Metody optymalizacji - Lista 1

Bartosz Rzepkowski

16 kwietnia 2017

1 Zadanie 1

1.1 Opis modelu

Celem zadania jest zminimalizowanie funkcji celu

$$c^T x$$

przy warunkach

$$Ax = b, x \geqslant 0,$$

gdzie

- x - wektor będący rozwiązaniem zadania (długości n),
- A - macierz Hilberta (o wymiarach n x n),
- b - wektor prawych stron (długości n),
- c - taki sam wektor, jak wektor b (stosowany w funkcji celu).

1.2 Stworzone funkcje

W celu rozwiązania zadania utworzono trzy funkcje:

- generujDane(n) - tworzy macierz Hilberta oraz wektory b i c,
- hilbertGLPK(A, b, c, n) - rozwiązuje zadanie za pomocą GLPKSolverLP,
- hilbertClp(A, b, c, n) - rozwiązuje zadanie za pomocą ClpSolver.

1.3 Wyniki

Dla danych n otrzymano następujące błędy względne (między rozwiązaniem obliczonym a dokładnym):

GLPKSolverLP:

n	$\|\|x - \hat{x}\|\|_2 / \|\|x\|\|_2$
4	3.494046307678745e-13
5	4.7929164423792106e-12
6	2.9850004756375035e-10
7	0.5534761248132727
8	0.7647088813981485
9	0.8188744441556137

ClpSolver:

n	$\|\|x - \hat{x}\|\|_2 / \|\|x\|\|_2$
4	4.743371841522313e-14
5	9.65134105129642e-12
6	2.883594385564789e-10
7	0.6356338710471922
8	0.5148382006675944
9	0.9737490339883091

1.3.1 Przykładowe wyniki

W poniższych tabelach przedstawiono przykładowe obliczone rozwiązania:

GLPKSolverLP:

n	$\hat{x}$
5	[1.0,1.0,1.0,1.0,1.0]
6	[1.0,1.0,1.0,1.0,1.0]
7	[0.999636,1.01455,0.85974,1.54545,0.0,1.864,0.716364]

ClpSolver:

n	$\hat{x}$
5	[1.0,1.0,1.0,1.0,1.0]
6	[1.0,1.0,1.0,1.0,1.0]
7	[1.00039,0.984127,1.15625,0.382716,2.14583,0.0,1.33102]

1.4 Wnioski

Możemy zauważyć, że błąd względny między wynikiem obliczonym a dokładnym (równym [1, 1, ...]) bardzo szybko rośnie (wraz ze wzrostem n). Ponadto obliczany $\hat{x}$ jest dokładny jedynie do $n = 6$. Jest to spowodowane tym, że macierz Hilberta jest przykładem macierzy źle uwarunkowanej.

2 Zadanie 2

2.1 Opis modelu

W modelu służącym do rozwiązania zadania będziemy korzystać z następujących tablic:

- odleglosci - tablica o wymiarach $7x7$, w której przechowywane są odległości między poszczególnymi miastami,
- dzwigi_nadmiar - tablica o wymiarach $7x2$, w której przechowywane są informacje o nadmiernej ilości dźwigów w poszczególnych miastach (w danym wierszu w kolumnie 1 jest informacja o dźwigach typu I, a w kolumnie 2 o dźwigach typu II),
- dzwigi_niedobor - tablica o wymiarach $7x2$, w której przechowywane są informacje o niedoborach dźwigów w poszczególnych miastach (działa analogicznie do tabeli dzwigi_nadmiar),

Rozwiązaniem modelu będzie tablica *transport* rozmiaru $7x7x2$. W komórkach *transport[_, _, 1]* są przechowywane informacje o transporcie dźwigu typu I, a w *transport[_, _, 2]* o dźwigach typu II. Wartości w wierszach tabeli odpowiadają dźwigom wysłanym z tego miasta do innego, natomiast w kolumnach dźwigom otrzymanym z innych miast.

2.1.1 Przykład tabeli transport

Poniżej zamieszczono przykład tabeli *transport*:

0.0, 4.0, 0.0, 0.0, 5.0, 0.0, 0.0
3.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
0.0, 5.0, 0.0, 1.0, 0.0, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 5.0, 0.0
2.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0

0.0, 0.0, 3.0, 0.0, 0.0, 5.0, 0.0
0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0
1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
0.0, 0.0, 0.0, 3.0, 4.0, 2.0, 1.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0

Możemy z niej odczytać, że z pierwszego miasta wysłano w sumie 9 dźwigów typu I (do miast 2 i 5) i 8 dźwigów typu II (do miast 3 i 6). Dotarło do niego również 5 dźwigów typu I (z miast 2 i 6) oraz 1 dźwig typu II (z miasta 3).

2.1.2 Funkcja celu

W modelu będziemy minimalizować następującą funkcję celu:

$$\sum_{i=1}^{n}\sum_{j=1}^{n}(transport[i,j,1]odleglosci[i,j]) + 1.2 * \sum_{i=1}^{n}\sum_{j=1}^{n}(transport[i,j,2]odleglosci[i,j]).$$

Mówi ona o tym, że koszt transportu dźwigu typu I między danymi miastami jest proporcjonalny do odległości między nimi, natomiast koszt transportu dźwigu typu II jest o 20% większy.

2.1.3 Ograniczenia

W modelu zastosowano następujące ograniczenia:

1. $transport[i,j,k] \geqslant 0$, dla $i,j = 1,2,...,n, k = 1,2$,
2. $\sum_{j=1}^{n} transport[i,j,k] = dzwigi\_nadmiar[i,k]$, dla $i = 1,2,...,n, k = 1,2$,
3. $\sum_{j=1}^{n} transport[j,i,2] \geqslant dzwigi\_niedobor[i,2]$, dla $i = 1,2,...,n$,
4. $\sum_{j=1}^{n} transport[j,i,2] + \sum_{j=1}^{n} transport[j,i,2] = dzwigi\_niedobor[i,1] + dzwigi\_niedobor[i,2]$, dla $i = 1,2,...,n$.

Ograniczenie 1 gwarantuje nieujemną liczbę transportowanych dźwigów. Ograniczenie 2 gwarantuje jednakową liczbę dźwigów wysyłanych do innych miast z ich nadmiarową liczbą w danym mieście. Ograniczenia 3 i 4 pozwalają na zamianę dźwigów typu I dźwigami typu II.

2.2 Wyniki

Zadanie rozwiązano dla następujących danych:
odleglosci:

$$\begin{array}{c}
[0.0, 42.7, 55.1, 50.5, 34.2, 56.3, 81.1; \\
42.7, 0.0, 53.2, 72.4, 77.0, 98.7, 125.3; \\
55.1, 53.2, 0.0, 28.7, 86.0, 79.0, 89.3; \\
50.5, 72.4, 28.7, 0.0, 62.2, 52.6, 62.4; \\
34.2, 77.0, 86.0, 62.2, 0.0, 22.0, 59.4; \\
56.3, 98.7, 79.0, 52.6, 22.0, 0.0, 35.7; \\
81.1, 125.3, 89.3, 62.4, 59.4, 35.7, 0.0]
\end{array}$$

dzwigi_nadmiar: [7, 0; 0, 1; 6, 2; 0, 10; 5, 0; 0, 0; 0, 0]
dzwigi_niedobor: [0, 2; 10, 0; 0, 0; 4, 0; 0, 4; 8, 2; 0, 1]
Za pomocą GLPKSolverLP otrzymano następującą tabelę *transport*:

$$\begin{array}{c}
0.0, 4.0, 0.0, 0.0, 0.0, 3.0, 0.0 \\
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 \\
0.0, 5.0, 0.0, 1.0, 0.0, 0.0, 0.0 \\
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 \\
0.0, 0.0, 0.0, 0.0, 0.0, 5.0, 0.0 \\
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 \\
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 \\
\\
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 \\
0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0 \\
2.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 \\
0.0, 0.0, 0.0, 3.0, 4.0, 2.0, 1.0 \\
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 \\
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 \\
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
\end{array}$$

Rozwiązanie zadania za pomocą GLPKSolverMIP dało takie samo rozwiązanie.

3 Zadanie 3

3.1 Opis modelu

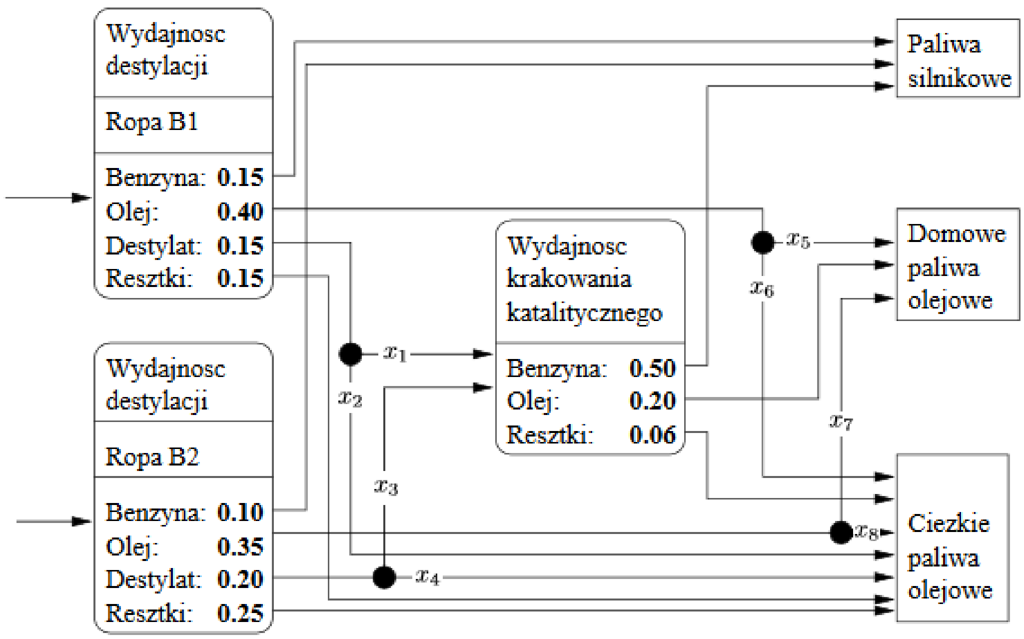
3.1.1 Zmienne

W celu rozwiązania zadania posłużono się następującymi zmiennymi:

- B1 - ropa B1,
- B2 - ropa B2,

- K - ropa poddana krakowaniu katalitycznemu,
- PS - paliwo silnikowe,
- DPO - domowe paliwa olejowe,
- CPO - ciężkie paliwa olejowe.

Ponadto wprowadzono zmienne x_i dla $i = 1, 2, ..., 8$, które zostały zaznaczone na poniższym schemacie:



Rysunek 1: Schemat działania rafinerii z zaznaczonymi dodatkowymi zmiennymi

3.1.2 Funkcja celu

Model będzie minimalizował następującą funkcję celu:

$$1300 \cdot B1 + 10 \cdot B1 + 1500 \cdot B2 + 10 \cdot B2 + 20 \cdot x_1 + 20 \cdot x_3$$

3.1.3 Ograniczenia

W modelu zastosowano następujące ograniczenia:

1. $B1, B2, K, PS, DPO, CPO, x_1, x_2, ..., x_8 \geqslant 0$,
2. $0.15 \cdot B1 + 0.1 \cdot B2 + 0.5 \cdot K = PS$,

3. $x_5 + x_7 + 0.2 \cdot K = DPO$,
4. $x_6 + x_8 + 0.06 \cdot K + x_2 + x_4 + 0.15 \cdot B1 + 0.25 \cdot B2 = CPO$,
5. $x_1 + x_3 = K$,
6. $x_1 + x_2 = 0.15 \cdot B1$,
7. $x_3 + x_4 = 0.2 \cdot B2$,
8. $x_5 + x_6 = 0.4 \cdot B1$,
9. $x_7 + x_8 = 0.35 \cdot B2$,
10. $PS \geqslant 200\,000$,
11. $DPO \geqslant 400\,000$,
12. $CPO \geqslant 250\,000$,
13. $x_5 \cdot 0.002 + x_7 \cdot 0.012 + 0.2 \cdot (x_1 \cdot 0.003 + x_3 \cdot 0.025) \leqslant 0.005 \cdot DPO$.

Ograniczenie 1 gwarantuje nie ujemność używanych zmiennych. Ograniczenia od 2 do 4 ustalają jakie składowe będą wchodziły w ostateczną wartość zmiennych PS, DPO i CPO. Ograniczenie 5 opisuje ilość paliwa, która zostanie poddana krakowaniu katalitycznemu. Ograniczenia od 6 do 9 gwarantują własność, że ilość paliwa wypływającego z każdego rozgałęzienia jest równa ilości paliwa do niego wpływającej. Ograniczenia od 10 do 12 narzucają minimalną ilość ostatecznych paliw, jaką chcemy wyprodukować. Ograniczenie 13 zapewnia właściwą ilość siarki zawartej w wyprodukowanych domowych paliwach olejowych.

3.2 Wyniki

Dla przedstawionego modelu GLPKSolverLP wygenerował następujące rozwiązanie:

- B1 = 1.0260303687635575e6,
- B2 = 0.0,
- PS = 200000.0,
- DPO = 400000.0,
- CPO = 250000.0.