STEM Games 2019

Mathematics

LET THERE BE LIGHT

a problem by

Petar Bakić, Maja Jurišić Bellotti, Mateja Đumić, Petra Gospodnetić,
Stjepan Šebek, Josip Žubrinić

There is a perfect solution for everything. There must be. The optimum surely exists, but it is a challenge to find it. This year, no small feat stands in front of us—using our mathematical knowledge, skills and tools to rebuild Earth's lungs and minimize our footprint. Assessing both discrete and continuous, playing with endless possibilities, we will only have three days to define the overall cost of our existence and make it as small as possible...

Day 1 – The bare minimum

For too long have the people of Earth ignored the grim predictions and warnings of climate activists and scientists. The damage is irreversible: after years of neglect, the Earth's atmosphere has finally been destabilized to the point where, ironically, global warming is not our problem anymore. Instead, the sun's rays are unable to penetrate the cloud of dust and heavy gasses which shrouds the Earth in near-total darkness.

The newly-formed International Space and Climate Regulation Agency (ISCRA) has been appointed by the United Nations to come up with a temporary solution to this problem. The plan is simple: using an ancient technology (something called GPS—admittedly, the NASA of the 1970's was naive, but they had some good ideas), a number of artificial light sources will be placed at fixed locations relative to the Earth's surface, providing us with the radiation we so desperately need.

As ISCRA's principal research team, you are assigned with positioning the light sources at their optimal locations. Time is running out and the margin of error is minimal: the project must be tested extensively before the light sources are launched. Of course, in order for you to develop great ideas, the problem has to be simplified first - to the bare minimum.

To test your ideas, you will be working on a 2D model of Earth, reconstructed from a set of several key locations on its surface. These locations describe the Earth as a convex polygon, an object well documented in ISCRA's files. The light sources will be approximated by points in the plane surrounding the polygon. The head of the department has provided you with a meager amount of information he thinks could be of interest to you. His memo is literally the following:

"Look, we don't have much funds for this feat. Whichever problem you tackle, I want it to be solved in an optimal way. If anything, use some methods of mathematical optimization, those never get old. I am not here to lecture you, just to give you several tips. Remember that the optimization problem comes down to finding the vector x which solves the following problem:

$$\begin{aligned} &\max f(x) \\ &\textit{subject to } x \in \Omega, \end{aligned}$$

where Ω is some set of feasible solutions. Actually, you would probably want to refresh your knowledge on the so called linear integer programming problems:

$$\begin{array}{ll} \max & c^T x \\ \textit{subject to} & Ax \leq b \\ & x \geq 0,\ x_i \textit{ is an integer } \forall i. \end{array}$$

I am sure that you know that you can solve these by using some old fashioned MIP solvers. I hope you are familiar with the concept of polygon, the convex hull of finitely many points. Of course, I probably don't have to tell you that polygons can be represented with the matrix inequality

$$Ax \leq b,$$

since obviously this defines an intersection of finitely many halfplanes. As far as I'm concerned, you might as well look up some of the typical linear integer programming problems such as set covering or set partitioning, but what do I care. You have three days to come up with something."

The team has singled out several problems which have to be solved as soon as possible. Each problem comes with its set of data from which you have to calculate the solution. You should come up with the model for the described situation and, if analyzed carefully, your model could lead to a solution which saves our planet from slowly withering and offers us a spark of hope. Good luck!

Problems

Problem 1

In the first task, the input contains coordinates of N points in the plane. The polygon we wish to illuminate is given as a convex hull of these points. Furthermore, ISCRA has already extracted from GPS archives the M possible positions that the light sources can occupy. Of course, it would be superfluous to use each of the M positions—due to limited funds, ISCRA urges us to use the minimal number of sources, while still illuminating the whole polygon. Therefore, the problem is the following: using as few sources as possible, illuminate every point on the perimeter of the polygon. Each source can only be placed at one of the prescribed positions.

We consider a point P to be illuminated if there exists a light source S such that a straight line can be drawn from P to S without intersecting the polygon at any point different from P.

Problem 2

After successfully dealing with Problem 1, ISCRA has noticed that merely illuminating every point on the perimeter is not enough. Social and ecological demands require a certain level of brightness to be available at every point of the Earth's surface. The following variation of Problem 1 is proposed to take care of this new requirement: using as few sources as possible, make sure that the mean illumination of each side of the polygon equals at least the required threshold. Again, each source can only be placed at one of the prescribed positions.

