Group: Amalia Bogri, Christian Berrig & Jonas Bolduan

Seasonal succession

Seasonal succession: model parameters

Parameter	Unit	Value
Concentration of nutrients, N	$\text{mM}\ m^{-3}$	
Concentration of phytoplankton, P	$\text{mM}\ m^{-3}$	
Concentration of zooplankton, Z	$\text{mM}\ m^{-3}$	
Time, t	$days$	$365 \cdot 5$
Latitude, ϕ	degrees	47
Diffusion rate, m	$m\ day^{-1}$	0.3
Depth of mixed layer, M	m	60
Concentration of deep nutrients, N_0	$\text{mM}\ m^{-3}$	10
Concentration of phytoplankton when grazing stops, P_0	$\text{mM}\ m^{-3}$	0.1
Half-saturation coefficient: concentration where phytoplankton feed at half their maximum rate, H_P	$\text{mM}\ m^{-3}$	0.5
Half-saturation coefficient, concentration where zooplankton feed at half their maximum rate: H_Z	$\text{mM}\ m^{-3}$	1.0
Phytoplankton metabolic loss rate, r	day^{-1}	0.07
Zooplankton mortality rate, d	day^{-1}	0.07
Grazing efficiency , ε	unitless	0.5
Zooplankton's maximum grazing rate of phytoplankton, C_{max}	day^{-1}	1.0

Seasonal succession: equations

$$\frac{\partial N}{\partial t} = -\left[g(t,\phi,M)\left(\frac{N}{H_P + N}\right) - r\right]P + \frac{m}{M}(N_0 - N)$$

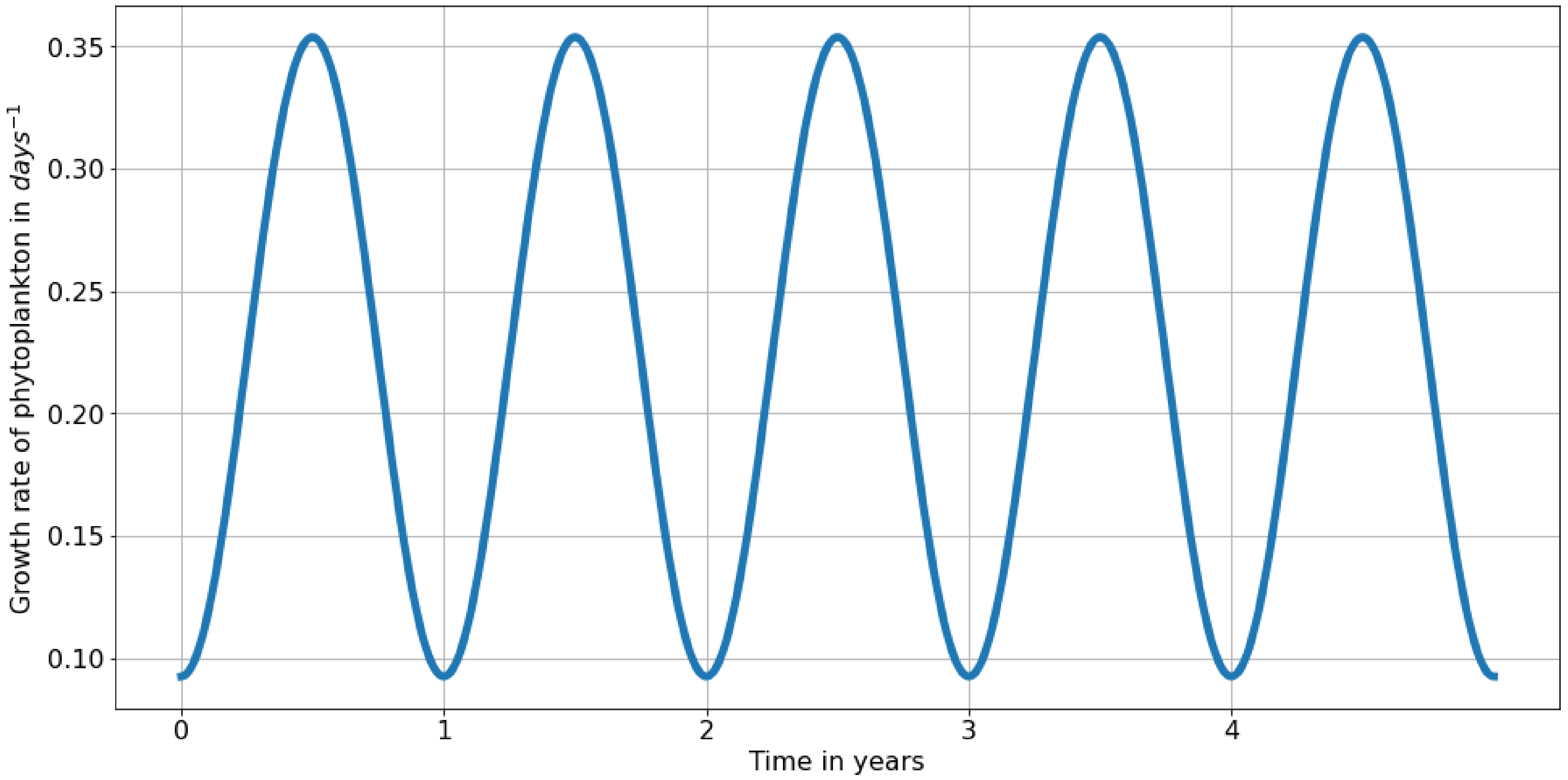
$$\frac{\partial P}{\partial t} = +\left[g(t,\phi,M)\left(\frac{N}{H_P + N}\right) - r\right]P - max\left\{0, C_{max}\frac{P - P_0}{H_Z + P - P_0}\right\}Z - \frac{m}{M}P$$

