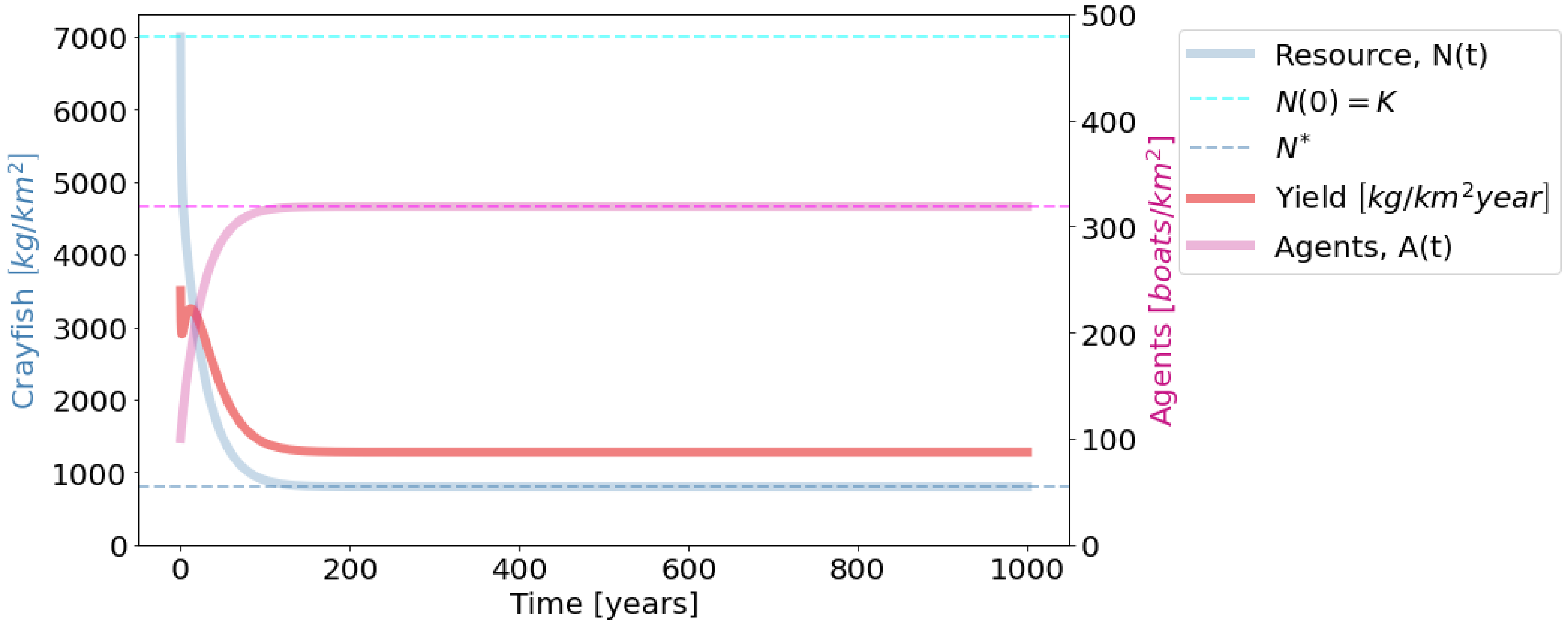


If the price triples (p = 15), the agents are doing better and have higher yield



If the cost triples (C = 60), the crayfish are doing better, and the agents have lower yield and numbers

Exploitation by agents



If the rate at which effort is increased triples (w = 1.5), the system reaches equilibrium faster

Exploitation by agents, python code

```
# Parameters:
K = 7000 # Carrying capacity
b = 10/5 # Birth rate # Unit: [1 / year]
d = 1/5 # Death rate # Unit: [1 / year]
r = b-d # Growth rate # Unit: [1 / year] # r = 1.8
N0 = K # Initial population equal to Carrying capacity
t = np.linspace(0, 1000, 10000) # in years
E = 0.8 # Effort # Exploitation  # Unit: [1/year]
w = 0.5 # rate at which effort is increased # Unit: [boat/sqkm/year]
b = 0.005 # clearance rate # Unit: [km^2/ boat/year]
p= 5 # price per catch # Unit: [dollar/ crayfish]
C = 20 #cost of exploitation per agent per time # Unit [dollar/year/boat]

params = (r, K, E, w, b, p )
init_state = [K, 100]

solve([r*(1-(Ns/K))*Ns - Ns*As*b, w*(p*b*Ns/C - 1)])

Y = lambda N, A: b*N*A
Ndot = lambda N: r*(1 - N/K)*N
Nstar_pred = lambda : C/(p*b)
Astar_pred = lambda Nst: (r/b)*(1-Nst/(K))
PI = lambda n: p*b*n - C # profit

def deriv_pred(state, t, *params):
    N, A = state
    r, K, E, w, b, p = params
    # dN_dt = (r*(1 - N/K) - E)*N
    dN_dt = Ndot(N) - Y(N, A)
    dA_dt = w*PI(N)/C
    return np.array([dN_dt, dA_dt])

sol = odeint(deriv_pred, init_state, t, params).T
N, A = sol
```

