Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра информатики

Отчет по лабораторной работе №6  
Задача квадратичного программирования.

Выполнил:

cтудент гр. 953505

Басенко К. А.

Руководитель:

доцент

Алёхина А. Э.

Минск 2022

Постановка задачи

Задача квадратичного программирования с *n* переменными и *m* ограничениями можно сформулировать следующим образом.

Дано:

* вещественный *n*-мерный вектор *c*,
* *n* × *n*-мерная вещественная симметричная матрица *D*,
* *m* × *m*-мерная вещественная матрица *A*
* вещественный *m*-мерный вектор *b*,

Целью задачи квадратичного программирования является поиск n-мерного вектора *x*, который минимизирует , при , где *x’* обозначает транспонированный вектор.

Краткие теоретические сведения

*Задача квадратичного программирования* в канонической форме состоит в минимизации квадратичной функции

,

при линейных ограничениях

.

Здесь – -матрица со столбцами x и *c* – *n*-векторы, *b* – *m*-вектор*, D* – симметричная (*D=D'*) положительно полуопределенная (*D³0* ) -матрица.

Пусть известны параметры *c*, *b*, *A*, *D*, определяющие задачу. Для реализации итерации метода необходимо знать текущий правильный опорный план {*x, Jоп , J\**}.

Опишем итерацию метода по шагам.

*Шаг 1.* Сформируем вектор и подсчитаем вектор потенциалов



и оценки

, *jÎJH = J \ J\*.*

*Шаг 2*. Если ³ *0, j Î JН*, то STOP: вектор *x* – решение задачи. В противном случае перейдем к шагу 3.

*Шаг 3*. Выберем индекс *j0 Î JН*, для которого < 0.

*Шаг 4.* По заданным множеству и индексу сформируем систему и найдем ее решение (*l\*, y*) , *l\* = (lj, j Î J\*),* *. Положим* *.*

*Шаг 5*. Вычислим

*= - xj / lj* при *lj < 0,* *= ¥* при *lj ³ 0, j Î J\*;*

*= |* */d |* при *d ¹ 0,* *= ¥* при *d = 0*.

Найдем = *min* *, j Î J\***j0*. Если = *¥,* то STOP: задача не имеет решения в силу неограниченности снизу целевой функции на множестве планов. В противном случае выберем индекс *j\* Î J\** *j0*, для которого , и перейдем к шагу 6.

## *Шаг 6.* Построим новый план по правилам:



*Шаг 7.* Предполагается, что множество индексов *Jоп* имеет вид *Jоп = {j1, j2, ..., js, ..., jm}.* Возможны случаи:

а) *j\* = j0;*

б) *j\* Î J\* \ Jоп;*

в) *j\* = js Î Jоп* и существует такой индекс *j+ Î J\* \ Jоп*, что *es¢Aоп-1Aj+ ¹ 0;*

г) *j\* = js Î Jоп* и *es¢Aоп-1Aj = 0, j Î J\* \ Jоп ,* либо *J\* = Jоп*.

В случае а полагаем

*= Jоп* , = *J\** *j0*

и переходим к новой итерации, т.е. к шагу 1, используя найденные данные 

В случае б полагаем

= *Jоп ,* = *J\* \ j\*,* 

и переходим к шагу 4, используя далее данные вместо .

В случае в полагаем

= *(Jоп \ j\*)* *j+,* = *J\* \ j\*,*

и переходим к шагу 4, используя обновленные данные  вместо .

В случае г полагаем

*= (Jоп \ j\*)* *j0,* = (*J\* \ j\*)* *j0*

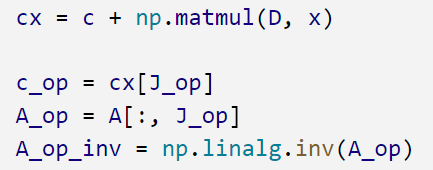
и переходим к новой итерации (т.е. к шагу 1), используя новые данные 

Программная реализация

Решение задачи квадратичного программирования методом опорных планов реализована в функции *quadratic\_programming\_problem* на вход в которую идут: параметры *A*, *b*, *c*, *D*, текущий базисный опорный план {*x, Jоп , J\**}.



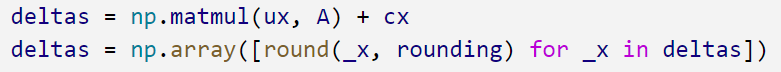
Сформируем вектор :



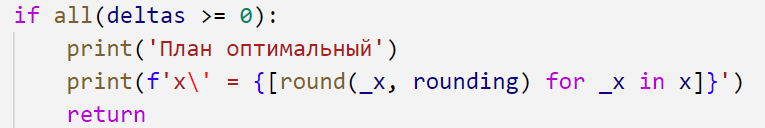
Подсчитаем вектор потенциалов:



Подсчитаем вектор оценок:



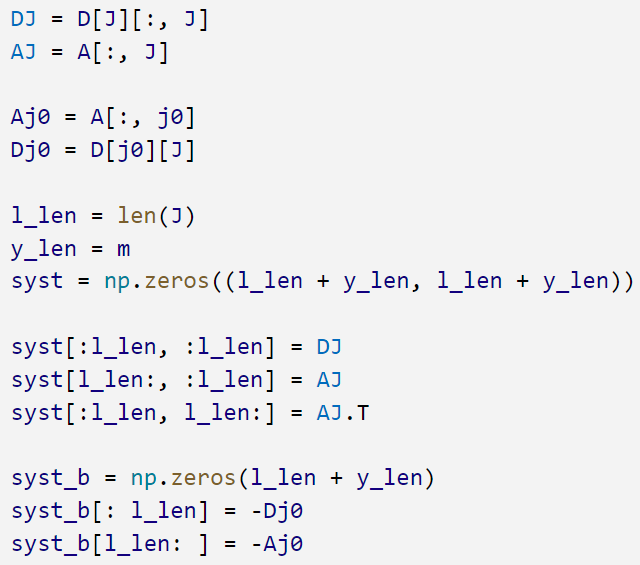
Проверяем условие оптимальности:



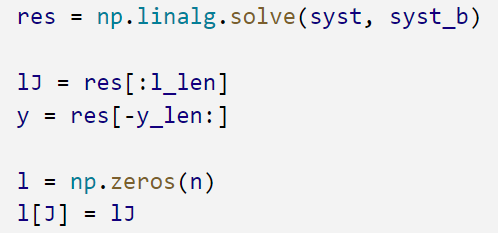
Запоминаем индекс, элемент которого не удовлетворил условие оптимальности:



Составим систему и найдем *l\**, *y*: 



Решим ее:

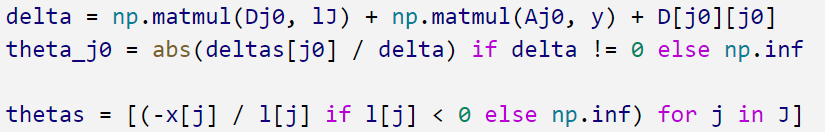


Вычислим

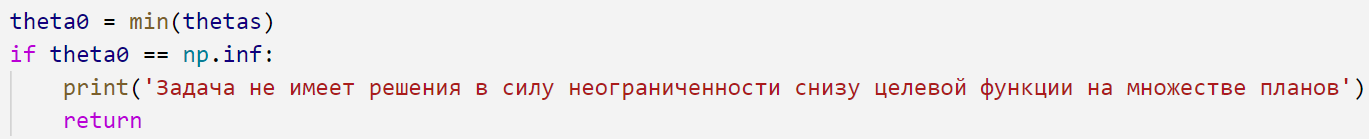


*= - xj / lj* при *lj < 0,* *= ¥* при *lj ³ 0, j Î J\*;*

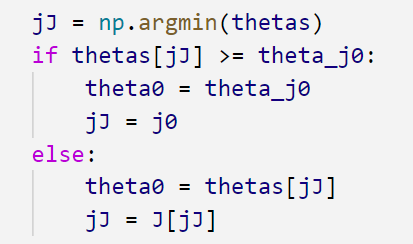
*= |* */d |* при *d ¹ 0,* *= ¥* при *d = 0*:



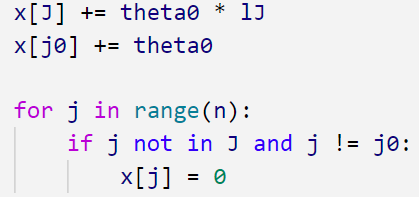
Проверяем ограниченность снизу целевой функции:



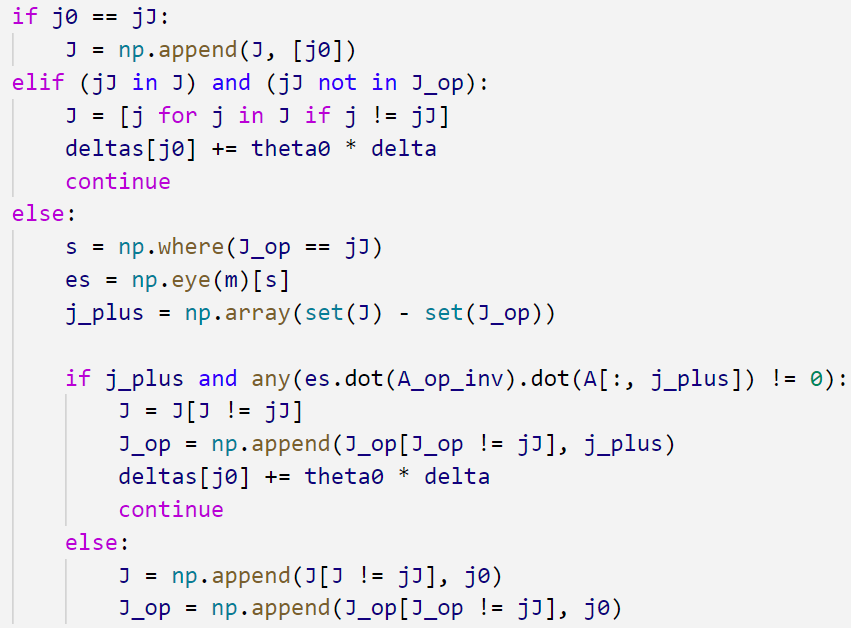
Выберем индекс *j\* Î J\** *j0*, для которого :



Строим новый план:



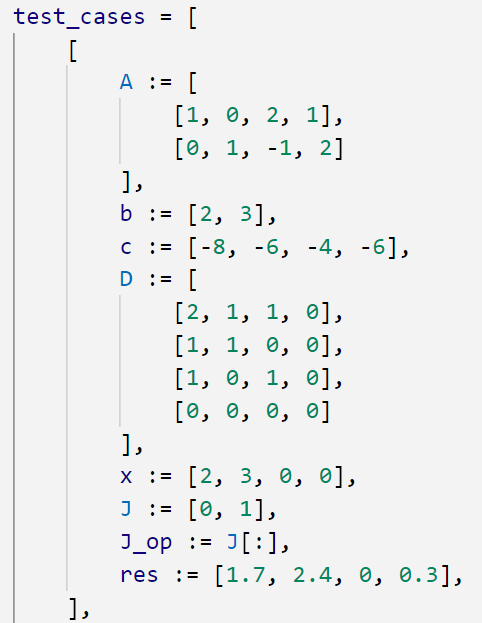
Обновляем множество индексов:

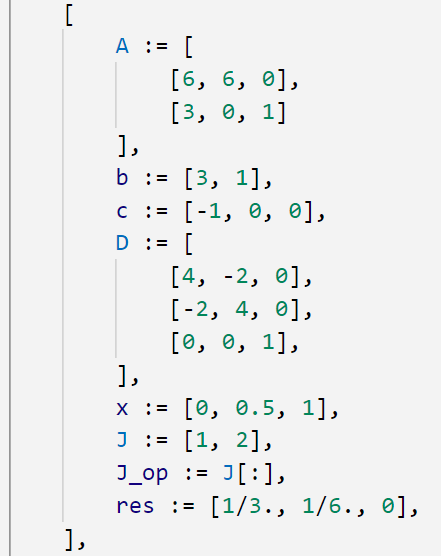


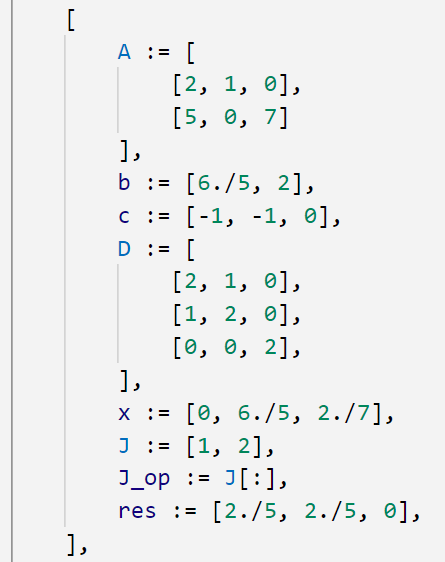
Далее повторяем вычисления с новым множеством индексов и планом.

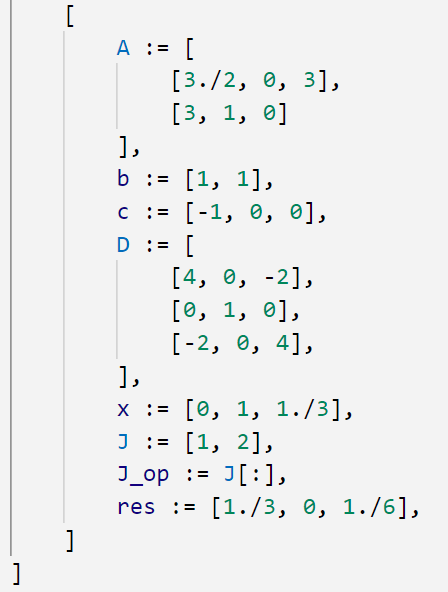
Тестовые примеры

Данные для тестирования:

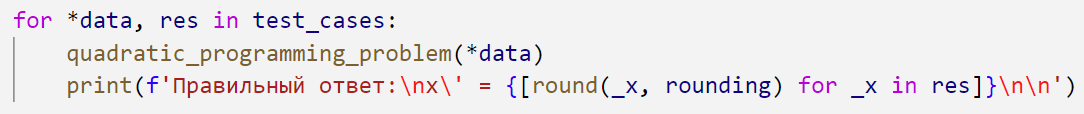








Выполняется метод с предоставленными данными, выводится полученный результат, выводится ожидаемый результат:



Вывод, соответствующий предоставленным входным данным:

