Piotr Grabarski 05.29.2020

Wyznaczanie optymalnego rozwiązania problemu planowania produkcji przy użyciu Cplex

Table of Contents

1. Treść zadania

WDWR 20202

Rozważamy następujące zagadnienie planowania produkcji:

- Realizacja umowy wymaga dostawy 1100 sztuk komponentu A oraz 1200 sztuk komponentu B po upływie okresu 3 miesięcy.
- Koszty produkcji komponentów (zł/szt.) określają składowe wektora losowego $\mathbf{R} = (R_1, \ldots, R_6)^T$:

	Miesiąc 1	Miesiąc 2	Miesiąc 3
A	R_1	R_2	R_3
B	R_4	R_5	R_6

- Wektor losowy $\mathbf{R}$ opisuje 6-wymiarowy rozkład t-Studenta z 5 stopniami swobody, którego wartości składowych zostały zawężone do przedziału $[20; 60]$. Parametry $\boldsymbol{\mu}$ oraz $\boldsymbol{\Sigma}$ niezawężonego rozkładu t-Studenta są następujące:

$$\boldsymbol{\mu} = \begin{pmatrix} 55 \\ 40 \\ 50 \\ 35 \\ 45 \\ 30 \end{pmatrix}, \qquad \boldsymbol{\Sigma} = \begin{pmatrix} 1 & 1 & 0 & 2 & -1 & -1 \\ 1 & 16 & -6 & -6 & -2 & 12 \\ 0 & -6 & 4 & 2 & -2 & -5 \\ 2 & -6 & 2 & 25 & 0 & -17 \\ -1 & -2 & -2 & 0 & 9 & -5 \\ -1 & 12 & -5 & -17 & -5 & 36 \end{pmatrix}.$$

- Koszt składowania komponentu A z miesiąca na miesiąc kształtuje się na poziomie 10% miesięcznych kosztów wytwarzania dla pierwszych 100 sztuk i 15% dla pozostałych. Analogicznie dla komponentu B.
- Firma w celu wytworzenia komponentów potrzebuje zasobów pozyskiwanych z zewnątrz. Szczegóły dostępnych dostaw i wymagań są następujące:

Zasób produkcyjny	Zapotrzebowanie na sztukę		Możliwe dostawy		
	A	B	Miesiąc 1	Miesiąc 2	Miesiąc 3
Z1	0,2	0,7	600	700	550
Z2	0,8	0,3	1400	900	1200

1. Zaproponować jednokryterialny model wyboru w warunkach ryzyka z wartością oczekiwaną jako miarą kosztu. Wyznaczyć rozwiązanie optymalne.
2. Jako rozszerzenie powyższego zaproponować dwukryterialny model kosztu i ryzyka z wartością oczekiwaną jako miarą kosztu i odchyleniem przeciętnym jako miarą ryzyka. Dla decyzji $\mathbf{x} \in Q$ odchylenie przeciętne jest definiowane jako $\delta(\mathbf{x}) = \sum_{t=1}^{T} |\mu(\mathbf{x}) - r_t(\mathbf{x})| p_t$, gdzie $\mu(\mathbf{x})$ oznacza wartość oczekiwaną, $r_t(\mathbf{x})$ realizację dla scenariusza t, p_t prawdopodobieństwo scenariusza t.
   a. Wyznaczyć obraz zbioru rozwiązań efektywnych w przestrzeni ryzyko–koszt.
   b. Wskazać rozwiązania efektywne minimalnego ryzyka i minimalnego kosztu. Jakie odpowiadają im wartości w przestrzeni ryzyko–koszt?
   c. Wybrać trzy dowolne rozwiązania efektywne. Sprawdzić czy zachodzi pomiędzy nimi relacja dominacji stochastycznej pierwszego rzędu. Wyniki skomentować, odnieść do ogólnego przypadku.

2. Wstęp teoretyczny

2.1. Rozkład t-studenta

Rozkładem t-Studenta nazywamy ciągły rozkład prawdopodobieństwa stosowany w statystyce do oceny niepewności pomiaru, testów hipotez statystycznych I konstrukcji przedziałów ufności.

Stosuje się go gdy rozmiar próby jest mniejszy niż 30 a odchylenie standardowe populacji jest nieznane.