$$\frac{\partial Z}{\partial t} = \varepsilon\, max\left\{0, C_{max}\frac{P - P_0}{H_Z + P - P_0}\right\}Z - dZ$$

$$g(t,\phi,M) = \exp(-0.025m^{-1}M)\left[1 - 0.8\sin\left(\frac{\pi\phi}{180°}\right)\cos\left(2\pi\frac{t}{365\text{ days}}\right)\right]$$

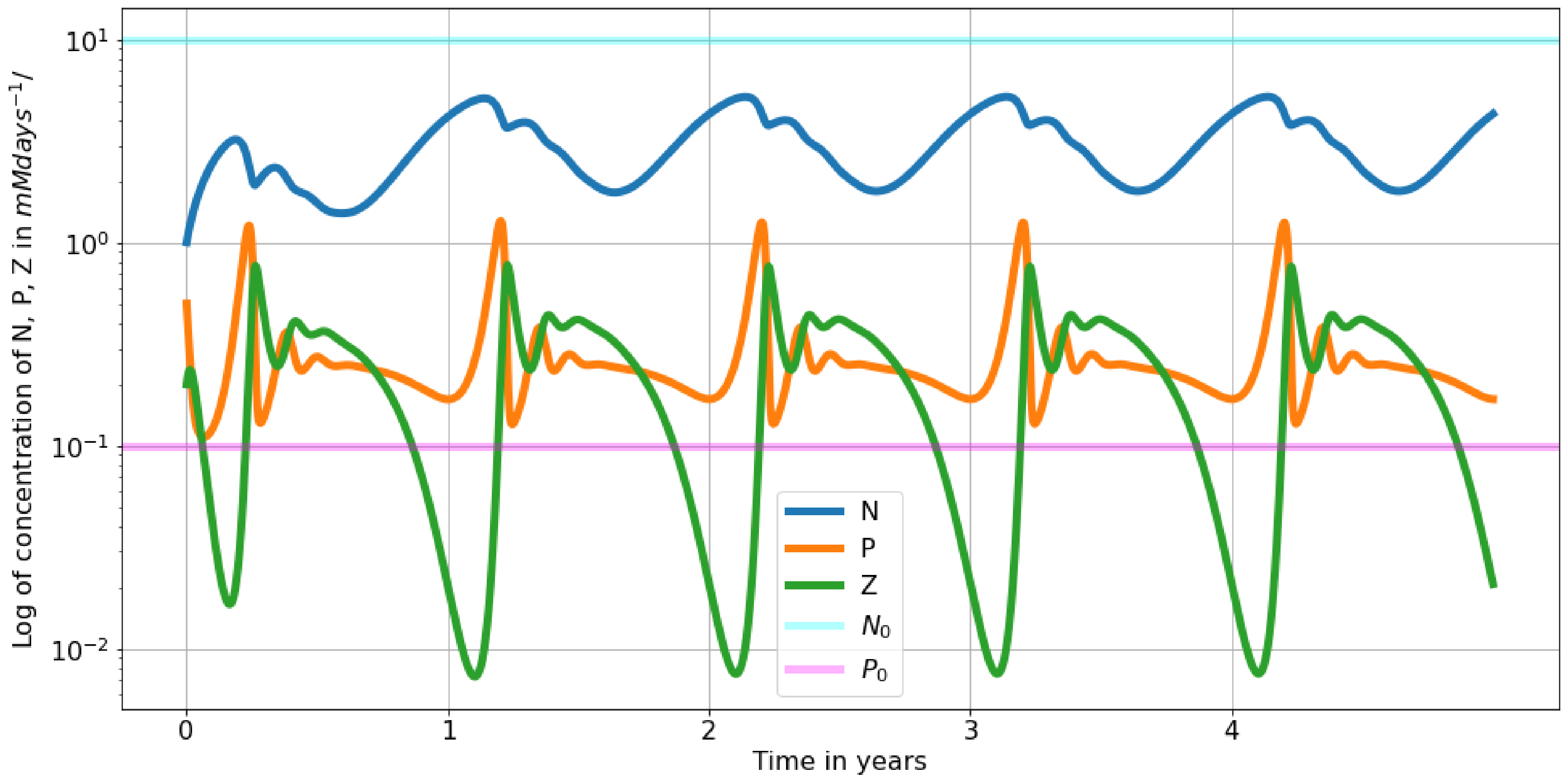
Seasonal succession: growth rate as function of time

$$g(t,\phi,M) = \exp(-0.025m^{-1}M)\left[1 - 0.8\sin\left(\frac{\pi\phi}{180°}\right)\cos\left(2\pi\frac{t}{365\text{ days}}\right)\right]$$



Oscillates between approximately 0.1 and 0.3

Seasonal succession of nutrients, phyto & zooplankton



After the first year, we see stable oscillations.

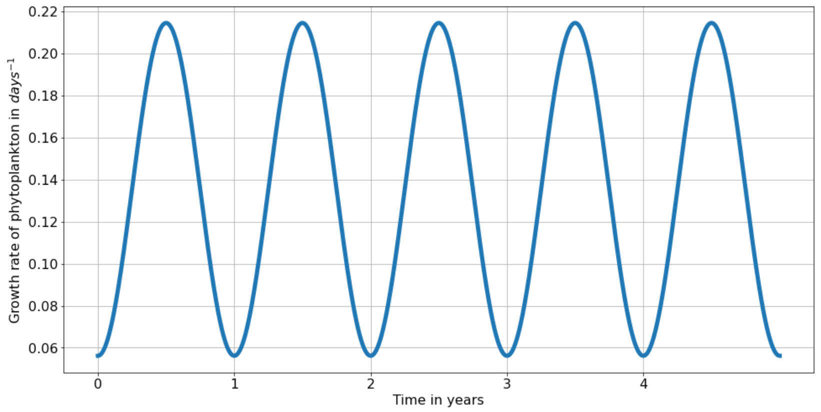
Each year, the nutrients peak first, followed by the phytoplankton that feeds on them, and then the zooplankton that grazes on phytoplankton.

