

Politechnika Warszawska

WYDZIAŁ MATEMATYKI  
I NAUK INFORMACYJNYCH



# RAPORT

## Laboratorium 4

*Miłosz Mazur*

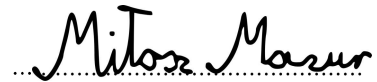
Grupa A

Programowanie Matematyczne

23 listopada 2023

## Oświadczenie

Oświadczam że niniejsza praca stanowiąca podstawę do oceny z przedmiotu Programowanie Matematyczne została przeze mnie wykonana samodzielnie.

A handwritten signature in black ink, reading "Miłosz Mazur". The signature is written in a cursive style with a dotted line underneath it.

Nr indeksu: 298884

## Spis treści

<b>1</b>	<b>Opis problemu</b>	<b>2</b>
<b>2</b>	<b>Opis rozwiązania</b>	<b>2</b>
2.1	Przekształcenie zadanie ZP do ZD	2
2.2	Wybór zmiennych bazowych	3
2.3	Rozwiązanie ZD	3
2.3.1	Rozpoznawanie braku rozwiązania optymalnego	4
2.4	Rozwiązania ZP z rozwiązania ZD	4
<b>3</b>	<b>Przykłady obliczeniowe</b>	<b>4</b>
3.1	Zadanie posiadające rozwiązanie	4
3.2	Zadanie nie posiadające rozwiązania	6
<b>4</b>	<b>Porównanie algorytmów</b>	<b>8</b>
	<b>Bibliografia</b>	<b>8</b>

## 1 Opis problemu

Celem zadanie było rozwiązanie następującego problemu programowania liniowego (oznaczane później jako zadanie prymalne ZP):

$$\min_x c^T x$$
$$\begin{cases} Ax \geq b \\ d \leq x \leq g \end{cases}$$

gdzie  $c \in R^n$ ,  $b \in R^m$ ,  $A \in R^{m \times n}$ .

Do testów należało przyjąć  $m = 10$ ,  $n = 5$  oraz wylosować dowolne dane z określonych przedziałów:

- dla  $c$  oraz  $A$  wartości z przedziału  $[-5, 5]$
- dla  $b$  wartości z przedziału  $[1, 1]$
- dla  $d$  wartości z przedziału  $[-30, -1]$
- dla  $g$  wartości z przedziału  $[1, 30]$

Zadanie należało rozwiązać zarówno wbudowaną funkcją *linprog* [1] jak i za pomocą zbudowania zadania dualnego ZD do zadania pierwotnego ZP własną implementacją algorytmu *simplex*.

## 2 Opis rozwiązania

Przedstawione zostaną kroki konieczne do rozwiązania zadania.

### 2.1 Przekształcenie zadanie ZP do ZD

Pierwszym krokiem będzie zapisanie naszego zadania w ogólnej postaci, włączymy nasze ograniczenia do nierówności w zadaniu:

$$\min_x c^T x$$
$$\begin{cases} Ax \geq b \\ -x \geq -g \\ x \geq d \end{cases}$$
$$x \in R$$

Korzystając z własności zadań dualnych przekształćmy powyższe do ZD.

$$\max_y [b^T, -g^T, d^T] y$$
$$\begin{cases} [A^T, -I, I] y = c \\ y \geq 0 \end{cases}$$

## 2.2 Wybór zmiennych bazowych

Korzystając z własności zadania, że w macierzy  $A_D = [A^T, -I, I]$  znajdują się macierze jednostkowe możemy wybrać zmienne bazowe bez korzystania ze zmiennych sztucznych. Zatem zauważmy że jeśli  $i$ -ty element wektora  $b_D = c$  jest dodatni to odpowiednią dla niego zmienną bazową jest zmienna  $y_{m+n+i}$ . Bowiem w kolumnie  $j = m + n + i$  znajduje się wektor zawierający same zera oraz 1 na  $i$ -tej pozycji.