Gdy mamy zmienną losowa R mającą niestandardowy rozkład t-Studenta zawężony do przedziału $(\alpha;\ \beta)$ to możemy to zapisać: $R \sim t_{(\alpha;\beta)}(\mu,\ \sigma^2;\ \nu))$.

Wartości oczekiwane zmiennej losowej R możemy wyznaczyć z następującego wzoru:

$$\mathbb{E}(R) = \mu + \sigma \frac{\Gamma((v-1)/2)((v+a^2)^{-(v-1)/2} - (v+b^2)^{-(v-1)/2})v^{v/2}}{2(F_v(b) - F_v(a))\Gamma(v/2)\Gamma(1/2)} \quad \text{dla } v > 1.$$

gdzie:

$\Gamma(\ \boldsymbol{\cdot}\ )$ to funkcja gamma Eulera

$F_\nu(\ \boldsymbol{\cdot}\ )$ to dystrybuanta standardowego rozkładu t-Studenta $t(0,\ 1;\ \nu)$ z ν stopniami swobody

$a = (\alpha - \mu)/\sigma$

$b = (\beta - \mu)/\sigma$

2.2. Modelowanie zależności

By zbudować model rzeczowy, trzeba najpierw zidentyfikować problem i zamodelować wiele złożonych zależności. Dla olbrzymiej liczby zależności wystarczająco dobrym przybliżeniem są funkcje lub nierówności liniowe. Odpowiednie zadania programowania matematycznego formułowane w oparciu o zależności liniowe nazywane są zadaniami programowania liniowego.

Obecnie istnieje na rynku wiele solwerów pozwalających rozwiązywać zadania programowania liniowego nawet o bardzo dużych rozmiarach (kilkadziesiąt tysięcy zmiennych i ograniczeń) np.

CPLEX, XPRESS CZY LINDO.

Typowym przykładem programowania liniowego jest właśnie problem planowania produkcji, który polega na przekształceniu zestawu surowców w zestaw produktów (podobnie jest w omawianym tu zadaniu).

2.3. Odchylenie przeciętne

Odchylenie przeciętne nazywane jest też czasami średnim odchyleniem bezwzględnym. Definiuje je się jako średnią arytmetyczną z odchyleń bezwzględnych dla wszystkich elementów zbioru danych statycznych.

Można je zapisać jako:

$$\delta(\mathbf{y}) = \frac{1}{2m} \sum_{i=1}^{m} |\mu(\mathbf{y}) - y_i| = \frac{1}{m} \sum_{i: y_i < \mu(\mathbf{y})} [\mu(\mathbf{y}) - y_i]$$

lub

$$\delta(\mathbf{y}) = \frac{1}{m} \sum_{i: \theta_i(\mathbf{y}) < \mu(\mathbf{y})} [\mu(\mathbf{y}) - \theta_i(\mathbf{y})] = \frac{1}{m} \max_{i=1,\ldots,m-1} [\frac{i}{m}\bar{\theta}_m(\mathbf{y}) - \bar{\theta}_i(\mathbf{y})].$$

2.4. Dominacja stochastyczna pierwszego rzędu

Słaba relacji dominacji stochastycznej pierwszego rzędu jest określona:

$$Y' \succeq_{FSD} Y'' \quad \Leftrightarrow \quad F_{Y'}(v) \leq F_{Y''}(v) \quad \forall v \in R$$

A relacja ścisła jest określona jako:

$$Y' \succ_{FSD} Y'' \quad \Leftrightarrow \quad Y' \succeq_{FSD} Y'' \quad \text{i} \quad Y'' \not\succeq_{FSD} Y'$$

Zmienna losowa Y' dominuje zmienną losową Y'' w sensie dominacji stochastycznej pierwszego rzędu jeżeli dla każdego v należącego do zbioru liczb rzeczywistych prawdziwa jest nierówność:

$F_{Y'}(v) \leqslant F_{Y''}(v)$ I dla przynajmniej jednej wartości x0 jest to nierówność ostra.