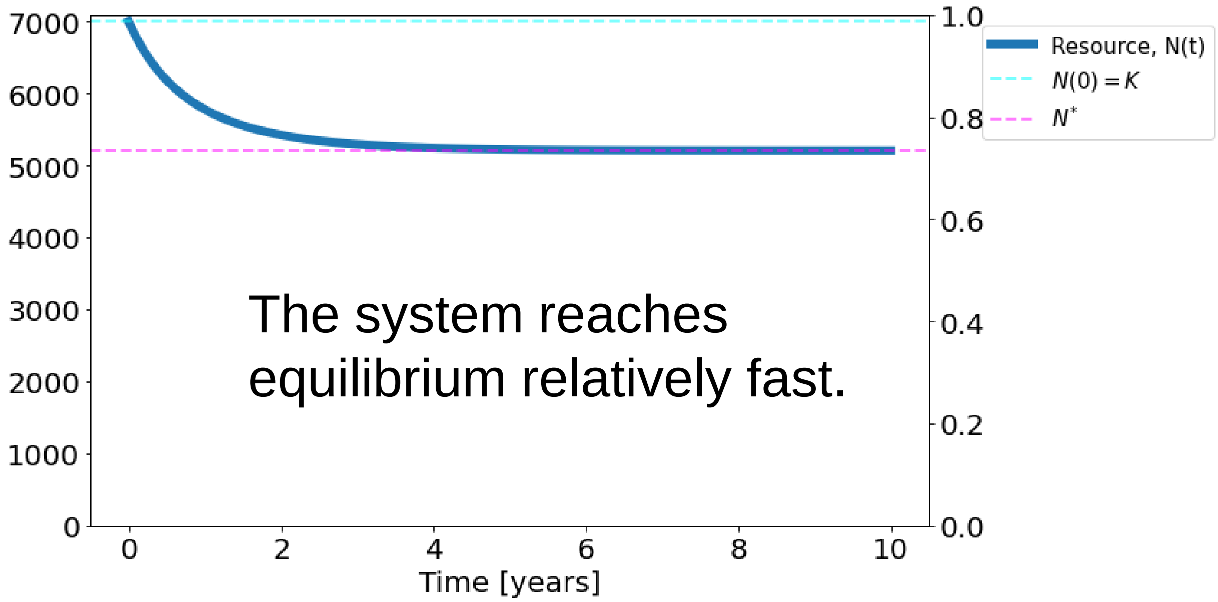
```
fig, ax1 = plt.subplots(1,1, figsize=(15, 6), tight_layout=True)
ax2 = ax1.twinx()

ax1.plot(t, N, label="Resource, N(t)", color='steelblue')
ax1.axhline(N[0], alpha=0.5, linestyle="--", color="cyan", linewidth=2, label="$N(0) = K$")
ax2.plot(t, A, label="Agents, A(t)", color='mediumvioletred')
ax1.axhline(Nstar_pred(), alpha=0.5, linestyle="--", color='steelblue', linewidth=2, label="$N^{*}$")
ax2.axhline(Astar_pred(Nstar_pred()), alpha=0.5, linestyle="--", color="magenta" , linewidth=2, label="$A^{*}$")
ax1.plot(t, Y(N,A), label='Yield $\\left[kg/km^{2} year\\right]$',  color = 'lightcoral')
ax1.set_ylabel('Crayfish $\\left[kg/km^{2}\\right]$', color='steelblue')
ax2.set_ylabel('Agents $\\left[boats/km^{2}\\right]$', color='mediumvioletred')
ax1.set_xlabel('Time [years]')
ax1.plot([], [], color='mediumvioletred', label="Agents, A(t)")

ax1.legend(bbox_to_anchor=(1.2, 1.), loc=2, borderaxespad=0.7, fontsize=20)
ax1.set_ylim(bottom=0)
ax2.set_ylim(bottom=0)
ax2.set_ylim(0, 500)

fig, ax1 = plt.subplots(1,1, figsize=(9, 6), tight_layout=True)
ax1.plot(N, A, color='mediumpurple')
ax1.set_xlabel('Crayfish $\\left[kg/km^{2}\\right]$', color='steelblue')
ax1.set_ylabel('Agents $\\left[boats/km^{2}\\right]$', color='mediumvioletred')
ax1.set_ylim(bottom=0)
ax1.grid()
plt.show()

print(min(N))
print(max(A))
print(Nstar_pred())
print(Astar_pred(Nstar_pred()))

ns = np.linspace(0, K, 1001)
fig, ax1 = plt.subplots(1,1, figsize=(14, 6), tight_layout=True)
ax1.plot(ns, Astar_pred(ns), color='mediumpurple', label="$A^{*} = \\frac{r}{b} \\left( 1- \\frac{N^{*}}{K} \\right)$")
ax1.axvline(K, alpha=0.5, linestyle="--", color='steelblue', linewidth=2, label="$K$")
ax1.axhline(r/b, alpha=0.5, linestyle="--", color='mediumvioletred', linewidth=2, label="$\\frac{r}{b}$")
ax1.axhline(C/(p*b), linestyle="--", color='mediumseagreen', linewidth=2, label="$\\frac{C}{pb}$")
ax1.set_xlabel("$N^{*}$")
ax1.set_xlabel('Crayfish $\\left[kg/km^{2}\\right]$', color='steelblue')
ax1.set_ylabel('Equilibrium Agents $\\left[boats/km^{2}\\right]$', color='mediumvioletred')
ax1.legend(bbox_to_anchor=(1.05, 1.), loc=2, borderaxespad=0.5, fontsize=20)
ax1.grid()
```