Natomiast dla przypadku kiedy  $i$ -ty element wektora  $b_D = c$  jest ujemny możemy przekształcić nasze zadanie tak by znalazła się również zmienna bazowa odpowiadające temu wierszowi. Jeśli przemnożymy  $i$ -ty wiersz macierzy  $A_D = [A^T, -I, I]$  oraz  $i$ -ty element wektora  $b_D = c$  przez  $-1$  to otrzymamy zarówno to że teraz  $i$ -ty element wektora  $b_D = c$  jest dodatni oraz możemy zanieść dla niego zmienną bazową. Zauważmy że w macierzy  $A_D = [A^T, -I, I]$  po powyższej operacji w kolumnie  $j = m + i$  znajduje się wektor zawierający same zera oraz 1 na  $i$ -tej pozycji. Zatem zmienna bazowa dla tego elementu to  $y_{m+i}$ .

Przykładową startową macierz po takich rozwiązaniach możemy zaobserwować na rysunku 1

T =

8x23 [table](#)

	xB	cB	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10
fC	NaN	NaN	2.5845	3.063	1.8191	1.8785	2.953	3.9721	2.5414	3.0013	3.9477	2.6625
r1	11	-13.785	1.4527	-4.8422	2.8821	3.9753	4.6125	-0.73419	4.3058	0.4137	-0.49944	1.6904
r2	17	-20.797	0.54295	-4.7979	0.42229	-0.35012	3.3828	0.26586	-4.4879	-0.76956	4.2292	-2.2253
r3	18	-9.4471	-2.3236	4.9516	-3.7474	-0.65711	0.33307	0.26902	4.577	-4.5854	2.6537	-1.0611
r4	19	-29.794	-1.7872	0.68823	0.39823	-4.6819	1.9191	-1.1966	2.107	0.89051	-0.70579	1.5349
r5	15	-7.1142	-2.297	1.5906	-0.17492	-4.3347	1.7751	4.9259	-3.5671	2.5838	3.9065	-0.45614
z	NaN	NaN	60.225	87.932	-23.729	129.02	-206.89	2.6565	-46.654	8.7069	-112.9	-9.4825
z-c	NaN	NaN	57.641	84.869	-25.548	127.14	-209.84	-1.3157	-49.195	5.7056	-116.85	-12.145

(a)

x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	B
-13.785	-4.8362	-19.245	-23.175	-7.1142	-4.459	-20.797	-9.4471	-29.794	-22.342	NaN
1	0	0	0	0	-1	0	0	0	0	3.1724
0	-1	0	0	0	0	1	0	0	0	4.8565
0	0	-1	0	0	0	0	1	0	0	1.7398
0	0	0	-1	0	0	0	0	1	0	4.5367
0	0	0	0	1	0	0	0	0	-1	2.6559
-13.785	20.797	9.4471	29.794	-7.1142	13.785	-20.797	-9.4471	-29.794	7.1142	-315.23
0	25.633	28.692	52.969	0	18.244	0	0	0	29.456	NaN

(b)

Rysunek 1: Przykład macierzy startowej

## 2.3 Rozwiązanie ZD

Rozwiązanie zadania dualnego uzyskujemy poprzez klasyczną implementację algorytmu *simplex* polegającą na zmaksymalizowanie funkcji celu poprzez doprowadzenie by wektor  $z - c$  był dodatni.

### 2.3.1 Rozpoznawanie braku rozwiązania optymalnego

Jeśli podczas któregoś kroku algorytm *simplex* rozwiązujący zadanie dualne nie będzie mógł wykonać kroku, to znaczy dla ujemnego elementu wektora  $z - c$  wszystkie elementy kolumny macierzy  $A_D = [A^T, -I, I]$  będą niedodatnie, oznacza to że zadanie dualne jest nieograniczone, co implikuje również że zadanie primalne nie posiada rozwiązania. Drugi warunek stopu to przekroczenie liczby iteracji przez algorytm (założono 50 możliwych iteracji).