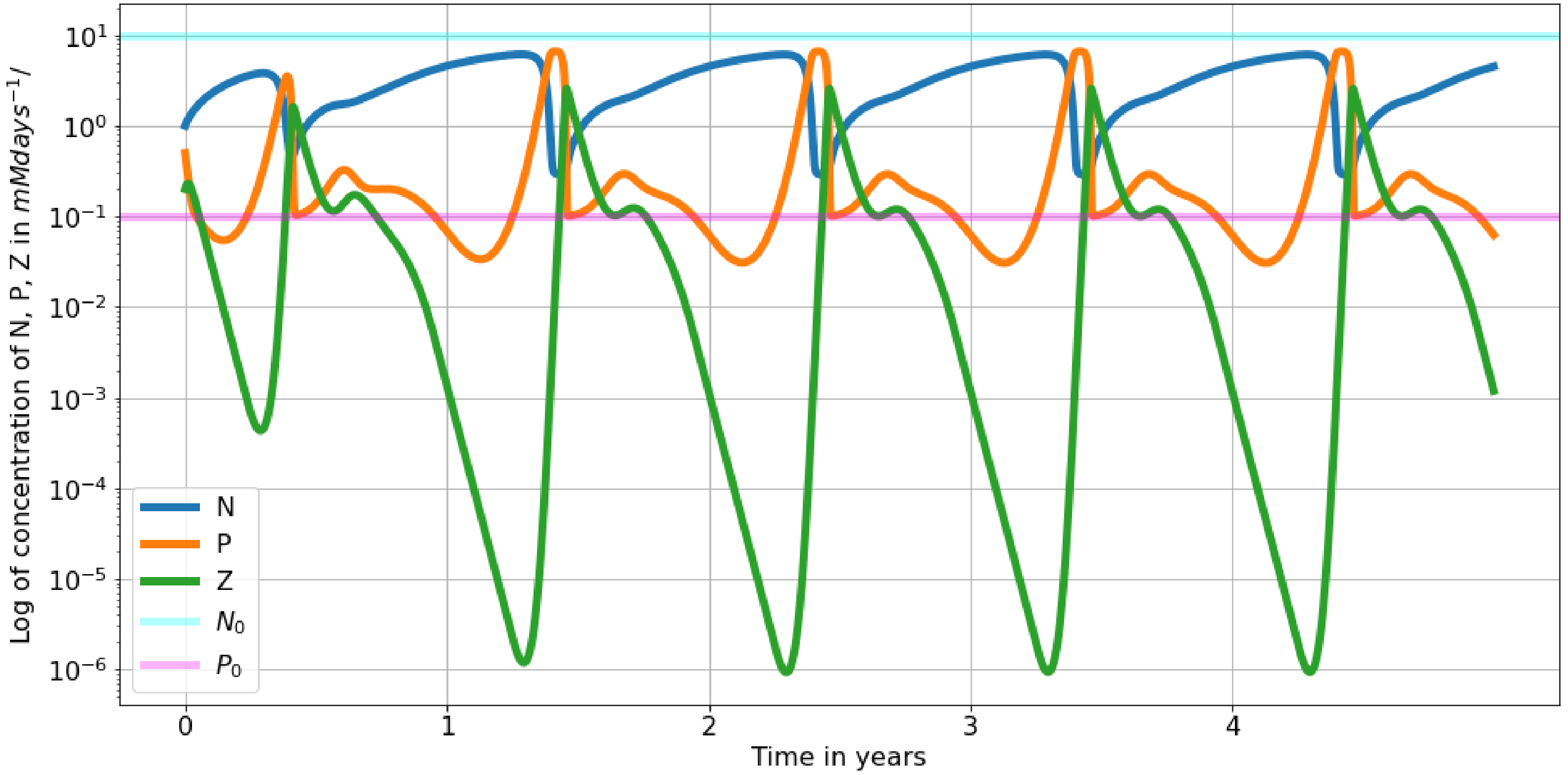
Likewise, the nutrients decrease first (used by phytoplankton), which decreases afterwards (grazed by zooplankton), which crashes last.

Nutrients recover and restart the seasonal succession due to the mixing with the deeper layer.

