УДК 519.6

РЕШЕНИЕ ЗАДАЧ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ В ПАКЕТЕ MAPLE

О.А. Широкова[1]

1 *oshirokova@mail.ru*; Казанский (Приволжский) федеральный университет

В статье представлены некоторые разделы практикума по теме «Решение задач линейного программирования в СКМ Maple». Рассматриваются особенности визуализации решения задач линейного программирования в пакете Maple.

Ключевые слова: задачи линейного программирования, оптимизация, целевая функция, ограничения, графическое решение, визуализация решения в Maple.

В пакете *simplex* системы компьютерной математики Maple имеется небольшой, но достаточно представительный набор функций для решения задач линейного программирования:

```
>  with(simplex);
```

Главными из этих функций являются *maximize* и *minimize*, оптимизирующие задачу симплекс-методом. Они записываются в следующих формах:

```
maximize(f, C)
minimize(f, C)
minimize(f , C, vartype)
maximize(f , C, vartype)
maximize(f , C, vartype, 'NewC', 'transform')
minimize(f , C, vartype, 'NewC', 'transform')
```

Здесь f — линейное выражение, Ñ — множество или список условий, *vartype* — необязательно задаваемый тип переменных *NONNEGATIVE* или *UNRESTRICTED*, *NewC* и *transform* — имена переменных, которым присваиваются соответственно оптимальное описание и переменные преобразования. Ниже даны примеры применения этих функций при решений задач линейного программирования

```
>  minimize(x+y, {4*x+3*y <= 5, 3*x+4*y <= 4}, NONNEGATIVE);
{x= 0, y = 0}
>  maximize(x+y, {4*x+2*y <= 10, 3*x+4*y <= 16}, NONNEGATIVE);
{x= 4/5, y = 17/5}
>  maximize(x+y, {3*x+2*y <= 5, 2*x+4*y <=4});
{x= 3/2, y = 1/4}
```

Приведем пример задания целевой функции и ограничений с помощью идентификаторов z и $cnts1$:

```
> z := 2*x1 - x2 + 3*x3:
> cnts1 := [x2+2*x3 <= 1, 2*x1-4*x2+6*x3 <= 3, -x1+3*x2+4*x3 <= 12]:
> sol1 := maximize(z,cnts1,NONNEGATIVE);
 sol1 := {x1= 7/2, x2 = 1, x3 = 0}
```

Решим задачу линейного программирования симплекс методом в СКМ Maple с подстановкой оптимального решения в целевую функцию.

Приведем пример задачи минимизации целевой функции:

$$z = x_1 + 3x_2 + x_3 \xrightarrow{min} x_1 + 4x_2 + x_3 <= 12;$$
$$3x_1 - 2x_2 + x_3 >= 6;$$
$$x_1 >= 0,\ x_2 >= 0,\ x_3 >= 0.$$

```
> restart; with(simplex);
> z:=x1+3*x2+x3:
> minimize(z,{z1+4*x2+3*x3<=12,3*x1-2*x2+x3>=6},NONNEGATIVE);
{x1= 2, x2 = 0, x3 = 0, z1 = 0}
> subs(%,z);
2
```

Ответ: (2,0,0), z_{min}=2.

Приведем пример задачи максимизации целевой функции:

$$z = x_1 + 3x_2 + x_3 \xrightarrow{max} z = x_1 + 3x_2 + x_3;$$
$$3x_1 - 2x_2 + x_3 >= 6;$$
$$x_1 >= 0,\ x_2 >= 0,\ x_3 >= 0.$$

```
>  restart:with(simplex):
> z:=x1+3*x2+x3:
> inits:=[x1+4*x2+x3<=12,3*x1-2*x2+x3>=6,x1>=0,x2>=0,x3>=0]:
> maximize(z,inits,NONNEGATIVE);
{x1= 0, x2 = 0, x3 = 12}
> subs(%,z);
12
```

Ответ: (0,0,12), z_{max}=12.