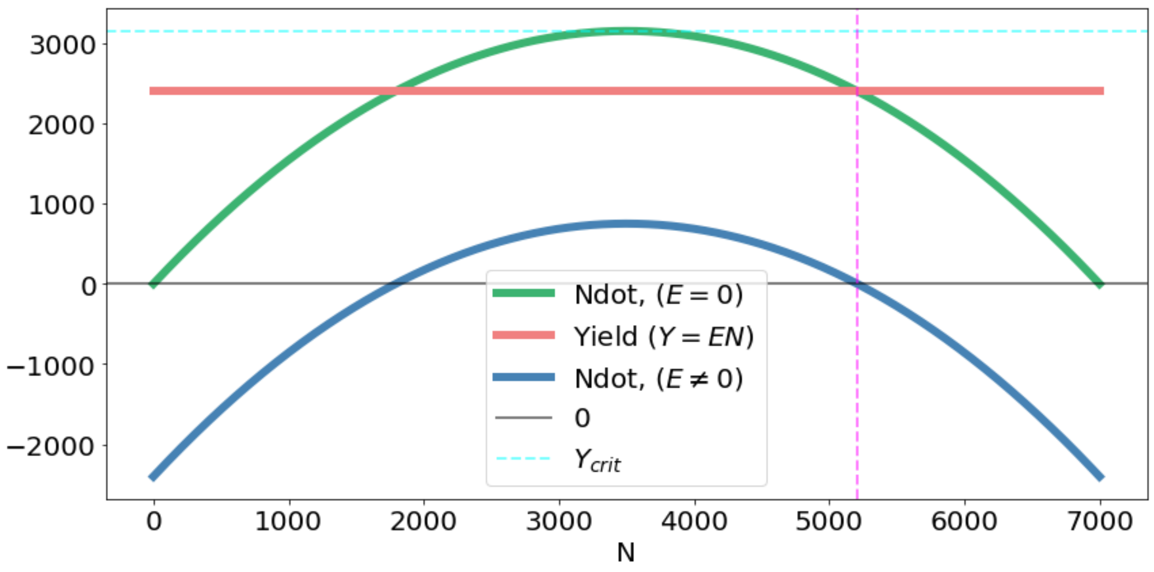
Yield independent of abundance

Crayfish: N

$$\frac{dN}{dt} = r\left(1 - \frac{N}{K}\right)N - zE$$

Parameter	Calculation	Unit	Value
Resource density, N		$[kg/km^2]$	
Carrying capacity, K		$[kg/km^2]$	7000
Growth rate, r	$\mathrm{r} = br - d$	$[1/\mathrm{year}]$	1.8
Effort, E		$[1/\mathrm{year}]$	0.8