## 2.4 Rozwiązania ZP z rozwiązania ZD

By przekształcić rozwiązanie ZD w rozwiązanie ZP skorzystamy z następujących własności zadania dualnego.

W przypadku znalezienia punktu optymalnego, należy utworzyć rozwiązanie zadania primalnego na podstawie rozwiązania dualnego. Wiemy, że jeżeli znamy rozwiązanie zadania primalnego:

$$\begin{aligned}x &= A_b^{-1}b \\ y &= c_B^T A_b^{-1}\end{aligned}$$

Znajdując odwrotność macierzy bazowej możemy z niej uzyskać zarówno rozwiązanie zadania primalnego oraz zadania dualnego.

Macierz  $A_b^{-1}$  znajduje się w macierzy rozwiązania dualnego - są to kolumny odpowiadające zmiennym bazowym przed pierwszym krokiem algorytmu *simplex*, zmienne, których kolumny tworzyły macierz jednostkową. Ponieważ w zadaniu dualnym wprowadzaliśmy modyfikacje, pozwalające na przeprowadzenie algorytmu sympleks, musimy tę macierz przekształcić, by móc otrzymać rozwiązanie primalne. Przekształcenie polega na tym, że jeżeli w zadaniu dualnym przemnażaliśmy  $i$ -ty wiersz przez  $-1$ , to musimy przemnożyć w macierzy  $A_b^{-1}$   $i$ -tą kolumnę przez  $-1$ . W ten sposób otrzymamy oryginalną odwrotność macierzy bazowej.

Zatem traktując nasze zadanie dualne, jako zadanie pierwotne, możemy z niego uzyskać rozwiązanie dualnego zadania, czyli naszego pierwotnego. Zatem wyliczając

$$x = c_{B_y}^T A_{b_y}^{-1}$$

otrzymamy rozwiązanie naszego zadania pierwotnego.

## 3 Przykłady obliczeniowe

Zaprezentowano ciekawe przypadki obliczeniowe

### 3.1 Zadanie posiadające rozwiązanie

Zaprezentowano zadanie posiadające rozwiązanie. Na rysunku 2 znajduje się tabela zadania dualnego przed pierwszą iteracją, natomiast na rysunku 3 znajduje się końcowa tabela zadania dualnego po wykonaniu algorytmu *simplex*. Rozwiązanie zadania dualnego zaprezentowano na rysunku 4.

T =

8x23 [table](#)

	<b>xB</b>	<b>cB</b>	<b>x1</b>	<b>x2</b>	<b>x3</b>	<b>x4</b>	<b>x5</b>	<b>x6</b>	<b>x7</b>	<b>x8</b>	<b>x9</b>	<b>x10</b>
<b>fC</b>	NaN	NaN	2.8645	2.6183	4.7112	3.7876	1.9188	3.8444	4.5872	3.7381	1.5054	1.5696
<b>r1</b>	16	-4.2964	2.0507	-0.20057	0.99578	-3.8045	3.342	2.9791	-2.6073	0.11928	-2.7345	-3.8295
<b>r2</b>	12	-19.332	-2.4131	-0.17206	4.3894	0.88158	1.8666	-0.21524	1.6919	-1.5383	-3.4069	4.808
<b>r3</b>	18	-12.504	-3.9408	3.45	-2.0151	2.6294	-1.884	1.1901	-4.3137	-2.1881	-3.127	-4.9666
<b>r4</b>	14	-19.434	3.506	4.4992	-3.4792	-1.8166	-4.9869	4.2385	-1.1026	3.5847	2.9162	3.1242
<b>r5</b>	20	-19.003	-0.16395	2.8025	-1.2399	2.8225	1.5175	2.4759	3.3904	2.9391	-0.29798	-2.8282
<b>z</b>	NaN	NaN	22.084	-179.64	27.219	-51.908	41.192	-152.94	-10.568	-68.933	65.699	-21.362
<b>z-c</b>	NaN	NaN	19.219	-182.26	22.508	-55.695	39.273	-156.78	-15.155	-72.671	64.194	-22.932