The mean illumination of a side $\overline{AB}$ of the polygon is defined by the following formula

$$\overline{L_{AB}} = \frac{1}{|AB|} \cdot \sum_{S} \int_{AB} \overrightarrow{F_S} \cdot \overrightarrow{n}\, ds,$$

where the summation is taken over all the light sources S which illuminate $\overline{AB}$,

$$\overrightarrow{F_S}(X) := \frac{1}{1 + ||\overrightarrow{SX}||^2} \cdot \frac{\overrightarrow{SX}}{||\overrightarrow{SX}||}$$

for any point X in the plane, and

$$\overrightarrow{n}(X) = \text{ inward unit normal to the side } \overline{AB} \text{ at point } X.$$

The required minimal threshold for the illumination of each side is 1.

Instructions

Each team is provided with the folder `Day_1`. The folder contains the following:

- Folder `cases` with the test datasets in .csv format
- Folder `solution` where the solution should be implemented
- MATLAB function

  ```
  [data] = loadData (casename)
  ```

  which loads the data from `casename.csv` into `struct data`.

Data

The input consists of one `struct` called `data`. This struct contains the following:

- `data.N` – a positive integer [the number of points which determine the polygon]
- `data.M` – a positive integer [the number of possible light source positions]
- `data.t` – a vector of length `data.N` containing pairs of coordinates
- `data.r` – a vector of length `data.M` containing pairs of coordinates
- `data.problemtype` – a string which specifies the type of the problem (‘`problem1`’/ ‘`problem2`’)

Solution

Each team is required to implement the following functions:

- `[chosen_positions] = optimize (data)`
- `visualize (data, chosen_positions)`

The function `optimize` should contain all the code used to arrive at the solution. Depending on the `data.problemtype` it should solve the corresponding problem. Any auxiliary functions should also be provided. The output is a vector

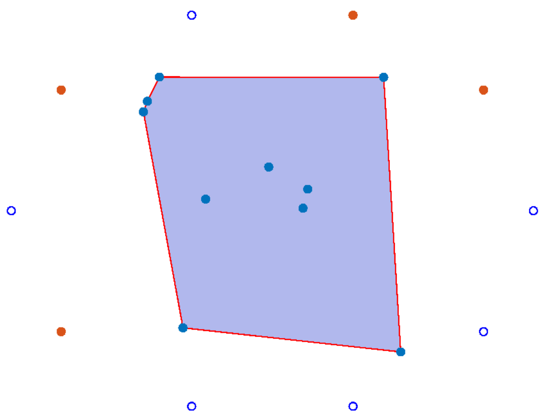
- `chosen_positions`

containing pairs of coordinates of the chosen light source positions.

The function `visualize` serves to graphically represent the input and the output. In particular, it should create a visualization which clearly shows the initial data (i.e. the points of `data.t`, the corresponding convex hull, and the positions given by `data.r`), as well as the points of `chosen_positions`.

Example

Each team is provided with several case files: `problem_1_case_1.csv`, `problem_1_case_2.csv`, etc. A possible solution to Problem 1 with the initial data provided by `problem_1_case_1.csv` is depicted by the following graphic:



- points given by `data.t` which determine the polygon
- unused source positions
- used source positions

In this example, four sources—but no less—are sufficient to illuminate the entire polygon.

NOTE:
— All the functions should be implemented from scratch except for the optimization solvers.
— All solutions need to be documented in a .txt or .md file, explaining all the underlying ideas and thought processes.

Day 2 – Global warning

The UN has issued a global warning. There is no higher priority than your project, right? Right? Well. . . you didn't really think that you would have unlimited funding?

"A whole day wasted! As it turns out, our little plan of fixing this whole light issue has come to a certain financial obstruction from the guys above. If they had insured that annoying dome project last year, we wouldn't have this problem now. They were running by the good old #ItWontHappenToMe idea...well guess what - the thing broke down and the fortune went down the drain. Foolishness! But that's just me. Now they are putting even more money to launching the asteroid defense system. It is obvious that even the tenth of the laser turrets would be enough to end the threat, but somebody has interest in building those giant things.
Anyway, the situation is clear, we can't afford enough light sources for the whole surface, so you will have to work with what you got. Since these sources are all we have, it would be a good idea not to waste them by putting them in a nonoptimal way. Make sure that this time it counts, and shine on as much of a surface as your can. Concentrate, don't waste your, and even more important, my time. "

Today is not your day. The hours you spent working on your last model were all in vain. Instead of bathing the whole surface of the Earth in light, you will have to make do with a limited number of light sources. To see what can be achieved, you will need to redesign your approach. Again, you will have to solve two similar problems.

