MINISTRY OF EDUCATION AND SCIENCE OF THE RUSSIAN FEDERATION
State autonomous educational institution of higher education
«Lobachevsky State University of Nizhni Novgorod»

Markina M.V., Yujie Geng

Pareto optimal solutions for multi-objective optimization problems

Teaching aid
For students studying in English

Recommended by the Methodological Commission of the IITMM
for students of UNN, students in the field of training
"Fundamental informatics and information technology"

Nizhni Novgorod 2018

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное

учреждение высшего образования

«Национальный исследовательский Нижегородский

государственный университет им. Н.И. Лобачевского»

М.В. Маркина. Юйцзе Гэн

Парето –оптимальные решения многокритериальных задач оптимизации

Учебно-методическое пособие

Для студентов, обучающихся на английском языке

Рекомендовано методической комиссией ИИТММ

для студентов ННГУ, обучающихся по направлению подготовки

“ фундаментальная информатика и информационные   технологии”

Нижний Новгород

2018

УДК 519.85
ББК В22.18

Маркина М.В., Юйцзе Гэн. Парето –оптимальные решения многокритериальных задач оптимизации: учебно-методическое пособие. – [электронный ресурс] Нижний Новгород: Нижегородский госуниверситет, 2018. - 37с.

Рецензент: к.ф.-м. н., доцент **Круглов Е.В.**

Настоящее пособие содержит материалы по курсу «Исследование операций», читаемого бакалаврам направления подготовки «фундаментальная информатика и информационные системы» ИИТММ на английском языке. Рассматриваются разделы: определения понятия оптимальности по Парето, нахождение решений оптимальных по Парето с использованием метода свертки и метода последовательных уступок. Приводятся коды программ на языке Python.

. Пособие предназначено для студентов и магистров, специализирующихся в области теории математического моделирования, оптимизации и математического программирования

УДК 519.85
ББК 22.18

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Statement of Multicriterial Optimization Problem

We may face such problems in our real life: suppose you want to buy a house in the city where you work, there are some aspects you need to consider: Convenience(close to the place where you work or near the metro station or bus stop), price which you may afford, house condition. Now there are four places that you feel satisfied but not yet make decision: let's denote them as H_1 , H_2, H_3, H_4. See as the followings:

	H1	H2	H3	H4
Convenience	2hours' drive	30mins' drive	Near the office	4hours' drive
Price	20 0000$	1million$	4millions$	15 0000$
House Condition	Normal	Bad	Brand-New	Good

Now your expectation is to buy a house that is cheap ,close to office and good-looking one, Now there are four alternatives and three criterions which made up of our decision factors. How do we decide?

Such problem is an example of multicriterion problem, there are also many areas which are tightly connected to optimization problem. A multi-objective optimization problem is an optimization problem that involves multiple objective functions. In mathematical terms, a multi-objective optimization problem can be formulated as:

$$\min \quad y = f(x) = [f_1(x), f_2(x), \cdots, f_n(x)]$$

$$n = 1, 2, \cdots, N$$

$$\text{s t} \quad g_i(x) \leqslant 0 \quad i = 1, 2, \cdots, m$$

$$h_j(x) = 0 \quad j = 1, 2, \cdots, k$$

$$x = [x_1, x_2, \cdots, x_d, \cdots, x_D]$$

$$x_{d\_\min} \leqslant x_d \leqslant x_{d\_\max} \quad d = 1, 2, \cdots, D$$

Here $f_n(X)$ are our objective functions, N is the number of our objective functions, $G_i(x) <=0$ and h_j are the constrains ,X is decision space that's made up of decision vector ,X_{d_min} snd X_{d_max} are the lower bounds and upper bound. Y is the objective space.

For a nontrivial multi-objective optimization problem, no single solution exists that simultaneously optimizes each objective. That is ,for a single criterion, we may get a optimum value at somewhere a point X^*: $f(X^{*)} =$ opt $f(X^*)$.but it is almost impossible to get all optimum values for each objective functions simultaneously such that y = opt $[f_1(X^*), f_2(X^*), \ldots ,f_n(X^*)]$.

1.2 Solutions of Multicriteria Optimization Problem

Pareto Optimality is also called Pareto efficiency. First let's see the definition：

Pareto improvement:

An allocation can be Pareto improvement if there exists another allocation such that at least one person is better off and nobody is worse off.

Pareto efficiency :【4】

An allocation is Pareto efficiency if no Pareto improvement is possible.

In mathematics, given a point $X^* \in S$，here S is the variable feasible region. If for any $X \in S$, $f(X^*)<f(X)$, then X^* is called the global optimum，if there doesn't exists $X \in S$, such that $f(X) < f(X^*)$, then X^* is called Pareto Optimum.

The Pareto frontier is the set of all Pareto efficient allocations.

Example1:【1】

If all bread and cloth were given to Ms. Smith, she would perceive let us say 700 units of utility. Mr. Novak would get nothing and his utility would be 0. If all the goods were given to Mr. Novak, he may perceive 500 units of utility and Ms. Smith would get 0. Pareto frontier is the black curve connecting these two points. It depicts the maximum possible welfare of this pair of people.

For example point A representing 200 units of welfare for Mr. Novak and 300 units of welfare of Ms. Smith is inefficient, because some changes in the allocation of resources, technology and/or distribution of consumer goods can improve the welfare of one or both of them.

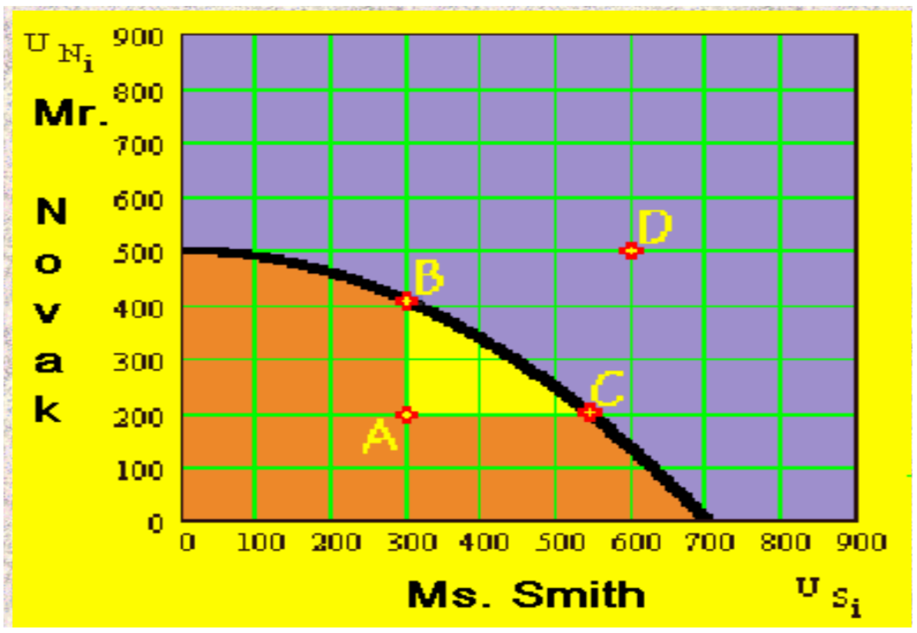


Figure 1

CHAPTER 2

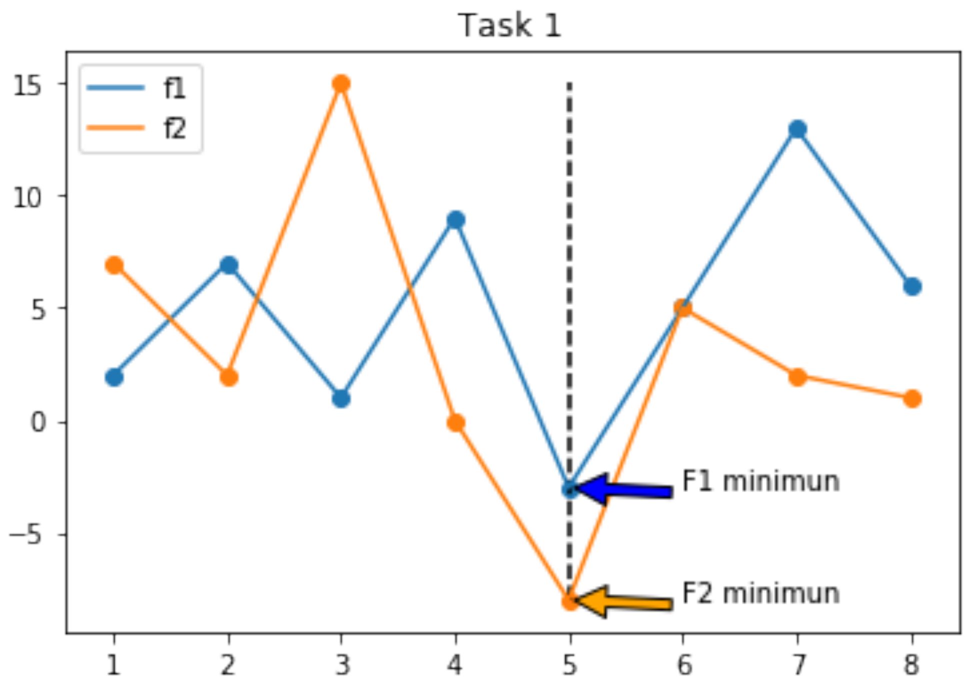
PARETO SET TASK EXAMPLES

2.1 Example 1

(F1,F2)→ min

x	1	2	3	4	5	6	7	8
F1	2	7	1	9	-3	5	13	6
F2	7	2	15	0	-8	5	2	1

```
'''find min[F1, F2]'''

import matplotlib.pyplot as plt

# Raw data
x = list(range(1,9))
F1 = [2,7,1,9,-3,5,13,6]
F2 = [7,2,15,0,-8,5,2,1]

#plot the scatter chart
plt.title("Task 1")
plt.scatter(x, F1)
plt.scatter(x, F2)



# plot the line chart
plt.plot(x, F1, label="f1")
plt.plot(x, F2, label="f2")
plt.legend(loc='upper left')
plt.annotate("F1 minimun", xy=(5,-3), xytext=(6,-3),
                    arrowprops=dict(facecolor='blue', shrink=0.05))
plt.annotate("F2 minimun", xy=(5,-8), xytext=(6,-8),
                    arrowprops=dict(facecolor='orange', shrink=0.05))
plt.vlines(5,-8,15,linestyles = "dashed")
plt.show()

# solution


'''It is easy to find that the minimum points are
the place where X=5, and actually they are global optimum
'''
```



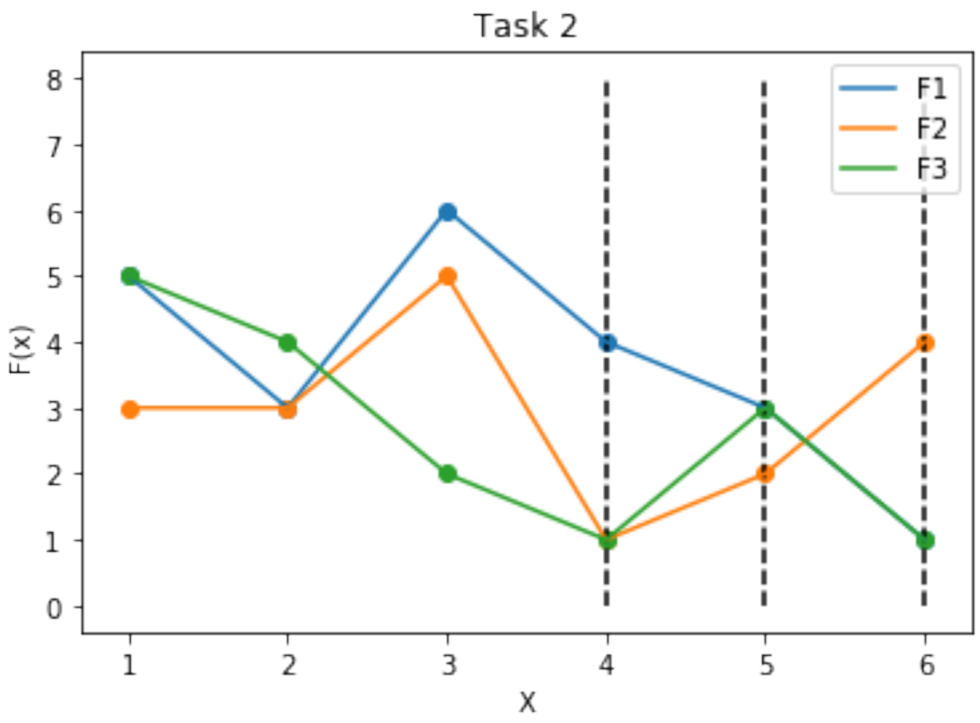
2.2 Example 2

(F1,F2,F3)→ min

x	1	2	3	4	5	6
F1	5	3	6	4	3	1
F2	3	3	5	1	2	4
F3	5	4	2	1	3	1

```
'''find min[F1, F2,F3]'''

import matplotlib.pyplot as plt

# Raw data
x = list(range(1,7))
F1 = [5,3,6,4,3,1]
F2 = [3,3,5,1,2,4]
F3 = [5,4,2,1,3,1]

# set the infomration of coord system
plt.title("Task 2")
plt.xlabel('X')
plt.ylabel('F(x)')

#plot the scatter chart
plt.scatter(x, F1)
plt.scatter(x, F2)
plt.scatter(x, F3)

# plot the line chart
plt.plot(x, F1, label="F1")
plt.plot(x, F2, label="F2")
plt.plot(x, F3, label="F3")
plt.legend(loc='upper right')
```

```
#draw the black vertical lines which stand for the Pareto Optimal


plt.vlines(4,0,8,linestyles = "dashed")
plt.vlines(5,0,8,linestyles = "dashed")
plt.vlines(6,0,8,linestyles = "dashed")


plt.show()

# solution

'''
Let P(x) =[F1,F2,F3]
it is easy to check that P(1)=[5,3,5] and P(2)=[3,3,4] are both
dominated
by P(5)=[3,2,3]

and P(3)=[6,5,2] is dominated by P(4)=[4,1,1] and P(6) = [1,4,1]

P(4) = [4,1,1],
P(5) = [3,2,3],
P(6) = [1,4,1] are not dominated by each other,thus they are Pareto
Efficient
shown by black vertical lines
'''
```



2.3 Example 3

$$f_1(x, y) = (x - 5)^2 + (y - 2)^2, x \in [-18, 18]$$

$$f_2(x, y) = x^2 + y^2, y \in [-18, 18]$$

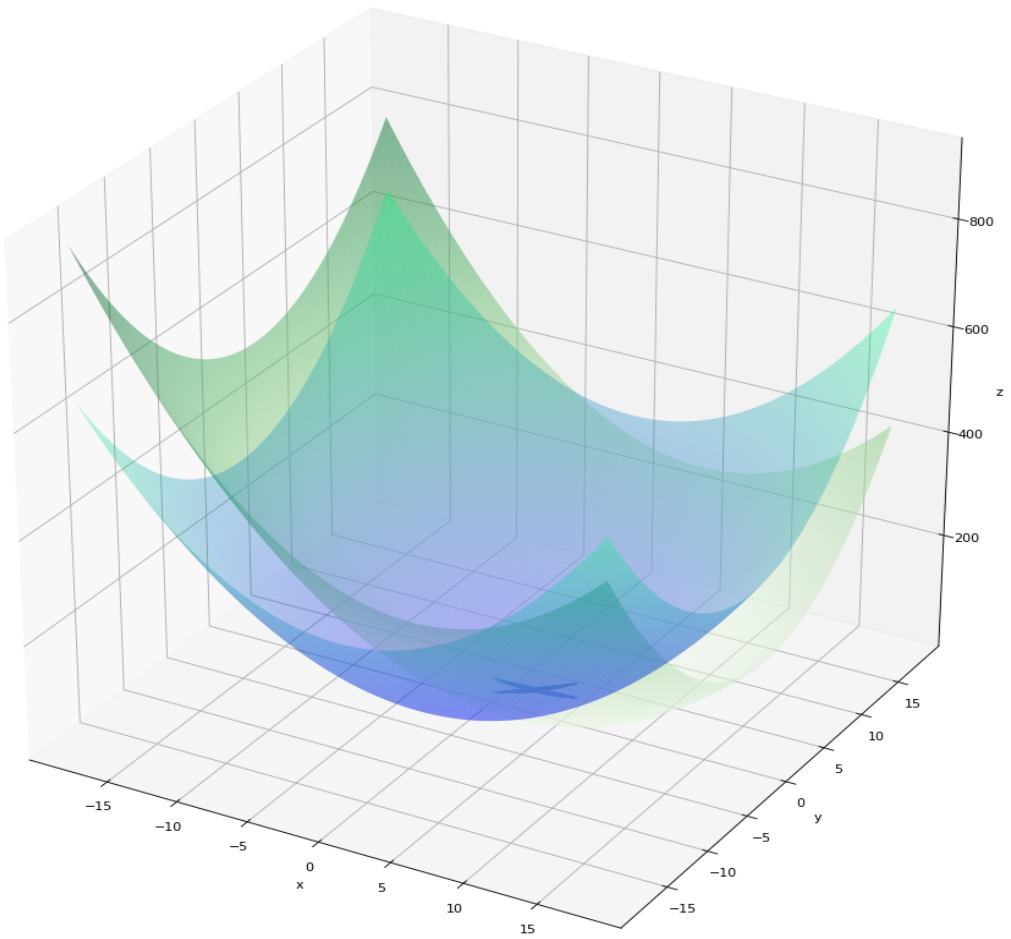
Find { f1,f2} -> min:

```
import numpy as np
import matplotlib.pyplot as plt
import mpl_toolkits.mplot3d
import random


def npmin(array):
        arrayindex = array.argmin(1)
        arrayvalue = array.min(1)
        i = arrayvalue.argmin()
        j = arrayindex[i]
        return i, j

fig = plt.figure(figsize=(16,16), dpi=80)

x,y=np.mgrid[-18:18:200j,-18:18:200j]



#f1= np.square(x) + y
f1 = np.square(x-5) + np.square(y-2)
f2 = np.square(x) + np.square(y)

w1 = [random.uniform(0,1) for i in range(1000)]
w2 = [1 - w1[i] for i in range(1000)]


x_val=[]
y_val=[]
F=[]
for i in range(1000):
        S = w1[i]*f1 + w2[i]*f2
        row,col = npmin(S)
        x_val.append(x[row][col])
        y_val.append( y[row][col])
        F.append(S.min())

X = np.array(x_val).reshape(100,10)
Y = np.array(y_val).reshape(100,10)
F1 = np.square(X-5) + np.square(Y-2)
```

```
#F1 = np.sin(X+Y)
F2 = np.square(X) + np.square(Y)


ax=plt.subplot(111,projection='3d')
ax.plot_surface(x,y,f1,rstride=2,cstride=1,cmap=plt.cm.Greens,alpha=
0.5)
ax.plot_surface(x,y,f2,rstride=2,cstride=1,cmap=plt.cm.winter,alpha=0
.3)


#ax.view_init(elev=18., azim=-170)
ax.plot_surface(X,Y,F1,rstride=2,cstride=1,cmap=plt.cm.winter,alpha=
1)
ax.plot_surface(X,Y,F2,rstride=2,cstride=1,cmap=plt.cm.Blues,alpha=
1)

#fig.colorbar(surf, shrink=0.5, aspect=5)

ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('z')
plt.show()
```

Brief idea：

- coefficient w1 and w2 are randomly generated in [0,1]
- w1 + w2 =1
- S = w1 *f1 + w2* f2
- find x, y ,and f1(x,y), f2(x,y) where S reaches its minimum
- draw the graph (x,y,f1(x,y)), (x,y,f2(x,y))
- try different coefficients to get different results



2.4 Example 4

(F1,F2)--. Min

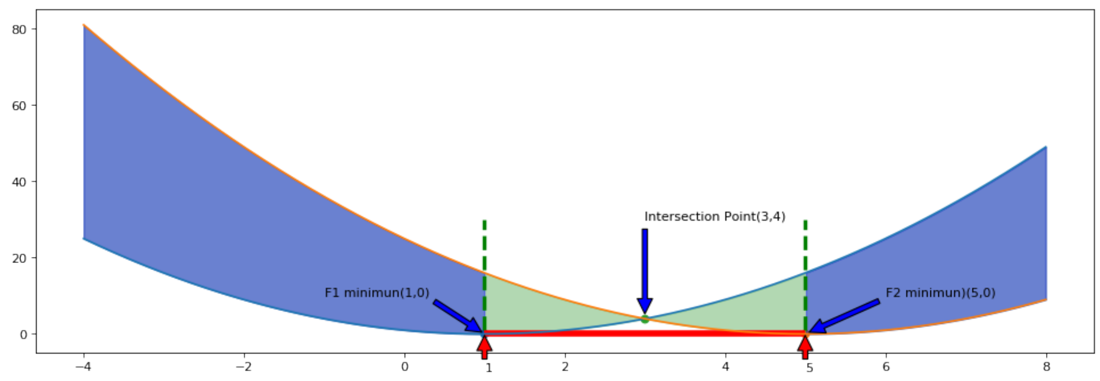
$F1(x)=(x-1)^2$

$F2(x)=(x-5)^2$

```
'''
[F1,F2] -->min
'''



import numpy as np
import matplotlib.pyplot as plt

# setting the area of the graph
plt.figure(figsize=(15,5), dpi=80)

# create points
x=np.linspace(-4,8,1000)
F1 = [np.square(i-1) for i in x]
F2 = [np.square(i-5) for i in x]

# draw the graph of F1 and F2
plt.plot(x, F1)
plt.plot(x, F2)

# Annotate the minimum of F1 and F2
plt.annotate("F1 minimun(1,0)", xy=(1,0), xytext=(-1,10),
                        arrowprops=dict(facecolor='blue', shrink=0.05))
```

```
plt.scatter(1,0)

plt.annotate("F2 minimun)(5,0)", xy=(5,0), xytext=(6,10),
                    arrowprops=dict(facecolor='blue', shrink=0.05))
plt.scatter(5,0)

plt.annotate("Intersection Point(3,4)", xy=(3,4), xytext=(3,30),
                    arrowprops=dict(facecolor='blue', shrink=0.05))
plt.scatter(3,4)




# Draw the boundary of the Pareto set
plt.vlines(1, 1, 30, color="green", linewidth=3, linestyles="dashed")
plt.vlines(5, 1, 30, color="green", linewidth=3, linestyles="dashed")



# Fill the color of intersction area to green
plt.fill_between(x, F1,F2, where= (x>=3),color = "g", alpha = 0.3)
plt.fill_between(x, F1,F2, where= (x>=5),color = "b", alpha = 0.4)

plt.fill_between(x, F1,F2, where= (x<3),color = "g", alpha = 0.3)
plt.fill_between(x, F1,F2, where= (x<=1),color = "b", alpha = 0.4)



# Indicate the range of x where belongs to Pareto Optimal
plt.hlines(0,1,5,color="red",linewidth=5)
plt.annotate("1", xy=(1,0), xytext=(1,-10),
                    arrowprops=dict(facecolor='red', shrink=0.05))
plt.annotate("5", xy=(5,0), xytext=(5,-10),
                    arrowprops=dict(facecolor='red', shrink=0.05))
plt.show()
```

```
# solution

'''when x locates between[1,5], the system[F1(x),F2(x)] has the Pareto
Optimal
Reason:
        In the interval[-∞,1],F1 and F2 are both decreasing,they are
Pareto improvement,
        the same situation as
        in the interval[5,+∞] if they come close to x=5.
        In the interval[1,5],F1 and F2 couldn't decrease itself without
increasing
        the other one.
'''
```



CHAPTER 3

METHOD OF LINEAR CONVOLUTION OF CRITERIA FOR SOLVING MULTICRITERIA PROBLEM

3.1 Introduction

The traditional method for finding efficient valuations in the n-criterion problem is the linear convolution of criteria, which can be expressed in the form of the following almost evident inclusion:

$$\Lambda(Y) \subseteq P(Y),$$

where

$$\Lambda(Y) = \bigcup_{\lambda \in \Lambda_n} \Lambda(Y, \lambda),$$

$$\Lambda_n = \left\{ \lambda \in \mathbf{R}^n : \sum_{i=1}^{n} \lambda_i = 1,\ \lambda_i > 0\ \forall i \in N_n \right\},$$

$$\Lambda(Y, \lambda) = \arg\min\{\langle \lambda, y \rangle : y \in Y\}, \quad \langle \lambda, y \rangle = \sum_{i=1}^{n} \lambda_i y_i$$

Here P(Y) is the Pareto set:【3】

$$P(Y) = \{y \in Y : \pi(y) = \emptyset\},$$

where

$$\pi(y) = \{y' \in Y : y \geq y',\ y \neq y'\},$$
$$Y = y(X) = \{y \in \mathbf{R}^n : y = y(x),\ x \in X\}$$

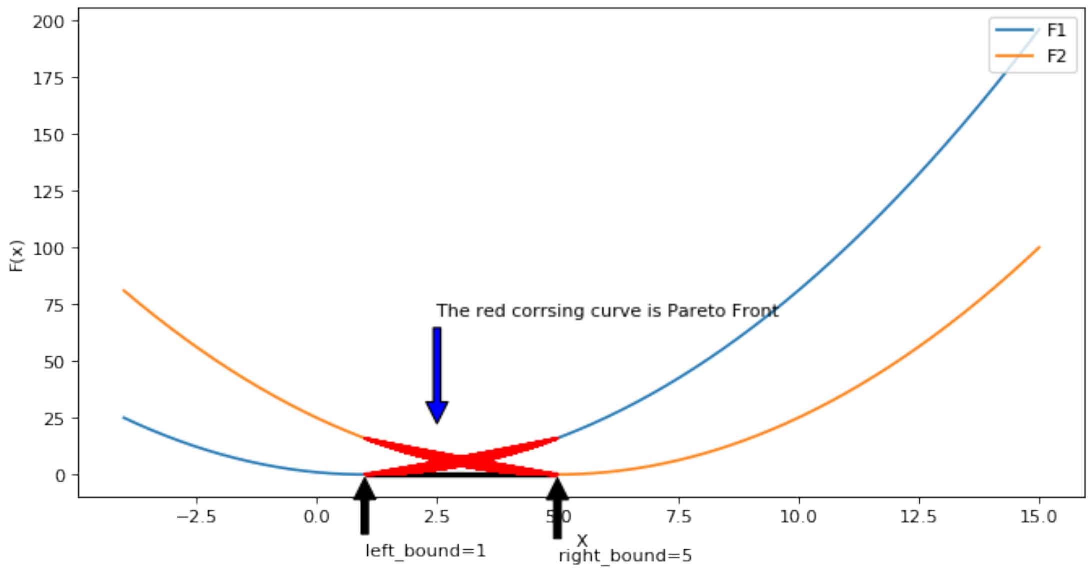
$\langle \lambda, y \rangle = \sum_{i=1}^{n} \lambda_i y_i$ is a linear convolution of the particular criteria and argmin{.} is the set of all optimal solutions of corresponding minimization problem.

3.2 Example

```
'''
To simplify problems, here we have only two criterias,
and assign coefficients to w1 and w2 between 0 and 1. By adding
w1*F1 + w2*F2 ,we denote it as S, we should try to find the minimum
of S,and get the corresponding value F1 and F2, then plot them
those F1 and F2 values will make up of Pareto Front.



Step:
        1. assign coefficients to w1,w2:
                w1,w2 >=0; w1+w2=1
        2. S = w1*F1 + w2*F2
        3. Find min(S)
        4. f1 =F1,f2 =F2 where min(S)
        5. Fetch the value of x where F(x) = f1.
        5. Store points(x,f1) and (x, f2)
        6. Repeat step 1 to 5
        7. Plot points (x,f1) and (x, f2)
'''



import numpy as np
import matplotlib.pyplot as plt
import random

# setting the area of the graph
plt.figure(figsize=(10,5), dpi=80)

# create points
'''F1 = (x-1)^2
    F2 = (x-5)^2
'''
x = np.linspace(-4,15,1000)
F1 = [np.square(i-1) for i in x]
F2 = [np.square(i-5) for i in x]



# Choose coefficients w1 and w2 where w1+w2 =1;
w1 = [random.uniform(0,1) for i in range(len(F1))]
w2 = [1 - w1[i] for i in range(len(F1))]

# Index that stores the index when the min_F get the minimum
index = []



# min_F(x) = w1*F1 + w2*F2
'''Get the minimum of min_F and store the index of F1 and F2 '''
```

```
for a in range(len(F1)):
        min_F = [w1[a]*F1[i] + w2[a]*F2[i] for i in range(len(F1))]
        index.append(min_F.index(min(min_F)))
# Fetch the value from F1 and F2
f1 = [F1[v] for v in index]
f2 = [F2[v] for v in index]

# Fetch the value of corresponding x
X = [x[i] for i in index]

# Draw the graph of (F1,F2)
plt.xlabel("X")
plt.ylabel("F(x)")
plt.plot(x, F1, label='F1')
plt.plot(x, F2, label='F2')
plt.legend(loc='upper right')



# Draw the Graph of Pareto Front


plt.plot(X, f1,color="red")
plt.plot(X, f2,color="red")
plt.annotate("The red corrsing curve is Pareto Front", xy=(2.5,20),
xytext=(2.5,70),
                    arrowprops=dict(facecolor='blue', shrink=0.05))

# The interval where the Pareto set starts and ends
left_bound = sorted(X)[0]
right_bound = sorted(X)[-1]



# Draw the boundary of Pareto set
```

```
plt.hlines(0,left_bound,right_bound ,color="black",linewidth=3)
plt.annotate("left_bound=1", xy=(1,0), xytext=(1,-35),
                    arrowprops=dict(facecolor='black', shrink=0.05))
plt.annotate("right_bound=5", xy=(5,0), xytext=(5,-37),
                    arrowprops=dict(facecolor='black', shrink=0.05))
plt.show()
```



CHAPTER 4

THE SUCCESSIVE CONCESSIONS METHOD FOR SOLVING MULTICRITERION OPTIMIZATION PROBLEMS

At the first step we determine the minimum of $\Phi_1(\alpha)$ for $\alpha \in D$. Let us denote this minimum by min Φ_1. Then a 'concession' h_1 is chosen for the criterion Φ_1 and the corresponding criterion constraint is specified:

$$\Phi_1^{**} = \min \Phi_1 + h_1 \ .$$

At the second step the minimum value of $\Phi_2(\alpha)$ is determined for $\alpha \in D$ under the constraint $\Phi_1(\alpha) \leq \Phi_1^{**}$. Upon calculating the minimum of Φ_2 and choosing a 'concession' h_2 we specify the second criterion constraint

$$\Phi_2^{**} = \min \Phi_2 + h_1 \ .$$

At the third step the minimum value of $\Phi_3(\alpha)$ is determined for $\alpha \in D$, $\Phi_1(\alpha) \leq \Phi_1^{**}$, $\Phi_2(\alpha) \leq \Phi_2^{**}$, and so on.

Finally, the minimum value of $\Phi_k(\alpha)$ is found for $a \in D$, $\Phi_1(\alpha) \leq \Phi_1^{**}$, …, $\Phi_{k-1}(\alpha) \leq \Phi_{k-1}^{**}$. If min Φ_k is attained at some point α' then this point is considered to be the best.

4.1 Introdcution【2】

4.2 Example

F = {F0, F1, F2} under the domain D and the importance : F0 > F1 > F2.

$$F_0 = (x - 1)^2$$

$$F_1 = (x - 5)^2$$

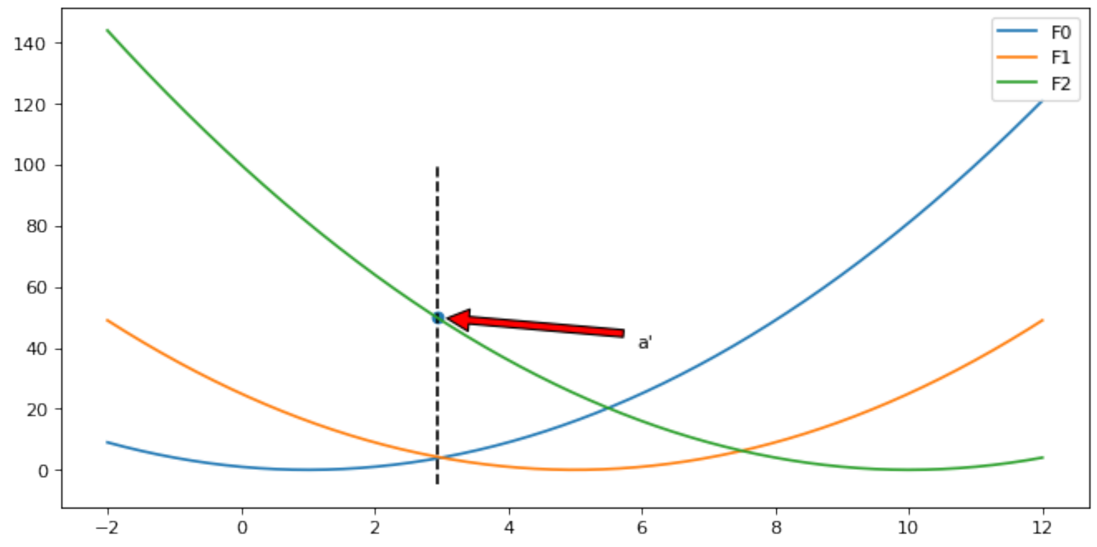
$$F_2 = (x - 10)^2$$

1. Find new domain D1 under :
   - $F0(x) < \min F0(x) + H , x \in D$
2. Find new domain D2 under :
   - $F1(x) < \min F1(x) + H , x \in D1$
3. Find minimum value a' under :
   - $a' = \min F2, x \in D2$

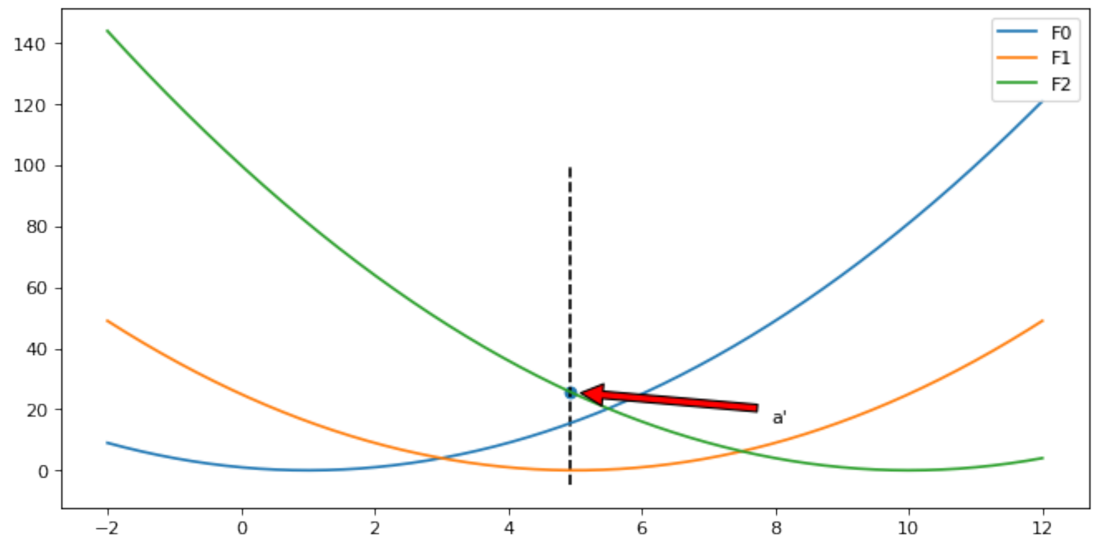
4.2.1 Solution

```
import numpy as np
import matplotlib.pyplot as plt

def minimum(f, d):
    ini_min = f[0]
    for i in range(len(d)):
        if(ini_min > f[i]):
            ini_min = f[i]
    return ini_min




# setting the area of the graph
plt.figure(figsize=(10,5), dpi=80)
plt.title("The best solution under concession H = %s" %(H))
x_index = 0

# Three Criterias
'''
Ranks: F1 > F2 > F3
'''
D=np.linspace(-2,12,1000)
Domain = D
F0 = [np.square(i-1) for i in D]
F1 = [np.square(i-5) for i in D]
F2 = [np.square(i-10) for i in D]
F =[F0,F1,F2]

# draw the graph of F1 and F2
plt.plot(D, F0, label="F0")
plt.plot(D, F1, label="F1")
plt.plot(D, F2, label="F2")
plt.legend(loc='upper right')
# Set concessions
H = 20



# Store the result
Pareto_set = []
F2_min = 0

#begin


for i in range(len(F)):
        tmp_domain = []
        for j in range(len(D)):
            if(F[i][j] < minimum(F[i], D)+H):
                tmp_domain.append(j)
        D =tmp_domain

        if(i==1):
                F2_min = minimum(F2,D)
                break



       # fetch the corresponding value of Domain D
Pareto_set.append(Domain[F2.index(F2_min)])
x_index = F2.index(F2_min)






#Display the Pareto Front
point_x = Pareto_set[0]
plt.scatter(point_x,F2[x_index])
plt.vlines(point_x,-5,100,linestyles = "dashed")

print("a'=%f" %(point_x))
plt.annotate("a'", xy=(point_x,F2[x_index]),
xytext=(point_x+3,F2[x_index]-10),
                arrowprops=dict(facecolor='red', shrink=0.05))
plt.show()
```

The best solution under concession H = 20



The best solution under concession H = 30



CHAPTER 5

EXAMPLES WITH MULTI-EXTREMAL FUNCTIONS

System F ={F1,F2}, where x ∈ $[-6, 6]$,Denote domain as D.

$$F_1(x) = 15sin5x + 10cos3x$$

$$F_2(x) = x^2 + x + 5$$

Rank: F1 > F2

1. Find minimum of F1 under domain D:
   - denote it as $minF_1$
2. Set concession H
3. Find new Domain D1 under:
   - F1(x) < $minF_1(x)$ + H, where x in D
4. Find minimum of F2 under this new domain D1:
   - Denote a' as $minF_2$

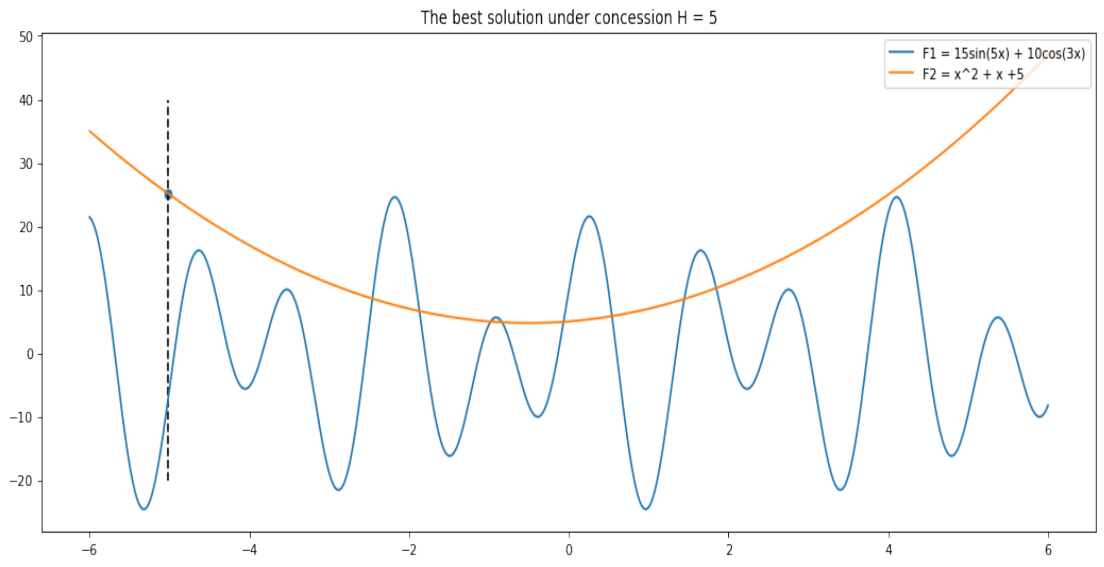
5.1 Task

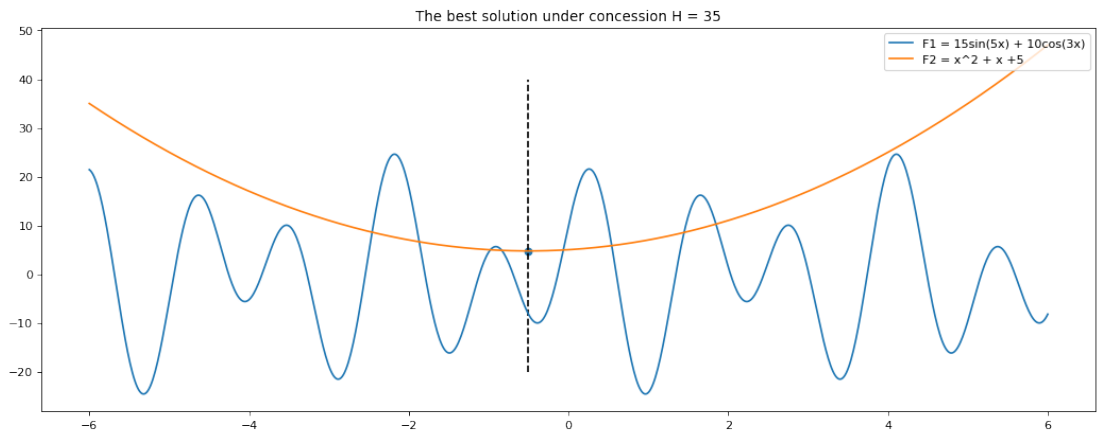
5.1.1 Solution

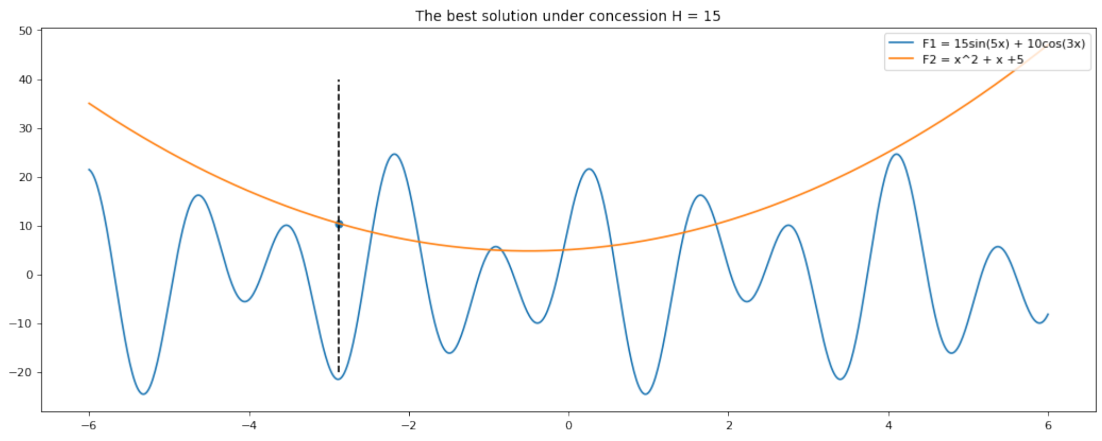
```
import numpy as np
import matplotlib.pyplot as plt

# setting the area of the graph
plt.figure(figsize=(16,6), dpi=80)




# create points
Domain = np.linspace(-6,6,800)
```

```
F1 = [15*np.sin(5*x) + 10*np.cos(3*x) for x in Domain]
F2 = [np.square(x)+x+5 for x in Domain]

# Plot the function
plt.plot(Domain, F1, label="F1 = 15sin(5x) + 10cos(3x)")
plt.plot(Domain, F2, label="F2 = x^2 + x +5")
plt.legend(loc='upper right')

# Find the minimum of function F
def minimum(f, d):
        ini_min = f[0]
        for i in range(len(d)):
                if(ini_min > f[i]):
                        ini_min = f[i]
        return ini_min



# Begin

# Step 1
minF1 = minimum(F1,Domain)

#set concession H
plt.title("The best solution under concession H = 5")
H = 5

#Find new Domain D1:
D1 = []
for i in range(len(Domain)):
        if(F1[i] < minF1 + H):
                D1.append(Domain[i])
```

```
minF2 = minimum(F2,D1)
X = Domain[F2.index(minF2)]



# Plot the best solution a'
plt.scatter(X, minF2)
plt.vlines(X,-20,40,linestyles='dashed')
plt.show()
```




The best solution under concession H = 35
F1 = 15sin(5x) + 10cos(3x)
F2 = x^2 + x +5
50
40
30
20
10
0
−10
−20
−6
−4
−2
0
2
4
6


The best solution under concession H = 15
F1 = 15sin(5x) + 10cos(3x)
F2 = x^2 + x +5
50
40
30
20
10
0
−10
−20
−6
−4
−2
0
2
4
6

CONCLUSION

In this report, the basic methods for solving multicriteria problems are studied, first introduce the concept of Pareto set which is the very important in Optimization field, then by giving several simple examples to show the Pareto set. In the method of linear convolution, the different coefficients (weight) assigned to the functions indicate the importance of that function. By concatenating the criterions into one system, we may build a more straight ,easy evaluated system. In the method of succession concession, the importance of criterions also matters, the big idea is to cover the best cases in all criterions but ordered by their importance, also. The concession h will also influence the result. By increasing the concession h, we may get a better result.

REFERENCE

【1】
http://econc10.bu.edu/GENEC/Basics/Efficiency/pareto_frontier.htm

【2】M. Ya. Marko, H. H. Tsehelyk　Using the method of seccessive concessions for solving the problem of increasing profitability of small enterprises. Scientific Paper. № 1(54). Pages 141-146,

【3】V. D. Nogin, “Linear convolution in multicriteria optimization,” Iskusstv. Intellekt Prinyat. Reshen., No. 4 (2014).

【4】https://en.wikipedia.org/wiki/Pareto_efficiency