Seasonal succession: increase depth M (80 m)

Growth rate oscillates between approximately 0.6 and 0.22

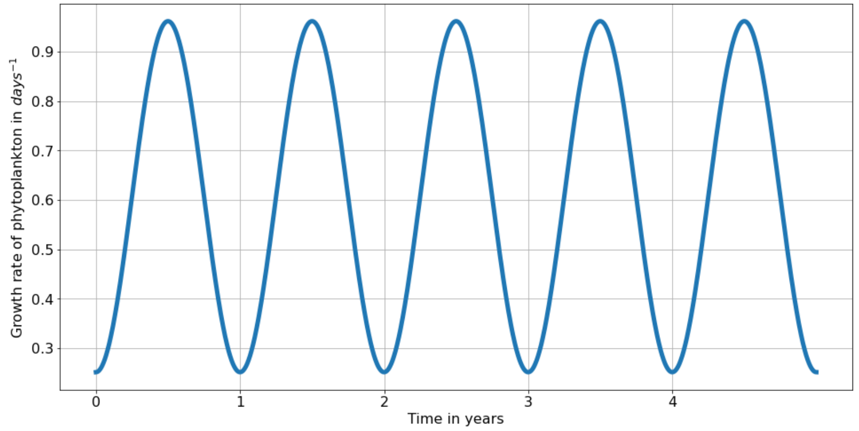


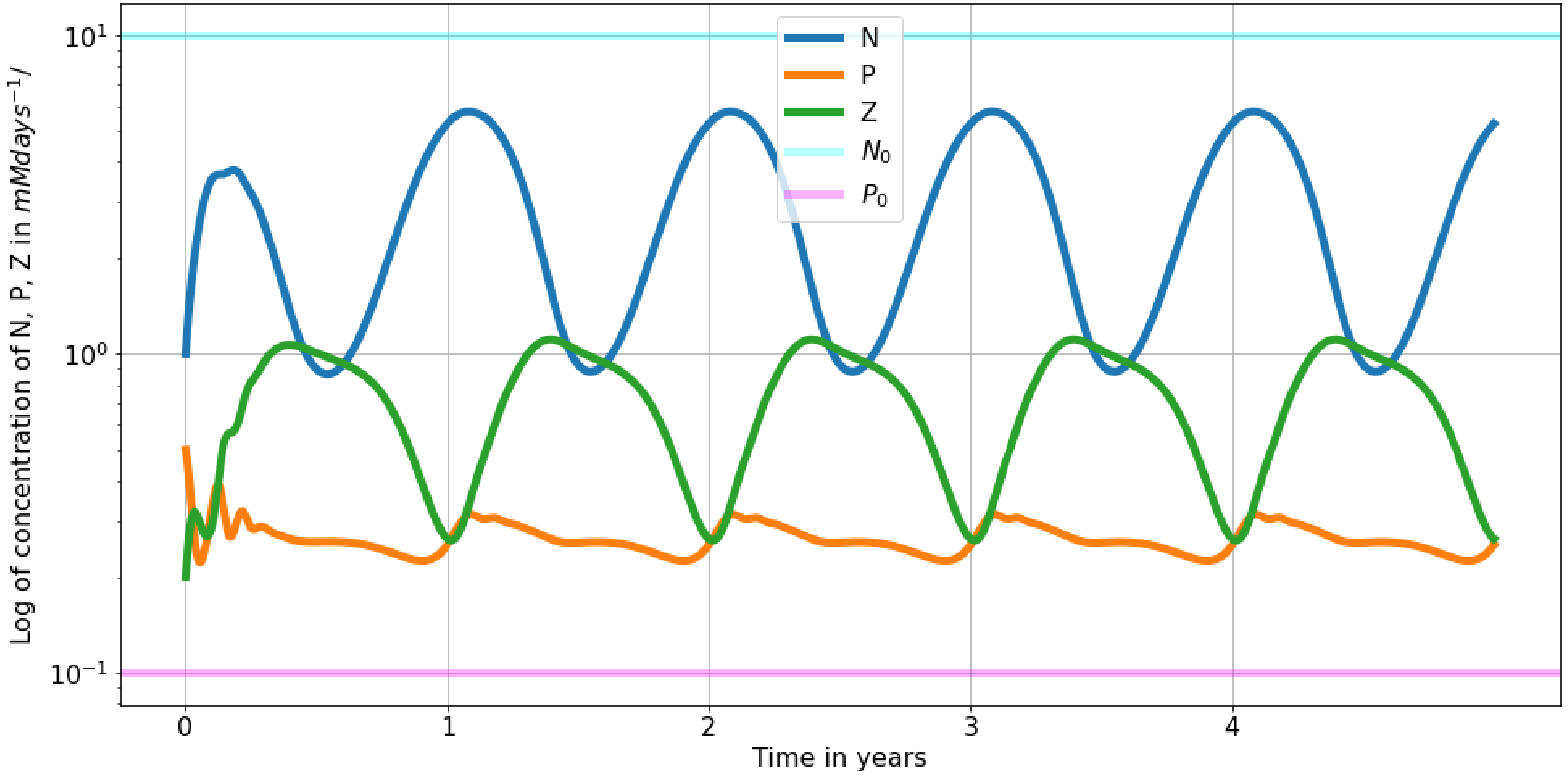


With M = 80 m, the oscillations are steeper, reaching higher maxima and lower minima compared to the ones with M= 60 m shown before.

Seasonal succession: increase depth M (20 m)