В систему Maple был добавлен новый пакет оптимизации *Optimization*, основанный на новейших существенно улучшенных алгоритмах оптимизации. С его помощью можно решать не только задачи линейного, но и квадратичного и нелинейного программирования с повышенной степенью визуализации.

Пакет оптимизации *Optimization* вызывается как обычно:

```
>with(optimization);
```

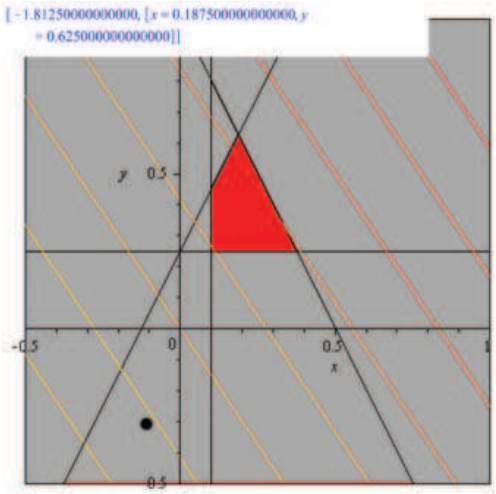
Для решения задач линейного программирования в пакете *Optimization* введена функция:

```
LPSolve(obj [, constr, bd, opts])
```

Она имеет следующие параметры: *obj* — алгебраическое выражение, целевая функция; *constr* — множество или список линейных ограничений; *bd* — последовательность вида *name = range*, задающая границы одной или многих переменных; opts — равенство или равенства в форме *option=value*, где *option* - одна из опций *assume*, *feasibilitytolerance*, *infinitebound*, *initialpoint*, *iterationlimit* или *maximize*, специализированных для команды *LPSolve*.

Рассмотрим примеры решения задач линейного программирования с графической визуализацией решения. Пусть оптимизируется целевая функция $z = -3x - 2y$, которая линейно зависит от переменных x и y. В этих примерах интересна техника графической визуализации решения.

```
> z:=-3*x-2*y:
> init:=[y<=2*x+1/4,y<=-2*x+1,x>=0.1,y>=0.25]:
> p1:=inequal(init,x=-0.5..1,y=-0.5..1,optionsfeasible=(color=red));
> p2:=contourplot(z,x=-0.5..1,y=-0.5..1);
> display([p1,p2]);
```




```
> LPSolve(z,init);
[-1.81250000000000,[x=0.18750000000000,y=0.62500000000000]]
```

Рассмотрим решение задачи линейного программирования с четырьмя переменными. Сведем ее к задаче с двумя переменными и затем решим графическим методом.

$$z = x_1 + x_2 + 3x_3 + 4x_4 \xrightarrow{min} 5x_1 - 6x_2 + x_3 - 2x_4 = 2;$$
$$11x_1 - 14x_2 + 2x_3 - 5x_4 = 2;$$
$$x_1 >= 0\,, x_2 >= 0,\ x_3 >= 0,\ x_4 >= 0.$$

Так как число переменных в задаче равно 4, в исходной постановке задача графическим методом не решается. В данном случае можно свести исходную задачу к задаче с двумя переменными. Рассмотрим систему ограничений задачи. Выразим какие-либо две переменные через остальные две переменные:

$$x_3 = 6 - 3x_1 + 2x_2;$$
$$x_4 = 2 + x_1 - 2x_2.$$

Нашли искомые выражения. Подставляем их в целевую функцию

$$z = x_1 + x_2 + 3(6 - 3x_1 + 2x_2) + 4(2 + x_1 - 2x_2) = -4x_1 - x_2 + 26.$$

Так как по условию задачи $x_3, x_4 >= 0$ получаем ограничения

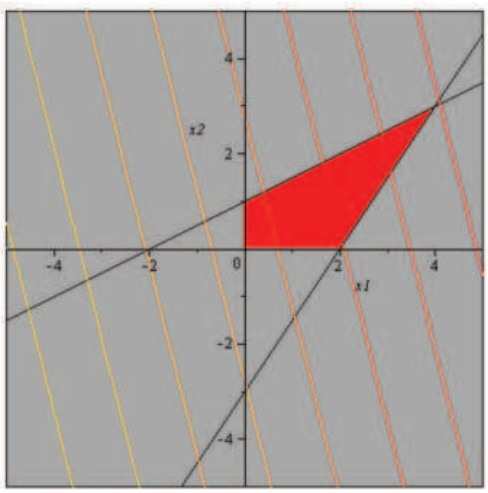
$$x_3 = 6 - 3x_1 + 2x_2 >= 0;$$
$$x_4 = 2 + x_1 - 2x_2 >= 0.$$

