Muhlenberg Water Optimization

David Abrahams

September 2015

1 Results

To whom it may concern,

I have created a framing for Muhlenberg's water supply, and have obtained the cheapest possible solution that meets all necessary safety requirements.

We sought out to obtain the cheapest solution that did not violate any of the following conditions:

- Each water supply can only produce a certain amount of water. See Appendix B.
- Each city has a water need we must meet. See Appendix C.
- No town's average water hardness is greater than $1200\text{kg}\,\text{Ml}^{-1}$. See Appendix D for each supply's hardness.

The cost of transporting water from one water source to a town is defined in Appendix A

A Water transportation costs

The cost of transporting water from each city to each town is defined in the following table, where cost is in $\$\text{Ml}^{-1}$.

	Peaceful Waters	Prinetown	Twin Lakes	Muhlenberg	Johnsville
Lake Elizabeth	450	460	440	445	455
Lake Marie	495	500	505	510	490
Paradise Aquifer	900	915	885	920	920
Green River	1800	1815	1795	1785	1820

B Maximum water supply

Each source is able to produce the following amount of water, in $\text{Ml}\,\text{d}^{-1}$.

Lake Elizabeth	Lake Marie	Paradise Aquifer	Green River
15	10	60	80

C Water needs

Each city requires the following amount of water, in $\text{Ml}\,\text{d}^{-1}$.

Peaceful Waters	Prinetown	Twin Lakes	Muhlenberg	Johnsville
30	10	50	20	40

D Maximum water supply

Each source's water has the following hardness, in $\text{kg}\,\text{Ml}^{-1}$.

Lake Elizabeth	Lake Marie	Paradise Aquifer	Green River
250	200	2300	700

E The source code

```matlab
% This is our objective function. Each row is a source, each column a town.
% Covert the objective function to a column vector.
Objective = [450, 460, 440, 445, 455;
             495, 500, 505, 510, 490;
             900, 915, 885, 920, 920;
             1800, 1815, 1795, 1785, 1820];
Objective = Objective';
Objective = Objective(:)';

% The maximum amount of water each source can supply
Supply_Maxes = [15; 10; 60; 80];

% This matrix, when multiplied by a column vector containing the amount of
% water each source supplies to each town, returns a column vector for how
% much water each supply gave off.
Total_Supply_Usage = zeros(4, 20);
for i=1:4

    Total_Supply_Usage(i, (5*i - 4) : (5*i)) = 1;

end

% The amount of water each town needs
City_Needs = [30; 10; 50; 20; 40];

% This matrix, when multiplied by the column vector, gives the total amount
% of water each town recieved.
Total_City_Water = zeros(5, 20);
for i=1:5
    Total_City_Water(i, i:5:20) = 1;
end

% This is the total amount of hardness each city can have. It is the
% hardness per Ml times the amount of water each city needs, in Ml
Hardness_Maxes = [1200; 1200; 1200; 1200; 1200] .* City_Needs;


% This matrix, when multiplied by the column vector, gives the total amount
% of hardness each town recieved.
Supply_Hardness = [250, 200, 2300, 700];
Hardness_Matrix = zeros(5, 20);
for i=1:5
    Hardness_Matrix(i, i:5:20) = Supply_Hardness;
end

% A vertical combination of Water Use and Hardness, both <= constraints
Usage_and_Hardness = vertcat(Total_Supply_Usage, Hardness_Matrix);
Supply_Hard_Max = vertcat(Supply_Maxes, Hardness_Maxes);

% Each city has to give at least 0 water.
for i=1:20
    minimum(i, 1) = 0;
end

% Get the optimal solution
x = linprog(Objective, Usage_and_Hardness, Supply_Hard_Max,...
    Total_City_Water, City_Needs, minimum, []);
% Convert to a 4x5 matrix and print it
for i=1:4
    result(i, 1:5) = x(5*i - 4:5*i, 1);
end
disp(result);
```