Problem 1

Today, the polygon we wish to illuminate is given by a set of N inequalities

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 &\leq b_1 \\ a_{21}x_1 + a_{22}x_2 &\leq b_2 \\ &\cdots \\ a_{N1}x_1 + a_{N2}x_2 &\leq b_N. \end{aligned} \tag{$*$}$$

This can be written in matrix form as $Ax \leq b$ where $x = (x_1, x_2)^t$, A is a $N \times 2$ matrix, and $b = (b_1, \ldots, b_N)^t$. You are still able to use the M positions prescribed by ISCRA, only this time, the number of sources you may use is limited. Therefore, your goal is to illuminate as much of the polygon's perimeter as possible, using only the allowed number of sources.

Instructions

Each team is provided with the folder `Day2` containing folder `Problem_1`. The folder `Problem_1` contains the following:

- Folder `cases` with the test datasets in .csv format
- Folder `solution` where the solution should be implemented
- MATLAB function

  ```
  [data] = loadData (casename)
  ```

  which loads the data from `casename.csv` into `struct data`.

Data

The input consists of one `struct` called `data`. This `struct` contains the following:

- `data.nRows` – a positive integer [the number of inequalities in $(*)$]
- `data.A` – the matrix A of size `data.nRows` $\times 2$ from $(*)$
- `data.b` – the vector b from $(*)$
- `data.M` – a positive integer [the number of possible light source positions]
- `data.r` – a vector of length `data.M` containing pairs of coordinates
- `data.maxNumber` – a positive integer specifying the maximum number of allowed sources

Solution

Each team is required to implement the following functions:

- `[chosen_positions] = optimize_problem_1 (data)`
- `visualize_problem_1 (data, chosen_positions)`

The function `optimize_problem_1` should contain all the code used to arrive at the solution. Any auxiliary functions should also be provided. The output is a vector

- `chosen_positions`

containing pairs of coordinates of the chosen light source positions.

The function `visualize_problem_1` serves to graphically represent the input and the output. In particular, it should create a visualization which clearly shows the initial data (i.e. the polygon and the positions given by `data.r`), as well as the points of `chosen_positions`.

Problem 2

"Guys...we don't have light source positions.
Head of the department"

To make things worse, it turned out that the positions provided by the GPS archives are not as accurate as you have hoped. In short, you have to recalculate them yourself. Thus, your second assignment today is the same as Problem 1, only now, there are no prescribed positions. Instead, you may place your sources anywhere inside a restricted area which surrounds the polygon. Since the light sources have to be able to communicate with the stations on the Earth, they have to be placed close enough to the Earth's surface. It turns out that this condition restricts the area on which the light sources can be placed to the polygon given by a slightly modified set of inequalities:

$$Ax \leq \eta \cdot b.$$

Here η represents relative range of the communicators, and is measured to be $\eta = 1.3$.

Instructions

Folder `Day_2` contains also folder `Problem_2` with the following content:

- Folder `cases` with the test data sets in .csv format
- Folder `solution` where the solution should be implemented
- MATLAB function

  `[data] = loadData (casename)`

  which loads the data from `casename.csv` into `struct data`.

Data

The input consists of one `struct` called `data`. This `struct` contains the following:

- `data.nRows` – a positive integer [the number of inequalities in $(*)$]
- `data.A` – the matrix A of size nRows $\times$ 2 from $(*)$
- `data.b` – the vector b from $(*)$
- `data.maxNumber` – a positive integer specifying the maximum number of allowed sources

Solution

The guidelines of Problem 1 also apply to Problem 2. The solution should contain the functions

- `[chosen_positions] = optimize_problem_2 (data)`

- `visualize_problem_2 (data, chosen_positions)`

with the vector

- `chosen_positions`

containing pairs of coordinates of the chosen light source positions.

NOTE:
— All solutions need to be documented in a .txt or .md file, explaining all the underlying ideas and thought processes.

Day 3 – There is a light that never goes out

The time has come to launch the light sources to their predestined locations. Unfortunately, the suspicions were true, the GPS archives are proven to be completely unreliable. ISCRA's teams must completely drop all of the suggestions and calculate the optimal positions themselves. The translation from discrete to continuous has left some of the researchers worried. Even the head of the department has given up on his usual grin and realized the seriousness of the situation.

"Researchers, it is all up to you now. I wanted to make sure that your spirit is up to the task. So far it has been easy, mining through the prescribed locations but we have lost that advantage. With the drastically limited funds and continuous space of solutions, I can only give you several advices.
Stay out of local optima, I am sure that there will be many. Also, make sure that you have some algorithm for nonlinear optimization up your sleeve, you might need it. But most importantly, whatever you do, do not fail! That is not an option!"

