

Практическая работа по дисциплине

Теория принятия решений

Вводное занятие 1

Пономарев Андрей Васильевич

Организационное

- Много информации, имеющей отношение к практике по курсу:
<http://avponomarev.bitbucket.io> (далее просто Сайт)
- Три задачи – как правило, две ЛП, одна ДП
 - Найти решение задачи оптимизации
 - Ответить на вопросы по заданию
- Варианты индивидуальных заданий
 - Сами задания лежат на Сайте
 - Для назначения вариантов необходимо **отправить мне на почту списки групп**
- Отчетность
 - Отчет
 - Формализация – решение – интерпретация – ответы на доп. вопросы
 - Требования по содержанию и оформлению (и шаблон) – на Сайте
 - Защита
 - Могу попросить изменить какое-нибудь условие, что-то добавить и т.п.

Организационное. Режим

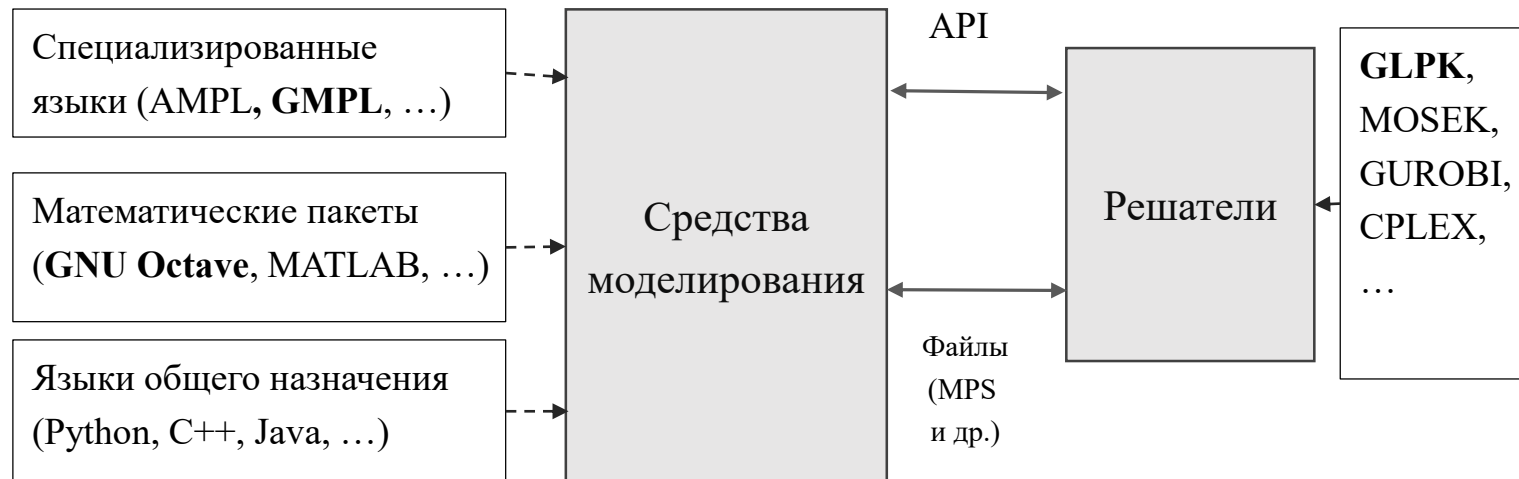
- Лекции – Николай Алексеевич
- Практики/лабораторные – Андрей Васильевич (то есть, я)
- Фазы нашего взаимодействия:
 1. Введение (~до конца октября - середины ноября)
 - Я рассказываю о том, как предполагается решать задачи
 2. Развитие
 - Консультации, прием
 3. Прием (последнее занятие)

Программное обеспечение

- Инструменты:
 - Рекомендуемый инструмент: Python (Jupyter, CVXOPT, seaborn, pandas etc.)
 - