BEE 4750/5750 Homework 2

Akanksha Srivastava (as2752)

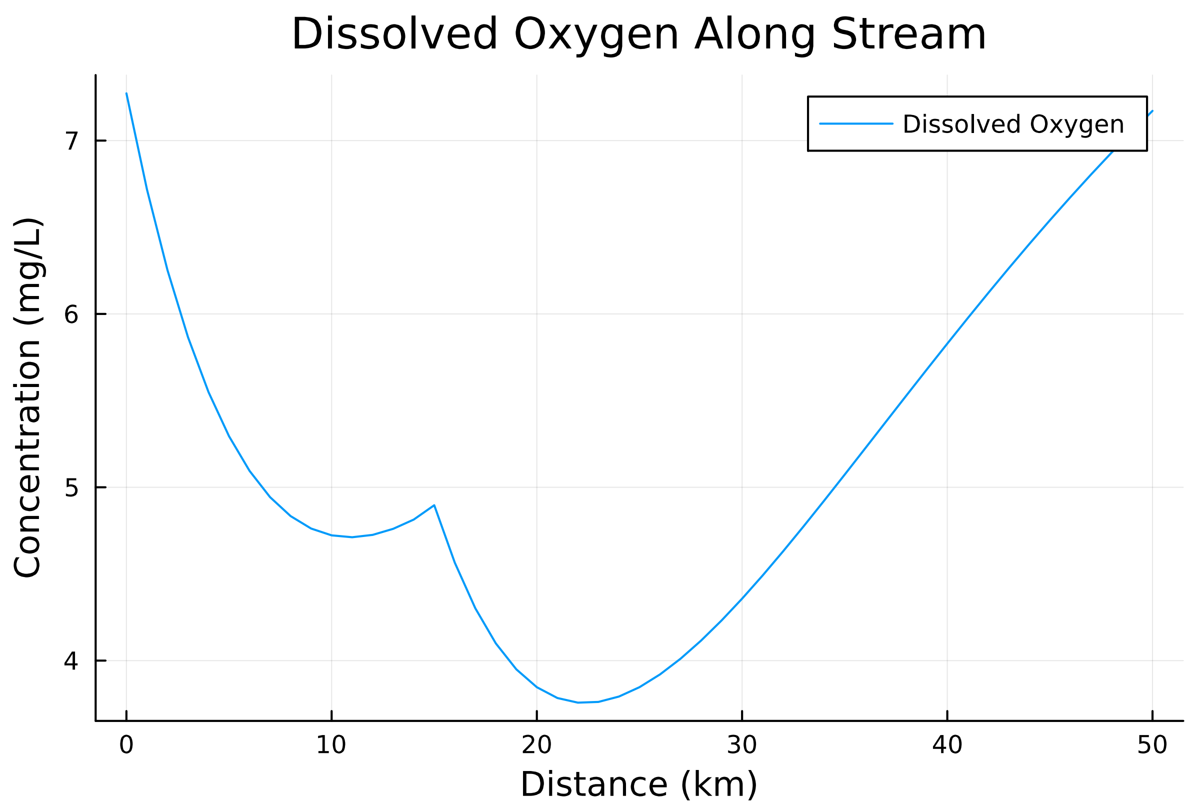
2022-09-25

Problem 1

Problem 1.1

The below code calculates the dissolved oxygen concentration over $0 - 50km$ from waste stream 1 and plots the resulting values.

```
julia> # Given Information
       Qr = 100000; #m3/day, river

julia> Q1 = 10000; #m3/day, waste stream 1

julia> Q2 = 15000; #m3/day, waste stream 2

julia> DOr = 7.5; #mg/L, river

julia> DO1 = 5; #mg/L, waste stream 1

julia> DO2 = 5; #mg/L, waste stream 2

julia> CBODr = 5; #mg/L, river

julia> CBOD1 = 50; #mg/L, waste stream 1

julia> CBOD2 = 45; #mg/L, waste stream 2

julia> NBODr = 5; #mg/L, river

julia> NBOD1 = 35; #mg/L, waste stream 1

julia> NBOD2 = 35; #mg/L, waste stream 2

julia> U = 6; #km/day

julia> ka = 0.55; #1/day, reaeration rate

julia> kc = 0.35; #1/day, CBOD decay rate

julia> kn = 0.25; #1/day, NBOD decay rate

julia> Cs = 10; #mg/L, saturated oxygen conc
julia> # Problem Setup
       x = 0:1:50; #km, downstream distance from waste source 1

julia> DO = zeros(length(x)); #mg/L, initialization

julia> CBOD = zeros(length(x)); #mg/L, initialization

julia> NBOD = zeros(length(x)); #mg/L, initialization

julia> C0 = (Qr*DOr + Q1*DO1) / (Qr+Q1); #mg/L, DO conc after waste stream 1

julia> B0 = (Qr*CBODr + Q1*CBOD1) / (Qr+Q1); #mg/L, CBOD conc after waste
stream 1

julia> N0 = (Qr*NBODr + Q1*NBOD1) / (Qr+Q1); #mg/L, NBOD conc after waste
stream 1

julia> # Function Definition
       function DO_conc(Cs, C0, B0, N0, ka, kc, kn, x, U)
         a1 = exp(-ka*x/U)
         a2 = (kc/(ka-kc))*(exp(-kc*x/U)-exp(-ka*x/U))
         a3 = (kn/(ka-kn))*(exp(-kn*x/U)-exp(-ka*x/U))
         C = Cs*(1-a1)+C0*a1-B0*a2-N0*a3
         return C
       end
DO_conc (generic function with 1 method)

julia> # Waste Stream 1: 0-14 km
       for i in 1:16
         DO[i] = DO_conc(Cs, C0, B0, N0, ka, kc, kn, x[i], U);
         CBOD[i] = B0*exp(-kc*x[i]/U);
         NBOD[i] = N0*exp(-kn*x[i]/U);
       end

julia> # Waste Stream 2: 15-50 km
       C0_new = (DO[16]*(Qr+Q1) + DO2*Q2) / (Qr+Q1+Q2);

julia> B0_new = (CBOD[16]*(Qr+Q1) + CBOD2*Q2) / (Qr+Q1+Q2);

julia> N0_new = (NBOD[16]*(Qr+Q1) + NBOD2*Q2) / (Qr+Q1+Q2);

julia> for i in 16:51
         DO[i] = DO_conc(Cs, C0_new, B0_new, N0_new, ka, kc, kn, x[i]-15, U)
         CBOD[i] = B0*exp(-kc*(x[i]-15)/U);
         NBOD[i] = N0*exp(-kn*(x[i]-15)/U);
       end

julia> # Plotting Dissolved Oxygen
       using Plots

julia> plot(x,DO, title="Dissolved Oxygen Along Stream", label="Dissolved
Oxygen", xlabel="Distance (km)", ylabel="Concentration (mg/L)")
```



Problem 1.2

Based on the plot from Problem 1.1, we can guess that the dissolved oxygen concentration of the river recovers to 6 mg/L somewhere in between 40 and 45 km. The following Julia code employs the bisection method for root finding and employs this method to determine the requested distance from waste stream 2.

```
julia> #Setting up Bisection Method
       val = 6; #mg/L

julia> tol = 0.001; #mg/L

julia> L = 40; #km

julia> R = 45; #km

julia> #Employing Bisection Method
       while R - L > tol
         mid = (R+L)/2
         DO_mid = DO_conc(Cs, Co_new, Bo_new, No_new, ka, kc, kn, mid-15, U);
         if DO_mid == val
           break
         elseif DO_mid < val
           global L = mid
         else
           global R = mid
         end
       end

julia> #Calculating Recovery Distance
```

```
       recovery_distance = (R+L)/2 - 15;

julia> print(recovery_distance)
26.16241455078125
```

The output of this code says that the dissolved oxygen concentration recovers to 6 mg/L at approximately 41.16 km, or 26.16 km from Waste Stream 2.

Problem 1.3

We can test several values of organic waste removal treatment, starting from 0% and increasing by 1%. With each removal, we will alter the Waste Stream 2 values accordingly and perform the calculations from Problem 1.1 again. If the minimum of the DO levels after the entrance to Waste Stream 2 is lower than 4, we increase the level of treatment.

```
julia> DO_removal = DO[1:51]; #initializing DO list with 0% removal values

julia> treatment = 0;

julia> while minimum(DO_removal) < 4
         global treatment = treatment + 0.01;

         # New Waste Stream 2 content
         CBOD2_treated = (1-treatment) * CBOD2;
         NBOD2_treated = (1-treatment) * NBOD2;

         # New river content post Waste Stream 2;
         B0_treated = (CBOD[16]*(Qr+Q1) + CBOD2_treated*Q2) / (Qr+Q1+Q2);
         N0_treated = (NBOD[16]*(Qr+Q1) + NBOD2_treated*Q2) / (Qr+Q1+Q2);

         # Recalculating DO after combination of river and Waste Stream 2
         for i in 16:51
           DO_removal[i] = DO_conc(Cs, C0, B0_treated, N0_treated, ka, kc, kn,
x[i]-15, U)
         end
       end

julia> print(treatment)
0.5100000000000002
```

Based on the above code, the treatment level of Waste Stream 2 that will keep the DO concentrations of the stream above 4 mg/L is 51% (to the closest 1%).

Problem 1.4

We will employ the same method as Problem 1.3, but alter the values from Waste Stream 1 AND Waste Stream 2 using our tested treatments. After a couple of test cases, it is noted that the level of treatment is much lower than in Problem 1.3, so we increase our treatment tests in increments of 0.1% instead of 1%.

```
julia> DO_treated = DO[1:51]; #initializing DO list with 0% removal values
julia> treatment2 = 0;

julia> while minimum(DO_treated) < 4
         global treatment2 = treatment2 + 0.001;

         # New Waste Stream 1 content
         CBOD1_treated = (1-treatment2) * CBOD1;
         NBOD1_treated = (1-treatment2) * NBOD1;

         # New river content post Waste Stream 1
         B01_treated = (Qr*CBODr + Q1*CBOD1_treated) / (Qr+Q1);
         N01_treated = (Qr*NBODr + Q1*NBOD1_treated) / (Qr+Q1);

         # Waste Stream 1: 0-14 km
         for i in 1:16
           DO_treated[i] = DO_conc(Cs, C0, B01_treated, N01_treated, ka, kc,
kn, x[i], U);
         end

         # New Waste Stream 2 content
         CBOD2_treated = (1-treatment2) * CBOD2;
         NBOD2_treated = (1-treatment2) * NBOD2;

         # New river content post Waste Stream 2;
         CBOD_15t  = B01_treated*exp(-kc*15/U);
         NBOD_15t  = N01_treated*exp(-kn*15/U);
         C02_treated = (DO_treated[16]*(Qr+Q1) + DO2*Q2) / (Qr+Q1+Q2);
         B02_treated = (CBOD_15t*(Qr+Q1) + CBOD2_treated*Q2) / (Qr+Q1+Q2);
         N02_treated = (NBOD_15t*(Qr+Q1) + NBOD2_treated*Q2) / (Qr+Q1+Q2);

         # Waste Stream 2: 15-50 km
         for i in 16:51
           DO_treated[i] = DO_conc(Cs, C02_treated, B02_treated, N02_treated,
ka, kc, kn, x[i]-15, U)
         end
       end

julia> print(treatment2)
0.06600000000000004
```

Based on the above code, the minimum level of treatment to both streams is 6.6%.

Problem 1.5

From Problems 1.3 and 1.4, treating Waste Stream 2 solely would require a 51% organics removal, whereas treating both streams simultaneously would require 6.6% organics removal in both streams. Based on these numbers alone and assuming there is a linear relationship between cost and the % removal, I would opt to treat both streams equally. This is equivalent to paying 13.2 units of cost (6.6 for each stream) as opposed to 51 units of cost.

However, this assumption of a linear relationship between cost and removal is not necessarily accurate. Installing a treatment system in two facilities as opposed to one could prove to be significantly more expensive just in installation/capital costs. Furthermore, there is most likely a diminishing marginal cost for every additional percentage of removal, which means that removing the 6th percent in the double-treatment approach could be significantly more expensive than removing the 50th percent in the single-treatment approach. Without knowing this information, it is difficult to make a definitive conclusion.

Problem 1.6

The strategy chosen in Problem 1.5 is the 6.6% removal from both streams. Here we will redo Problem 1.1 but incorporate this treatment and test over the given range of CBOD (4-7 mg/L) and NBOD (3-8 mg/L) to determine probability.

```
julia> fails = 0;

julia> DO_probability = zeros(length(x));

julia> # Note: flow rates, DO concs, decay rates, velocity, sat oxygen are same

       # Treated values, subscript 6 to represent this problem
       treatment = 0.066;

julia> CBOD1_6 = (1-treatment)*CBOD1;

julia> CBOD2_6 = (1-treatment)*CBOD2;

julia> NBOD1_6 = (1-treatment)*NBOD1;

julia> NBOD2_6 = (1-treatment)*NBOD2;

julia> # River Ranges
       CBOD_range = 4:0.1:7;

julia> NBOD_range = 3:0.1:8;

julia> for i in 1:length(CBOD_range)
         for j in 1:length(NBOD_range)
           CBOD_6 = CBOD_range[i];
           NBOD_6 = NBOD_range[j];
           DO_6 = zeros(length(x));

           C01_6 = (Qr*DOr + Q1*DO1) / (Qr+Q1);
           B01_6 = (Qr*CBOD_6 + Q1*CBOD1_6) / (Qr+Q1);
           N01_6 = (Qr*NBOD_6 + Q1*NBOD1_6) / (Qr+Q1);

           # Waste Stream 1: 0-14 km
           for i in 1:16
             DO_probability[i] = DO_conc(Cs, C01_6, B01_6, N01_6, ka, kc, kn,
x[i], U);
           end

           # Waste Stream 2: 15-50 km
           CBOD2_river = B01_6*exp(-kc*15/U);
           NBOD2_river = N01_6*exp(-kn*15/U);
           C02_6 = (DO_probability[16]*(Qr+Q1) + DO2*Q2) / (Qr+Q1+Q2);
           B02_6 = (CBOD2_river*(Qr+Q1) + CBOD2_6*Q2) / (Qr+Q1+Q2);
           N02_6 = (NBOD2_river*(Qr+Q1) + NBOD2_6*Q2) / (Qr+Q1+Q2);
```

```
          for i in 16:51
            DO_probability[i] = DO_conc(Cs, Co2_6, Bo2_6, No2_6, ka, kc, kn,
x[i], U);
          end

          if minimum(DO_probability) < 4
            global fails = fails + 1;
          end
        end
      end

julia> probability = fails/(length(CBOD_range)*length(NBOD_range));

julia> print(probability);
0.05502846299810247
```

As shown the results of the above code, the probability of the strategy identified in Problem 1.5 failing is approximately 5.5%.

Problem 1.7

Now suppose that the CBOD and NBOD values are correlated: specifically, they have a correlation coefficient of 0.7. How does that impact the probability of maintain- ing a dissolved oxygen concentration above 4 mg/L under the treatment strategy from Problems 1.5? The solution template includes a code block defining a function sam-ple*correlated*uniform() that you can use to produce correlated uniform variates.

Problem 1.8

Discuss how the presence of uncertainty and dependence between variables might affect your decision-making process. Would you stick with the same strategy as in Problem 1.5, or find a new strategy? What other information would you need, if any?

References

1. Bisection Method in Julia: https://mmas.github.io/bisection-method-julia