(a)

<b>x11</b>	<b>x12</b>	<b>x13</b>	<b>x14</b>	<b>x15</b>	<b>x16</b>	<b>x17</b>	<b>x18</b>	<b>x19</b>	<b>x20</b>	<b>B</b>
-24.868	-19.332	-23.093	-19.434	-5.2026	-4.2964	-13.208	-12.504	-5.5746	-19.003	NaN
-1	0	0	0	0	1	0	0	0	0	2.955
0	1	0	0	0	0	-1	0	0	0	4.707
0	0	-1	0	0	0	0	1	0	0	2.0574
0	0	0	1	0	0	0	0	-1	0	2.653
0	0	0	0	-1	0	0	0	0	1	4.6512
4.2964	-19.332	12.504	-19.434	19.003	-4.2964	19.332	-12.504	19.434	-19.003	-269.36
29.164	0	35.597	0	24.206	0	32.54	0	25.008	0	NaN

(b)

Rysunek 2: Początek algorytmu dla zadania dualnego

T =

8x23 [table](#)

	<b>xB</b>	<b>cB</b>	<b>x1</b>	<b>x2</b>	<b>x3</b>	<b>x4</b>	<b>x5</b>	<b>x6</b>	<b>x7</b>	<b>x8</b>	<b>x9</b>	<b>x10</b>
<b>fC</b>	NaN	NaN	2.8645	2.6183	4.7112	3.7876	1.9188	3.8444	4.5872	3.7381	1.5054	1.5696
<b>r1</b>	1	2.8645	1	0	0	0	-0.4831	0	1.756	1.2002	1.4151	2.2676
<b>r2</b>	3	4.7112	0	0	1	0	-0.022735	0	0.97563	0.15145	-0.085238	2.4357
<b>r3</b>	6	3.8444	0	0	0	0	2.546	1	-0.061649	0.27506	-1.4608	-4.1326
<b>r4</b>	2	2.6183	0	1	0	0	-2.7468	0	-0.057228	0.069533	0.96251	4.4598
<b>r5</b>	4	3.7876	0	0	0	1	0.9937	0	1.842	0.86694	0.26462	-0.6048
<b>z</b>	NaN	NaN	2.8645	2.6183	4.7112	3.7876	4.8688	3.8444	16.216	8.6746	1.5585	11.469
<b>z-c</b>	NaN	NaN	0	0	0	0	2.95	0	11.629	4.9365	0.053166	9.8996

(a)

<b>x11</b>	<b>x12</b>	<b>x13</b>	<b>x14</b>	<b>x15</b>	<b>x16</b>	<b>x17</b>	<b>x18</b>	<b>x19</b>	<b>x20</b>	<b>B</b>
-24.868	-19.332	-23.093	-19.434	-5.2026	-4.2964	-13.208	-12.504	-5.5746	-19.003	NaN
0.15922	0.050469	0.25342	0.13249	-0.090974	-0.15922	-0.050469	-0.25342	-0.13249	0.090974	0.020296
0.068618	0.26753	0.11173	0.098276	0.0087088	-0.068618	-0.26753	-0.11173	-0.098276	-0.0087088	1.0468
-0.36732	-0.15429	-0.056752	-0.24166	-0.33491	0.36732	0.15429	0.056752	0.24166	0.33491	1.3926
0.30053	0.29705	-0.008412	0.37746	0.26278	-0.30053	-0.29705	0.008412	-0.37746	-0.26278	0.30664
0.063149	-0.039256	0.12185	-0.112	-0.32287	-0.063149	0.039256	-0.12185	0.112	0.32287	0.58252
0.39325	1.4409	1.4736	0.47754	-2.042	-0.39325	-1.4409	-1.4736	-0.47754	2.042	13.353
25.261	20.773	24.567	19.911	3.1606	3.9032	11.767	11.03	5.0971	21.045	NaN

(b)