3. Model jednokryterialny

3.1 Wyznaczenie wartości oczekiwanych z rozkładu t-Studenta

Zadanie zostało zrealizowane przy pomocy programu napisanego w Pythonie z użyciem biblioteki SciPy. Kod został napisany zgodnie z dobrymi praktykami programistycznymi oraz z testami jednostkowymi. Do testów został wykorzystany przykład z materiałów uzupełniających.

Przykład. Wektor losowy $\mathbf{R}$ opisuje 3-wymiarowy rozkład t-Studenta z 4 stopniami swobody, którego wartości składowych zostały zawężone do przedziału $[20; 50]$. Parametry $\boldsymbol{\mu}$ oraz $\boldsymbol{\Sigma}$ niezawężonego rozkładu t-Studenta są następujące:

$$\boldsymbol{\mu} = \begin{pmatrix} 45 \\ 35 \\ 40 \end{pmatrix}, \qquad \boldsymbol{\Sigma} = \begin{pmatrix} 1 & -2 & -1 \\ -2 & 36 & -8 \\ -1 & -8 & 9 \end{pmatrix}.$$

Należy wyznaczyć wartości oczekiwane zawężonych rozkładów brzegowych.

Rozwiązanie. Rozkład t-Studenta jest ciągły, więc wartość oczekiwana na przedziale domkniętym jest taka sama jak na przedziale otwartym. Z warunków zadania mamy $R_1 \sim Tt_{(20;50)}(45, 1; 4)$, $R_2 \sim Tt_{(20;50)}(35, 36; 4)$ i $R_3 \sim Tt_{(20;50)}(40, 9; 4)$. Po podstawieniu danych do (2) dostajemy:

$$\mathbb{E}(R_1) = 45 + 1 \cdot \frac{\Gamma(3/2)((4 + (-25)^2)^{-3/2} - (4 + 5^2)^{-3/2}) \cdot 4^2}{2(F_4(5) - F_4(-25))\Gamma(2)\Gamma(1/2)} \approx 44{,}97,$$

$$\mathbb{E}(R_2) = 35 + 6 \cdot \frac{\Gamma(3/2)((4 + (-5/2)^2)^{-3/2} - (4 + (5/2)^2)^{-3/2}) \cdot 4^2}{2(F_4(5/2) - F_4(-5/2))\Gamma(2)\Gamma(1/2)} = 35,$$

$$\mathbb{E}(R_3) = 40 + 3 \cdot \frac{\Gamma(3/2)((4 + (-20/3)^2)^{-3/2} - (4 + (10/3)^2)^{-3/2}) \cdot 4^2}{2(F_4(10/3) - F_4(-20/3))\Gamma(2)\Gamma(1/2)} \approx 39{,}83.$$

Sprawdzam w nich czy dla tych samych danych wynik z powyższego przykładu jest zgodny ze zwróconym przez program, z dokładnością do dwóch liczb po przecinku.

Wyznaczone wartości oczekiwane:

- R1 = 54.98679995962599
- R2 = 40.0
- R3 = 49.97403331742613
- R4 = 35.24083494467262
- R5 = 44.966804314236484
- R6 = 31.193302160152246

Poniżej zamieszczam kod źródłowy.

t_dist.py+

```python
import math
from scipy.stats import t
from scipy.special import gamma


class TStudentDist:
    """
    Class responsible for calculating expected values of student-t distribution
    """

    def __init__(self, mu_matrix, sigma_matrix, alpha, beta, v):
        self.mu_matrix = mu_matrix
        self.sigma_matrix = sigma_matrix
        self.alpha = alpha
        self.beta = beta
        self.v = v

    def _calculate_param(self, base, mu, sigma):
        """
        Calculate value of beta or alpha parameter
        """
        return (base - mu) / sigma

    def calculate_expected_value(self, index):
        """
        Calculate expected value using formula from lecture
        """
        result = self.mu_matrix[index]
        sigma = math.sqrt(self.sigma_matrix[index][index])
        a = self._calculate_param(self.alpha, self.mu_matrix[index], sigma)
        b = self._calculate_param(self.beta, self.mu_matrix[index], sigma)
        result += sigma * (gamma((self.v - 1) / 2) * ((self.v + a ** 2) ** ((1 - self.v) / 2) -
                                                      (self.v + b ** 2) ** ((1 - self.v) / 2))
                           * self.v ** (self.v / 2) / (2 * (t.cdf(b, self.v) - t.cdf(a, self.v))
                                                       * gamma(self.v / 2) * gamma(1 / 2)))
        return result


def main():
    mu_matrix = [55, 40, 50, 35, 45, 30]
    sigma_matrix = [
        [1, 1, 0, 2, -1, -1],
        [1, 16, -6, -6, -2, 12],
        [0, -6, 4, 2, -2, -5],
        [2, -6, 2, 25, 0, -17],
        [-1, -2, -2, 0, 9, -5],
        [-1, 12, -5, 17, -5, 36]
    ]
    alpha = 20
    beta = 60
    v = 5

    t_student_dist = TStudentDist(mu_matrix, sigma_matrix, alpha, beta, v)

    e_vector = []
    for mu in range(0, len(mu_matrix)):
        e_vector.append(t_student_dist.calculate_expected_value(mu))

    for exp in e_vector:
        print(exp)


if __name__ == '__main__':
    main()
```

3.2 Model i rozwiązanie

Zadane zagadnienie planowania produkcji zamodelowane zostało w Pythonie przy użyciu solvera Cplex. Model zamieściłem niżej w postaci kodu z podziałem na odpowiednie sekcje – zmienne, ograniczenia, funkcję celu oraz z odpowiednimi komentarzami. Jako wartości kosztów produkcji komponentów w danym miesiącu użyłem tutaj wartości oczekiwanych uzyskanych w punkcie 3.1 z rozkładu t-Studenta.

Przyjąłem założenie, że produkty można w każdym miesiącu produkować i składować jedynie w całości – nie możemy zamówić np. bardzo dużo zasobu Z1 do produkcji komponentu A w pierwszym miesiącu i potem zasobu Z2 do produkcji komponentu A w kolejnym miesiącu, bo chcemy by w każdym miesiącu wyprodukowana została całkowita liczba produktu A i jedynie pełnego produktu A (a nie jedynie części z zasobu Z1).

Zastanawiałem się nad interpretacją kosztu składowania danego komponentu w danym miesiącu. Finalnie przyjąłem, następujące założenia:

- Zasada 100 tańszych sztuk odnawia się w każdym miesiącu, by zachęcić do regularnego składowania.
- W każdym kolejnym miesiącu musimy płacić dodatkowo za składowanie produktów wyprodukowanych w poprzednich miesiącach
- Koszt składowania produktów wyprodukowanych w poprzednich miesiącach jest taki sam, jak w miesiącach, gdy zostały wyprodukowane. Oznacza to, że „promocja” na pierwsze 100 sztuk z pierwszego miesiąca, będzie działać też w kolejnych miesiącach.

Wynik:

- Wartość funkcji celu – minimalny koszt - **objective: 101296.561**
- Ilość produktu A wyprodukowanego w drugim miesiącu - **A_month_2=1100.000**
- Ilość produktu A wyprodukowanego w trzecim miesiącu **- B_month_3=1200.000**
- Ilość zasobu Z1 przeznaczonego na produkt A w drugim miesiącu - **Z1Am2=220.000**
- Ilość zasobu Z1 przeznaczonego na produkt B w trzecim miesiącu - **Z1Bm3=840.000**
- Ilość zasobu Z2 przeznaczonego na produkt A w drugim miesiącu - **Z2Am2=880.000**
- Ilość zasobu Z2 przeznaczonego na produkt B w trzecim miesiącu - **Z2Bm3=360.000**

```python
from docplex.mp.model import Model


def main():
    m = Model(name='production_planning')

    #############################
    ### Continuous Variables ###
    #############################

    # Ilość wyprodukowanego produktu A i B w każdym z miesięcy
    Am1 = m.continuous_var(name='A_month_1')
    Am2 = m.continuous_var(name='A_month_2')
    Am3 = m.continuous_var(name='A_month_3')
    Bm1 = m.continuous_var(name='B_month_1')
    Bm2 = m.continuous_var(name='B_month_2')
    Bm3 = m.continuous_var(name='B_month_3')

    # Ilość zasobu Z1 przeznaczonego na produkt A i B w każdym z miesięcy
    Z1Am1 = m.continuous_var(name='Z1Am1')
    Z1Am2 = m.continuous_var(name='Z1Am2')
    Z1Am3 = m.continuous_var(name='Z1Am3')
    Z1Bm1 = m.continuous_var(name='Z1Bm1')
    Z1Bm2 = m.continuous_var(name='Z1Bm2')
    Z1Bm3 = m.continuous_var(name='Z1Bm3')

    # Ilość zasobu Z2 przeznaczonego na produkt A i B w każdym z miesięcy
    Z2Am1 = m.continuous_var(name='Z2Am1')
    Z2Am2 = m.continuous_var(name='Z2Am2')
    Z2Am3 = m.continuous_var(name='Z2Am3')
    Z2Bm1 = m.continuous_var(name='Z2Bm1')
    Z2Bm2 = m.continuous_var(name='Z2Bm2')
    Z2Bm3 = m.continuous_var(name='Z2Bm3')

    ###################
    ### Constraints ###
    ###################

    # Ograniczenia liczby sztuk wymaganych do dostawy
    m.add_constraint(Am1 + Am2 + Am3 == 1100)
    m.add_constraint(Bm1 + Bm2 + Bm3 == 1200)

    # Ograniczenia, określające ile komponentu A i B można wyprodukować w każdym miesiącu
    # zależnie od wielkości dostaw komponentu Z1 i Z2
    m.add_constraint(Am1 == (Z1Am1 / 0.2 + Z2Am1 / 0.8) / 2)
    m.add_constraint(Am2 == (Z1Am2 / 0.2 + Z2Am2 / 0.8) / 2)
    m.add_constraint(Am3 == (Z1Am3 / 0.2 + Z2Am3 / 0.8) / 2)
    m.add_constraint(Bm1 == (Z1Bm1 / 0.7 + Z2Bm1 / 0.3) / 2)
    m.add_constraint(Bm2 == (Z1Bm2 / 0.7 + Z2Bm2 / 0.3) / 2)
    m.add_constraint(Bm3 == (Z1Bm3 / 0.7 + Z2Bm3 / 0.3) / 2)

    # Ograniczenia wymuszające produkowanie komponentów A i B w każdym miesiacu w całości
    m.add_constraint(Z1Am1 / 0.2 == Z2Am1 / 0.8)
    m.add_constraint(Z1Am2 / 0.2 == Z2Am2 / 0.8)
    m.add_constraint(Z1Am3 / 0.2 == Z2Am3 / 0.8)
    m.add_constraint(Z1Bm1 / 0.7 == Z2Bm1 / 0.3)
    m.add_constraint(Z1Bm2 / 0.7 == Z2Bm2 / 0.3)
    m.add_constraint(Z1Bm3 / 0.7 == Z2Bm3 / 0.3)
```

```python
    # Ograniczenia możliwych dostaw komponentów A i B w każdym miesiącu
    m.add_constraint(Z1Am1 + Z1Bm1 <= 600)
    m.add_constraint(Z2Am1 + Z2Bm1 <= 700)
    m.add_constraint(Z1Am2 + Z1Bm2 <= 550)
    m.add_constraint(Z2Am2 + Z2Bm2 <= 1400)
    m.add_constraint(Z1Am3 + Z1Bm3 <= 900)
    m.add_constraint(Z2Am3 + Z2Bm3 <= 1200)

    ##########################
    ### Objective function ###
    ##########################

    # Wyliczone własności oczekiwane
    R1 = 54.98679995962599
    R2 = 40.0
    R3 = 49.97403331742613
    R4 = 35.24083494467262
    R5 = 44.966804314236484
    R6 = 31.193302160152246

    # Koszt produkcji
    prod_costA = Am1 * R1 + Am2 * R2 + Am3 * R3
    prod_costB = Bm1 * R4 + Bm2 * R5 + Bm3 * R6

    # Koszt składowania komponentu A
    store_A_mon1 = R1 * (1/10 * Am1 + (5/100 * (Am1 - 100) + m.abs(5/100 * (Am1 - 100)) / 2))
    store_A_mon2 = store_A_mon1 + R2 * \
        (1/10 * Am2 + (5/100 * (Am2 - 100) + m.abs(5/100 * (Am2 - 100)) / 2))
    store_A_mon3 = store_A_mon1 + store_A_mon2 + R3 * \
        (1/10 * Am3 + (5/100 * (Am3 - 100) + m.abs(5/100 * (Am3 - 100)) / 2))
    store_costA = store_A_mon1 + store_A_mon2 + store_A_mon3

    # Koszt składowania komponentu B
    store_B_mon1 = R4 * (1/10 * Bm1 + (5/100 * (Bm1 - 100) + m.abs(5/100 * (Bm1 - 100)) / 2))
    store_B_mon2 = store_B_mon1 + R5 * \
        (1/10 * Bm2 + (5/100 * (Bm2 - 100) + m.abs(5/100 * (Bm2 - 100)) / 2))
    store_B_mon3 = store_B_mon1 + store_B_mon2 + R6 * \
        (1/10 * Bm3 + (5/100 * (Bm3 - 100) + m.abs(5/100 * (Bm3 - 100)) / 2))
    store_costB = store_B_mon1 + store_B_mon2 + store_B_mon3

    # Funkcja celu - minimalizacja kosztu produkcji i składowania
    m.minimize(prod_costA + prod_costB + store_costA + store_costB)

    # Wyświetlenie wyników
    m.print_information()
    result = m.solve()
    print("\n")
    m.print_solution()


if __name__ == "__main__":
    main()
```

4. Model rozszerzony

4.1. Wygenerowanie scenariuszy

Do wygenerowania scenariuszy napisany został skrypt w R, gdyż w Pythonie funkcja dostępna w SciPy (t.rvs()) nie pozwalała na użycie macierzy sigma oraz ograniczeń zdefiniowanych w zadaniu.

```r
library(tmvtnorm)

mu_matrix = c(55, 40, 50, 35, 45, 30)
sigma_matrix = matrix(c(1, 1, 0, 2, -1, -1,
                        1, 16, -6, -6, -2, 12,
                        0, -6, 4, 2, -2, -5,
                        2, -6, 2, 25, 0, -17,
                        -1, -2, -2, 0, 9, -5,
                        -1, 12, -5, -17, -5, 36),
                      nrow=6, ncol=6)

# Generate scenarios
data <- rtmvt(n=1000, mean=mu_matrix, sigma=sigma_matrix, df=5,
              lower=rep(20, 6), upper=rep(60, 6))

# Write scenarios to file
write.table(format(data, digits=15, drop0trailing=F), "scenarios.txt",
            quote=F, sep="\t", eol="\n\t", col.names = F, row.names = T)
```

4.2. Model dwukryterialny

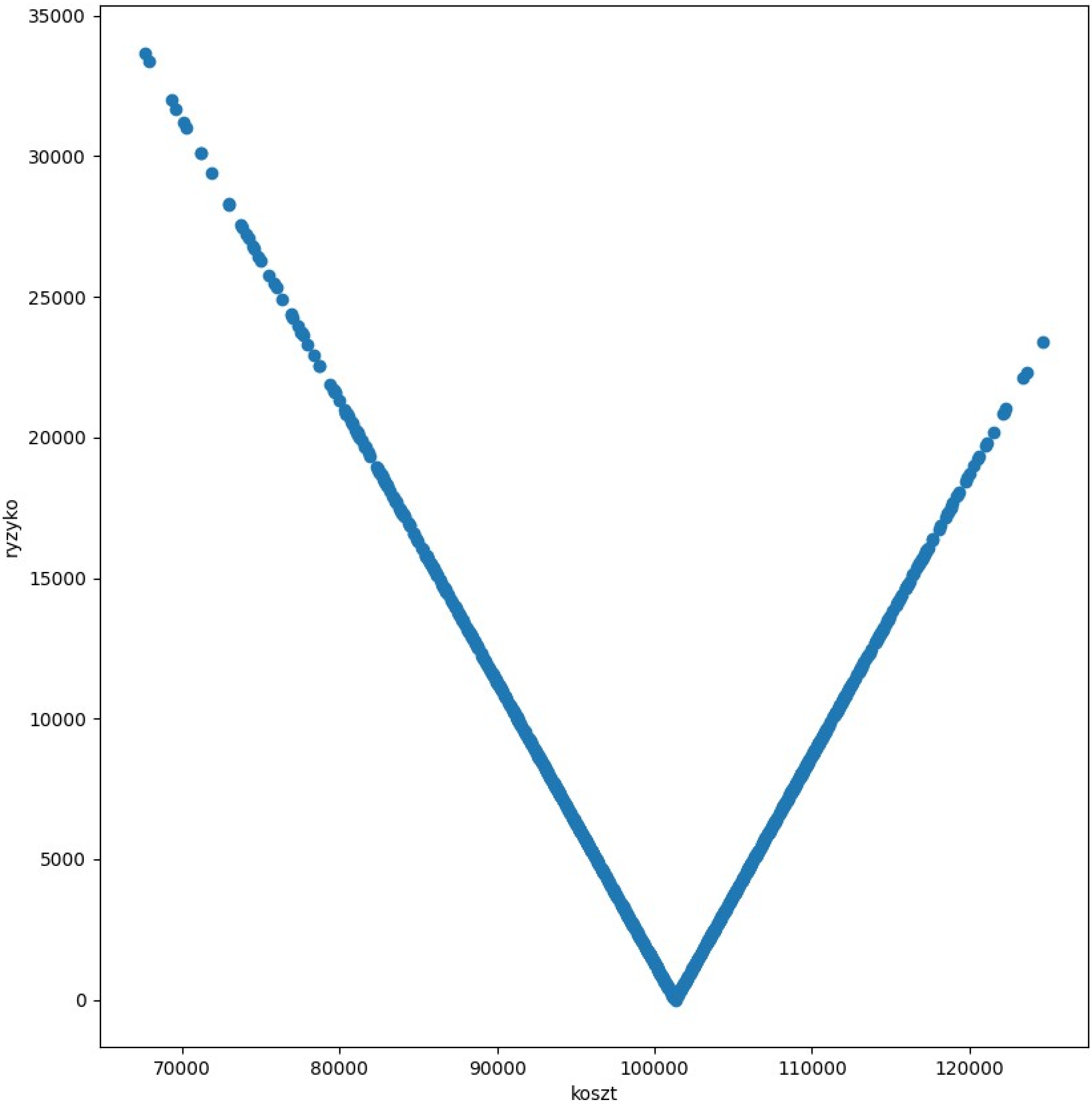
Podczas tworzenia zadanego modelu dwukryterialnego w narzędziu, w którym wykonałem model jednokryterialny (docplex), natrafiłem na problem. Po dodaniu drugiego kryterium – ryzyka, równego odchyleniu przeciętnemu, nie mogłem doprowadzić do zakończenia obliczeń przez solwer. Niezależnie od liczby scenariuszy, nie pozwalał on na tego typu operację i zwracał poniższy błąd:

```
cplex.exceptions.errors.CplexSolverError: CPLEX Error  1016: Community Edition. Problem size limits exceeded. Purchase
at http://ibm.biz/error1016.
```

Zgodnie z tym co wyświetlał komunikat o błędzie mogło to być skutkiem ograniczeń darmowej wersji docplex. Z tego względu wykonałem prostszy model dwukryterialny z odchyleniem od wartości oczekiwanej dla scenariusza ***t*** jako miarą ryzyka.

risk = model.abs(expected_value - cost(t))

4.3. Obraz zbioru rozwiązań efektywnych



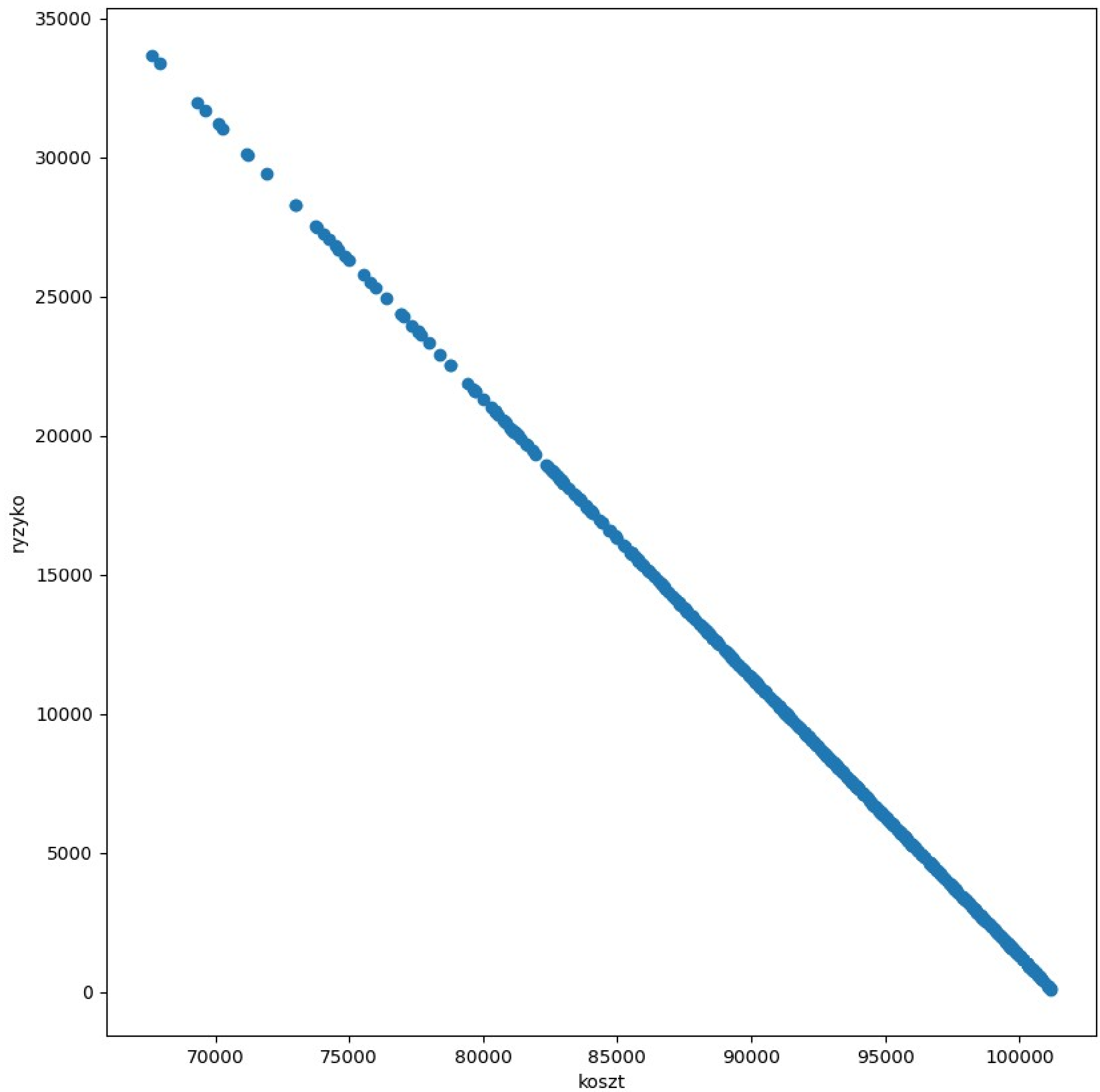
Do narysowania wykorzystana została Pythonowa biblioteka matplotlib.

```
draw_plot.py+
import matplotlib.pyplot as plt

def main():
    cost = []
    risk = []
    with open("risk.txt") as f:
        lines = f.readlines()
        for line in lines:
            values = line.split(", ")
            cost.append(float(values[0]))
            risk.append(float(values[1]))

    plt.scatter(cost, risk)
    plt.ylabel('ryzyko')
    plt.xlabel('koszt')
    plt.show()
```

4.4. Rozwiązania efektywne minimalnego kosztu i ryzyka



Przykładowo:

1. dla minimalnego ryzyka - koszt: 101306.0, ryzyko: 9.0, suma: 101315.0
2. dla minimalnego kosztu i maksymalnego ryzyka: koszt: 67620.0, ryzyko: 33676.0, suma: 101296.0
3. dla innej wartości - koszt: 98925.0, ryzyko: 2371.0, suma: 101296.0

Zgodnie z definicją dominacji stochastycznej można stwierdzić, że rozwiązanie 2) i 3) dominuje w sensie FSD rozwiązanie 1)