Growth rate oscillates between approximately 0.3 and 0.9

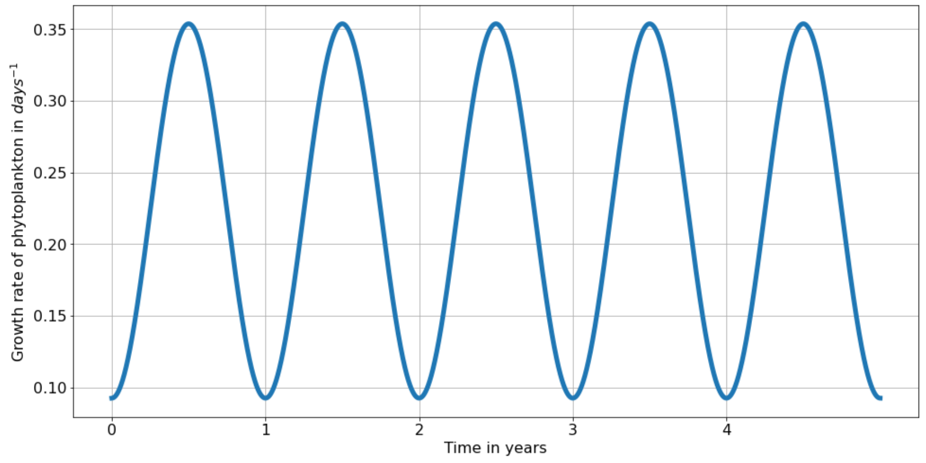


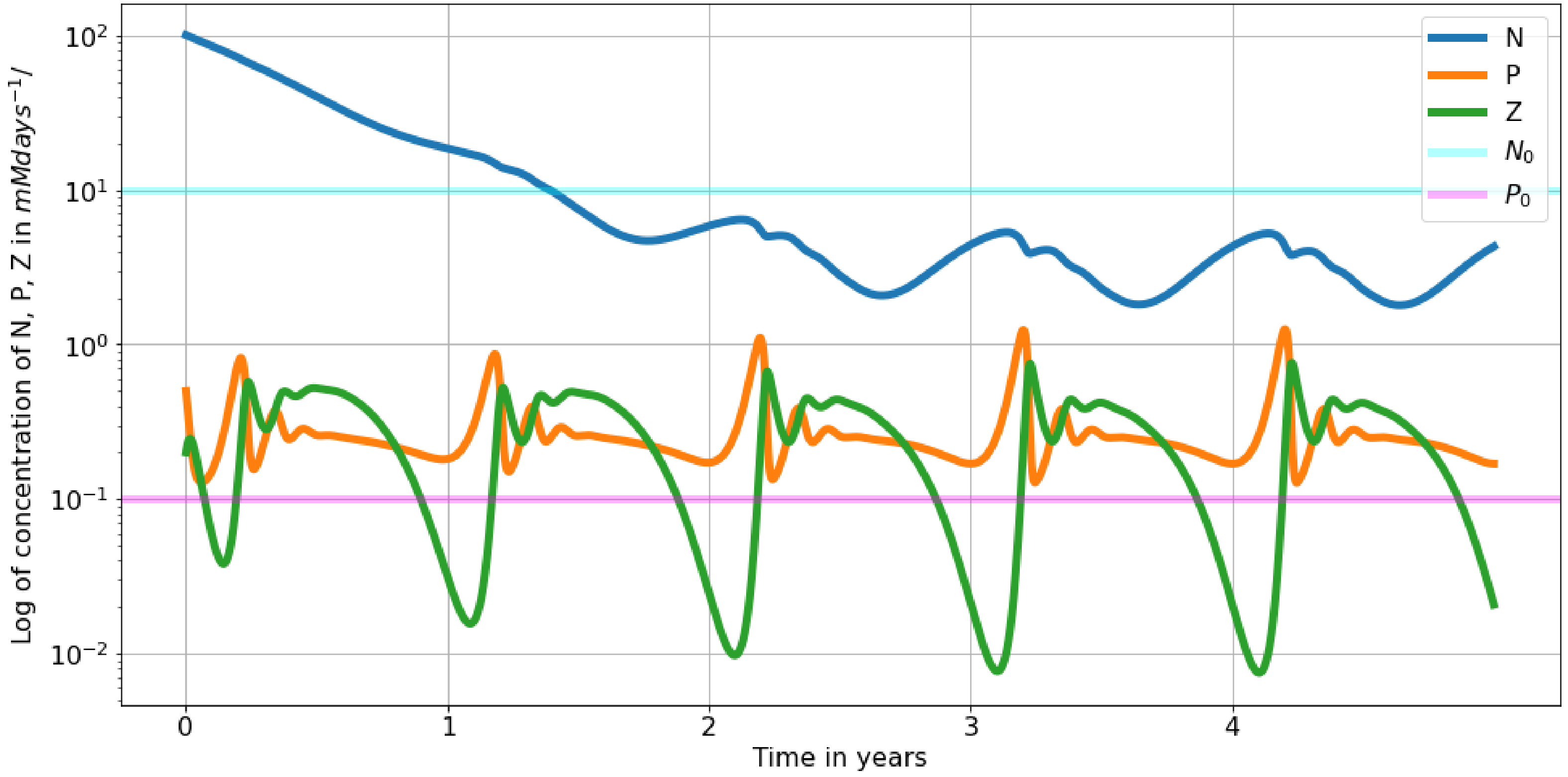


With M = 20 m, the oscillations are much smoother, reaching ‘lower’ maxima and ‘higher’ minima compared to the ones with M= 60 m and M= 80 m shown before.

Seasonal succession: dependence on N_0=100

As expected, the growth rate does not change from the original model.





As expected, after the initial two years, the model reaches it's oscillatory equilibrium. Therefore the amount of nutrients in the deep layer does not affect the long term production of the 3 components of the model.