Rysunek 3: Koniec algorytmu dla zadania dualnego

```

zDy =

Columns 1 through 15
    0.0203    0.3066    1.0468    0.5825         0    1.3926         0         0         0         0         0         0         0         0         0

Columns 16 through 20
         0         0         0         0         0

```

Rysunek 4: Rozwiązanie zadania dualnego

By otrzymać rozwiązanie zadania pierwotnego skorzystano z własności macierzy bazowej opisanych w 2.4. Na rysunku 5 zaprezentowano odwrotność macierzy bazowej jak, wektor  $c_{B_y}$  oraz uzyskane z nich rozwiązanie zadania prymalnego.

```

Macierz A_b^-1|
A_b_1 =
    -0.1592    -0.0505    -0.2534    -0.1325     0.0910
    -0.0686    -0.2675    -0.1117    -0.0983    -0.0087
     0.3673     0.1543     0.0568     0.2417     0.3349
    -0.3005    -0.2971     0.0084    -0.3775    -0.2628
    -0.0631     0.0393    -0.1218     0.1120     0.3229

Wektor c_B
ans =
     2.8645     4.7112     3.8444     2.6183     3.7876

BRD zadanie pierwotne
zPx =
    -0.3933    -1.4409    -1.4736    -0.4775     2.0420

```

Rysunek 5: Rozwiązanie zadania prymalnego

### 3.2 Zadanie nie posiadające rozwiązania

Zaprezentowano zadanie nie posiadające rozwiązania. Na rysunku 6 znajduje się tabela zadania dualnego przed pierwszą iteracją, natomiast na rysunku 7 znajduje się końcowa tabela zadania dualnego po wykonaniu algorytmu *simplex*. Jak możemy zaobserwować na komunikacie końcowym znajdującym się na rysunku 8 kolumna 2 nie spełnia warunków ograniczoności, współczynnik  $z - c$  dla niej jest ujemny oraz wszystkie jej elementy również, co pokazuje że zadanie dualne jest nieograniczone, zatem zadanie prymalne nie ma rozwiązania.



T =

8x23 [table](#)

	xB	cB	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10
fC	NaN	NaN	1.5744	1.899	3.4383	2.3459	1.9434	1.3529	4.3637	4.1743	1.3563	2.4217
r1	16	-14.539	3.6318	2.2636	-1.3972	-1.1362	-1.7217	1.608	-2.5315	4.9311	-1.1444	-2.4565
r2	17	-9.937	-2.4042	-1.6172	3.3405	-1.3424	-2.6905	0.81349	2.5527	-1.7053	-2.8333	-4.2478
r3	13	-4.2739	1.3185	-4.2807	-0.93744	-3.3448	4.6275	-0.019082	-1.8141	2.4652	-3.0607	1.4879
r4	19	-16.484	-3.1171	3.8401	1.5245	2.914	-4.223	1.8934	-3.56	0.56538	-2.0005	0.66481
r5	20	-5.5431	-3.4137	-4.7192	-3.3381	-0.32636	-3.6544	4.0556	2.1498	3.1961	2.7843	-3.7271
z	NaN	NaN	35.757	-35.688	-15.501	-2.0726	121.86	-85.073	65.96	-92.321	75.417	81.269
z-c	NaN	NaN	34.182	-37.587	-18.939	-4.4185	119.92	-86.426	61.597	-96.495	74.061	78.847

(a)

x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	B
-22.755	-2.8103	-4.2739	-8.8611	-26.261	-14.539	-9.937	-22.348	-16.484	-5.5431	NaN
-1	0	0	0	0	1	0	0	0	0	1.7977
0	-1	0	0	0	0	1	0	0	0	4.5494
0	0	1	0	0	0	0	-1	0	0	4.7403
0	0	0	-1	0	0	0	0	1	0	1.1966
0	0	0	0	-1	0	0	0	0	1	2.6574
14.539	9.937	-4.2739	16.484	5.5431	-14.539	-9.937	4.2739	-16.484	-5.5431	-126.06
37.294	12.747	0	25.345	31.804	0	0	26.622	0	0	NaN