Приходим к задаче линейного программирования с двумя переменными

$$z = -4x_1 - x_2 + 26 \xrightarrow{min} 3x_1 + 2x_2 <= 6;$$
$$11x_1 - 14x_2 + 2x_3 - 5x_4 = 2;$$
$$x_1 >= 0,\ x_2 >= 0.$$

Решим задачу графически, а затем симплекс методом.

```
> restart:
> with(plots):with(Optimization):
> z:=-4*x1-x2+26:
> init:=[3*x1-2*x2<=6,-x1+2*x2<=2,x1>=0,x2>=0]:
>p1:=inequal(init,x1=5..5,x2=5..5,optionsfeasible=(color=red));
> p2:=contourplot(z,x1=-5..5,x2=-5..5);
> display([p1,p2]);
```




```
> with(simplex);
> minimize(z,init);
{x1= 0, x2 = 0, x3 = 12}
> subs(%,z);
7
> LPSolve(z,init);
[7.,[x1= 4., x2 = 3.]]
```

Таким образом, получено решение задачи линейного программирования двумя способами : симплекс методом из пакета simplex и командой LPSolve из пакета Optimization.

PROBLEM SOLVING LINEAR PROGRAMMING IN MAPLE PACKAGE

O.A. Shirokova

The article presents some sections of the workshop on "The solution of the linear programming problems CAS Maple". The features of the visualization solving linear programming in Maple package.
Keywords: the problem of linear programming, optimization, objective function, constraints, graphical solution, visualization solutions in Maple.

УДК 519.6

СОЗДАНИЕ ОБЪЕКТОВ ФРАКТАЛЬНОЙ ГРАФИКИ СРЕДСТВАМИ DELPHI

О.А. Широкова[1]

1 *oshirokova@mail.ru*; Казанский (Приволжский) федеральный университет

В статье представлены некоторые разделы практикума по курсу «Фрактальная графика». Показана эффективность использования визуальных компонентов интегрированной среды разработки Delphi при демонстрации проектов, посвященных фрактальной графике.

Ключевые слова: фрактальная графика, объектно-ориентированное программирование, визуальный проект.

В статье представлены некоторые разделы практикума по курсу «Фрактальная графика». В основе данного курса лежит проектирование программного продукта, при помощи которого пользователю предоставляется возможность наглядно увидеть не только изображение, но и всю красоту фрактальной графики ([1]-[2]).

Одно из важнейших требований в данных проектах это простота интерфейса. Показана эффективность использования визуальных компонентов ([3]-[4]) интегрированной среды разработки Delphi при демонстрации проектов, посвященных фрактальной графике. В настоящее время объектно-ориентированный стиль применяется при разработке широкого круга приложений ([3]-[4]).

В данной статье рассматривается создание объектов фрактальной графики средствами Delphi на примере построения аттракторов Клиффорда и Лоренца.

Аттракторы — это множества, к которым приближаются точки при последовательных итерациях отображения. Если мы хотим найти аттрактор, то нам не нужно вычислять эти итерации и анализировать наше отображение. Аттрактор Лоренца, как и аттрактор Клиффорда, строятся по своим установленным координатам, иначе они перестают быть таковыми.

Построение аттрактора Клиффорда

Аттрактор задается базовыми уравнениями:

$$x = \sin(ay) + c\cos(ax);$$
$$y = \sin(bx) + d\cos(by),$$

где $a = 1,5$, $b = -1,8$, $c = 1,6$, $d = 0,9$.