Python code

1
```
# Import stuff:
from scipy.integrate import odeint
import matplotlib.pyplot as plt
import numpy as np
import scipy as sp
from sympy import Symbol
from sympy.solvers import solve

# This is for reasonable fontsize universally defined:
fs_label = 16
parameters = {
                'figure.titlesize': fs_label+6,
                'axes.labelsize': fs_label,
                'axes.titlesize': fs_label+4,
                'xtick.labelsize': fs_label,
                'ytick.labelsize': fs_label,
                'legend.fontsize': fs_label,
                'lines.linewidth': 5
             }
plt.rcParams.update(parameters)

# Time
years = 5
t_ticks = np.arange(0,365*years,365)
t_ticks_labels = np.arange(0, years)
t_list = np.linspace(0, 365*years, 10000)


params_dict=dict(
    H_P = 0.5, # mM m-3 # half-saturation coefficient: concentration where phytoplankton feed at half their maximum rate
    H_Z = 1.0, # mM m-3 # half-saturation coefficient: concentration where zooplankton feed at half their maximum rate
    P_0 = 0.1, # mM m-3 # concentration of phytoplankton below which grazing by zooplankton stops
    N_0 = 10, # mM m-3 ## concentration of deep nutrients
    C_max = 1.0, # d-1 # maximum grazing rate of zooplankton
    r = 0.07, # d-1 # phytoplankton metabolic loss rate # carefull!! extremely sensitive param.
    eps = 0.5, # unitless # grazing efficiency
    d = 0.07, # d-1 # loss of zooplankton to predators rate
    M = 60, # m # depth of mixed layer
    phi = np.pi*47/180, # latitude (normally as 47 # degrees), here phi is the azimuthal angle in radians, s.t. 0<=phi <=pi
    m = 0.3 # m d-1# diffusion rate #3 #1 #0.3 # carefull!! extremely sensitive param.
)
```

2
```
for k, v in params_dict.items():
    assign_str = f"{k} = {v}"
    exec(assign_str) # Q:is this to just take the dictionary content and turn it into
variables?
    print(assign_str)

params = tuple(params_dict.items())

N_init = 1
P_init = 0.5
Z_init = 0.2
state_init = [N_init, P_init, Z_init]

g = lambda t, phi, M: np.exp(-0.025*M)*(1- 0.8*np.sin(phi)*np.cos(t*(2*np.pi/365)))
uptake = lambda N, g, H_P=H_P, r=r: g*(N/(H_P + N)) - r
grazing = lambda P, P_0=P_0, C_max=C_max, H_Z=H_Z: max(0, C_max*(P-P_0)/(H_Z + (P - P_0)))


# Question 1: plot the maximum growth rate of phytoplankton as a function of time
fig, ax = plt.subplots(figsize=(16,8))
ax.plot(t_list, g(t_list, phi, M))
ax.set_xlabel('Time in years')
ax.set_ylabel('Growth rate of phytoplankton in $days^{-1}$')
ax.set_xticks(t_ticks)
ax.set_xticklabels(t_ticks_labels)
ax.grid()
plt.show()
```

For the rest of the model variants we just changed the parameters in the dictionary, and run the same code otherwise.

3
```
# Question 2: implement the model
def deriv(state, t, *params):
    for k, v in params:
        assign_str = f"{k} = {v}"
        exec(assign_str)
    N, P, Z = state
    Up = uptake(N, g(t, phi, M), H_P=H_P, r=r)
    Gr = grazing(P, P_0=P_0, C_max=C_max, H_Z=H_Z)
    dNdt = -Up*P + (m/M)*(N_0 - N) #+ exchange(N, N_0, m=m, M=M)
    dPdt = Up*P - Gr*Z - (m/M)*P # + exchange(P+P_0, P_0, m=m, M=M)
    dZdt = eps*Gr*Z - d*Z
    return np.array([dNdt, dPdt, dZdt])
ns = odeint(deriv, state_init, t_list, args=params)

fig, ax = plt.subplots(figsize=(16,8))
lbls = ["N", "P", "Z"]
for i, sol in enumerate(ns.T):
    ax.plot(t_list, sol, label=lbls[i])

ax.axhline(N_0, color="cyan", alpha=0.3, label="$N_{0}$")
ax.axhline(P_0, color="Magenta", alpha=0.3, label="$P_{0}$")
ax.set_xlabel('Time in years')
ax.set_ylabel('Log of concentration of N, P, Z in $ mMdays^{-1}$/')
ax.set_yscale("log")

ax.set_xticks(t_ticks)
ax.set_xticklabels(t_ticks_labels)
ax.legend()
ax.grid()
```