(b)

Rysunek 6: Początek algorytmu dla zadania dualnego

T =

8x23 [table](#)

	xB	cB	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10
fC	NaN	NaN	1.5744	1.899	3.4383	2.3459	1.9434	1.3529	4.3637	4.1743	1.3563	2.4217
r1	8	4.1743	0.2616	-4.2744	0	-2.9929	3.135	0	0	1	-2.5256	0
r2	7	4.3637	-0.84942	-4.7062	0	-2.422	2.6946	0	1	0	-1.549	0
r3	10	2.4217	-1.1172	-3.6982	0	-1.6347	2.5374	0	0	0	-1.5843	1
r4	3	3.4383	-0.80144	-3.4065	1	-2.2152	2.1355	0	0	0	-2.8869	0
r5	6	1.3529	-2.284	-1.5029	0	0.23641	-0.71042	1	0	0	-0.33421	0
z	NaN	NaN	-11.166	-61.081	3.4383	-34.318	37.371	1.3529	4.3637	4.1743	-31.517	2.4217
z-c	NaN	NaN	-12.74	-62.98	0	-36.664	35.428	0	0	0	-32.873	0

(a)

x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	B
-22.755	-2.8103	-4.2739	-8.8611	-26.261	-14.539	-9.937	-22.348	-16.484	-5.5431	NaN
0.082173	-0.25322	0.72395	0.10752	-0.035391	-0.082173	0.25322	-0.72395	-0.10752	0.035391	4.4015
0.28448	-0.23875	0.59642	0.13295	-0.12978	-0.28448	0.23875	-0.59642	-0.13295	0.12978	3.5878
0.33168	-0.095781	0.60043	-0.053124	-0.090319	-0.33168	0.095781	-0.60043	0.053124	0.090319	2.9893
0.18709	-0.35483	0.63215	-0.053793	0.019135	-0.18709	0.35483	-0.63215	0.053793	-0.019135	4.2891
0.24325	-0.053965	0.18543	-0.24831	-0.21714	-0.24325	0.053965	-0.18543	0.24831	0.21714	1.5614
3.36	-3.6238	9.503	0.37945	-1.1608	-3.36	3.6238	-9.503	-0.37945	1.1608	58.124
26.115	-0.81349	13.777	9.2405	25.1	11.179	13.561	12.845	16.105	6.7038	NaN

(b)

Rysunek 7: Koniec algorytmu dla zadania dualnego

```

Kolumna 2 spełnia warunki nieograniczoności.
Zadanie nieograniczone.

x =

     []

zPx =

     []

```

Rysunek 8: Komunikat końcowy

## 4 Porównanie algorytmów

Zostały porównane funkcje `linprog` oraz własna implementacja algorytmu *simplex* rozwiązująca zadanie dualne oraz przekształcająca to rozwiązanie w rozwiązanie zadania primalnego. W tabeli 1 zaprezentowane są wyniki określające czy algorytmy uzyskiwały te same wyniki dla tego samego problemu. Przeprowadzono 1000 losowych symulacji. Z powodu naturalnego dla obliczeń numerycznych błędu zaokrągleń zastosowano następującą formułę porównania rozwiązań obu algorytmów:

$$abs(x_{linprog} - x_{simplex}) < eps$$

W obliczeniach przyjęto  $eps = 10^{-5}$

Tabela 1: Porównanie algorytmów - 1000 symulacji

Przypadek		Liczba przypadków
Linprog	Symplex	
Brak rozwiązania		668
Brak rozwiązania	Rozwiązanie	0
Rozwiązanie	Brak rozwiązania	0
To samo rozwiązanie		332
Różne rozwiązania		0

Możemy zauważyć że algorytmy zawsze uzyskiwały te same wyniki co dowodzi poprawności implementacji algorytmu *simplex* rozwiązującego zadanie dualne.

## Bibliografia

- [1] Linprog Matlab <https://uk.mathworks.com/help/optim/ug/linprog.html>