Exploitation by agents, python code

```
# Parameters:
K = 7000 # Carrying capacity
b = 10/5 # Birth rate # Unit: [1 / year]
d = 1/5 # Death rate # Unit: [1 / year]
r = b-d # Growth rate # Unit: [1 / year] # r = 1.8
N0 = K # Initial population equal to Carrying capacity
t = np.linspace(0, 10, 100) # in years
E = 0.8 # Effort # Exploitation  # Unit: [1/year]


w = 0.5 # rate at which effort is increased # Unit: [boat/sqkm/year]
b = 0.005 # clearance rate # Unit: [km^2/ boat/year]
p= 5 # price per catch # Unit: [dollar/ crayfish]
C = 20 #cost of exploitation per agent per time # Unit [dollar/year/boat]


params = (r, K, E, w, b, p )
init_state = [K, 100]

Y = lambda : 3000*E ## TODO: choose parameters wisely!
Ndot = lambda N: r*(1 - N/K)*N
Nstar_const_Y = lambda : (1/2)*(K + np.sqrt((K*K) - 4*K*Y()/r))
# Astar_const_Y = lambda Nst: (r/b)*(1-Nst/(K))
#PI = lambda n: p*b*n - C # profit

def deriv_const_Y(state, t, *params):
    N, A = state
    r, K, E, w, b, p = params
    # dN_dt = (r*(1 - N/K) - E)*N
    dN_dt = Ndot(N) - Y()
    dA_dt = w*PI(N)/C
    return np.array([dN_dt, dA_dt])

params = (r, K, E, w, b, p)
sol = odeint(deriv_const_Y, init_state, t, params).T
N, A = sol
```

```
fig, ax1 = plt.subplots(1,1, figsize=(12, 6), tight_layout=True)
ax2 = ax1.twinx()
# ax1.grid()

ax1.plot(t, N, label="Resource, N(t)")
ax1.axhline(N[0], alpha=0.5, linestyle="--", color="cyan", linewidth=2, label="$N(0) = K$")
# ax1.plot(t, A, label="Agents, A(t)")
ax1.axhline(Nstar_const_Y(), alpha=0.5, linestyle="--", color="magenta", linewidth=2, label="$N^{*}$")
#ax1.axhline(Astar_pred(Nstar_pred()), alpha=0.5, linestyle="--", color="lime", linewidth=2, label="$A^{*}$")

# ax1.set_ylabel('Resources [FILL]')
# ax2.set_ylabel('Agents [FILL]', color = 'mediumseagreen')
ax1.set_xlabel('Time [years]')
ax1.set_ylim(bottom=0)

ax1.legend(bbox_to_anchor=(1.05, 1.), loc=2, borderaxespad=0.5, fontsize=15)

ns = np.linspace(0, K, 1001)

Y_crit = K*r/4 # for higher yields than y_crit,
                    # no real solutions exists => no stabile real fixed-points,
                    # everything is exterminated

fig, ax1 = plt.subplots(1,1, figsize=(12, 6), tight_layout=True)
ax1.plot(ns, Ndot(ns), label="Ndot, ($E = 0$)", color='mediumseagreen')
ax1.plot(ns, [Y()]*len(ns), label="Yield ($Y = E N$)", color = 'lightcoral')
ax1.plot(ns, Ndot(ns) - Y(), label="Ndot, ($E \\neq 0$)", color='steelblue')
ax1.axhline(0, alpha=0.5, linestyle="-", color="black", linewidth=2, label="0")
ax1.axhline(Y_crit, alpha=0.5, linestyle="--", color="cyan", linewidth=2, label="$Y_{crit}$")
# ax1.plot(ns, Ndot(ns) - Y(), label="Ndot, ($E \\neq 0$)")

ax1.axvline(Nstar_const_Y(), alpha=0.5, linestyle="--", color="magenta", linewidth=2, )
ax1.set_xlabel('N')
ax1.legend()
```