With the final remark from the head of the department, ISCRA's researchers have singled out two problems which have to be solved before launching the light sources. With no constraint on their final destination, make sure that they serve their purpose—to provide the light that never goes out.

Problems

Problem 1

Once again, the polygon which represents the Earth is given as a convex hull of points provided in data files. The Problem 1 comes down to placing only one light source in an area surrounding the polygon. This light source must be placed in the position where the total illumination of the polygon will be at its highest. The total illumination by the light source S is the sum of illuminations of each side, namely:

$$L(S) = \sum_{\overline{AB}} L_{AB}(S),$$

where the summation is taken over each side $\overline{AB}$ of the polygon illuminated by the light source S. The illumination of the side is given with the formula:

$$L_{AB}(S) = \int_{\overline{AB}} \overrightarrow{F_S} \cdot \overrightarrow{n}\, ds,$$

where

$$\overrightarrow{F_S}(X) := \frac{1}{1 + ||\overrightarrow{SX}||^2} \cdot \frac{\overrightarrow{SX}}{||\overrightarrow{SX}||},$$

$$\overrightarrow{n}(X) = \text{ inward unit normal to the side } \overline{AB} \text{ at point } X.$$

Therefore, your goal is to find the location S for the light source where the total illumination of the polygon is as high as possible. The location S may be any point outside the polygon.

Problem 2

The Problem 2 is the culmination of all of your efforts so far. ISCRA has calculated the budget for the light sources and came up with the exact number of sources which will be deployed. Your task is to calculate all the positions for the prescribed number of light sources. The final configuration has to be placed in such way that it maximizes the illumination of the least illuminated side of the polygon. In other words, find the configuration of positions of light sources, such that the minimal illumination over sides of the polygon is maximal.

"Look, I don't trust any of the nonlinear optimization solvers out there. I want you to write your own version. Don't argue with me, but get to work.

Head of the department"

In order to complete the tasks in Problems 1 and 2 successfully you will have to implement the solution algorithms from the scratch. This means that it is not possible to get maximal number of points by using premade solvers and, after all, we want to maximize things here, right?

Instructions

Each team is provided with the folder `Day_3`. The folder contains two folders: `Problem_1` and `Problem_2`. These two folders both contain the following:

- Folder `cases` with the test datasets in .csv format
- Folder `solution` where the solution should be implemented
- MATLAB function

  ```
  [data] = loadData (casename)
  ```

  which loads the data from `casename.csv` into `struct data`.

Data

The input consists of one `struct` called `data`. This `struct` contains the following:

- `data.N` – a positive integer [the number of points which determine the polygon]
- `data.nrOfSources` – a positive integer [the number of light sources] — for Problem 1 `nrOfSources` is always equal to 1.

- `data.t` – a vector of length `data.N` containing pairs of coordinates

Solution

Each team is required to implement the following functions:

- `[opt_pos] = optimize_problem_1 (data)`
- `[opt_pos] = optimize_problem_2 (data)`
- `visualize_problem_1 (data, opt_pos)`
- `visualize_problem_2 (data, opt_pos)`

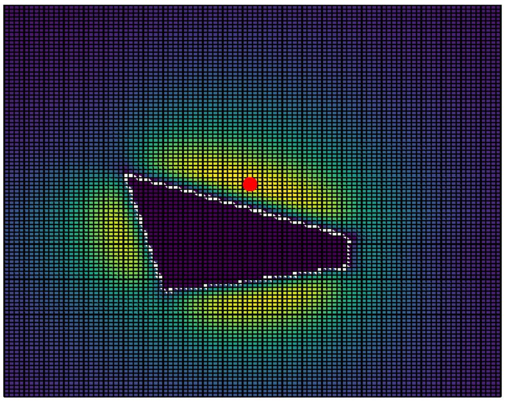
Functions `optimize_problem_1` and `optimize_problem_1` should contain all the code used to arrive at the solution. Any auxiliary functions should also be provided. The output consists of a vector

- `opt_pos`

containing pairs of coordinates of the chosen light source positions. In the case of Problem 1, `opt_pos` only contains one position.
Functions `visualize_problem_1` and `visualize_problem_2` serve to graphically represent the input and the output. In particular, they should create a visualization which clearly shows the initial data (i.e. the points of `data.t`, the corresponding convex hull, as well as the points of `opt_pos`.

Example

Solution to Problem 1 with initial data provided by `problem_1_case_2` is depicted by the following graphic:



The light of the color represents the quality of illumination.

NOTE:
— All solutions need to be documented in a .txt or .md file, explaining all the underlying ideas and thought processes.