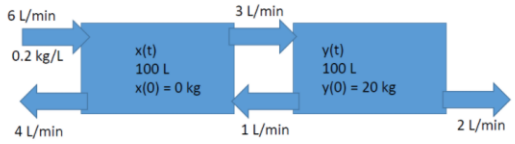
Final assignment: Gekoppelde tanks met zoutoplossingen

Opdracht

Twee grote tanks, elk gevuld met 100 Liter vloeistof, zijn met pijpleidingen aan elkaar verbonden. De vloeistof stroomt van tank A in tank B met een snelheid van 3 L/min en van B in A met 1 L/min. Een zoutoplossing met een concentratie van 0.2 kg/L stroomt met een snelheid van 6 L/min tank A in. De oplossing stroomt met een snelheid van 4 L/min tank A uit en verlaat met een snelheid van 2 L/min tank B.



Ik heb ervoor gekozen om twee ODE solvers te implementeren:

- Forward Euler
- Heun's method

Forward Euler:

$$x_{k+1} = x_k + hf(t_k, x_k)$$

Het implementeren van forward Euler is relatief makkelijk. Eerst bepalen we welke variabelen we nodig hebben:

```
def solve_with_forward_euler(f, h: float, x_min: float, x_max: float) -> list:
    """
    solves a set of two ODEs using forward euler

    :param f: The function that we use to calculate the derivative
    :param h: Stepsize
    :param x_min: Minimum x-axis value
    :param x_max: Maximum x-axis value (not inclusive)
    :return: a list of floats for tank1, a list of floats for tank2
    """
```

Vervolgens stellen we een differentieerfunctie op die ons de afgeleiden, gebaseerd op bepaalde y-waarde(s), aanlevert. Ons probleem heeft geen lineair verband dus we hoeven geen x (tijd) variabele mee te geven. De functie ziet er als volgt uit:

```
def f(yt1: float, yt2: float) -> list:
    """
    :param yt1: current y value of tank 1
    :param yt2: current y value of tank 2
    :return: deriv tank 1 , deriv tank 2
    """

    st1 = 1.2 + (1 / 100 * yt2) - (3 / 100 * yt1) - (4 / 100 * yt1)
    st2 = (3 / 100 * yt1) - (2 / 100 * yt2) - (1 / 100 * yt2)

    return st1, st2
```

Vervolgens initialiseren we wat lijsten om de nieuwe waardes bij te houden en gebruiken we forward Euler:

```
t = np.arange(x_min, x_max, h)

t1 = np.zeros(len(t))
t2 = np.zeros(len(t))

# tank 2 starts with a salt weight of 20
t2[0] = 20

# use forward euler to calculate the new values
for i in range(len(t) - 1):
    t1[i + 1] = t1[i] + f(t1[i], t2[i])[0] * h
    t2[i + 1] = t2[i] + f(t1[i], t2[i])[1] * h

return t1, t2
```

Heun's method:

$$\text{Slope}_{\text{left}} = f(x_i, y_i)$$
$$\text{Slope}_{\text{right}} = f(x_i + h, y_i + hf(x_i, y_i))$$
$$\text{Slope}_{\text{ideal}} = \frac{1}{2}(\text{Slope}_{\text{left}} + \text{Slope}_{\text{right}})$$

$$y_{i+1} = y_i + h\text{Slope}_{\text{ideal}}$$

Gelukkig voor ons gebruikt Heuns method dezelfde inputs als forward Euler. Het enige deel wat dus anders is, is het deel in de for-loop:

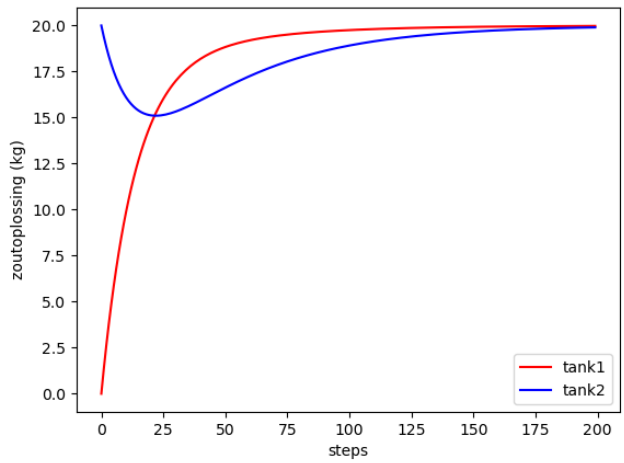
```
for i in range(len(t) - 1):
    sl = h * f(t1[i], t2[i])[0]
    sr = h * f(t1[i] + sl, t2[i])[0]
    t1[i + 1] = t1[i] + (sl + sr) / 2

    sl = h * f(t1[i], t2[i])[1]
    sr = h * f(t1[i] + sl, t2[i])[1]
    t2[i + 1] = t2[i] + (sl + sr) / 2
```

Resultaat

De gegenereerde resultaten van de twee methodes lijken erg veel op elkaar, hier zijn bijvoorbeeld de laatste vijf y-waardes van tank 1 voor de methodes:

```
[19.97655882 19.97711085 19.97764988 19.9781762  19.97869012]
[19.97549105 19.97607321 19.97664153 19.97719633 19.97773795]
```

De resultaten zijn op de grafiek hieronder geplot:



Je kan dus duidelijk zien dat tank 2 met een hogere zoutconcentratie start maar na ongeveer 22 minuten wordt ingehaald door tank 1. Het is ook logisch dat de som van de zoutoplossing in beiden tanks nooit boven de 40 komt omdat de zoutoplossing een concentratie van 0.2kg/L heeft en er in totaal 200 liter water aanwezig is.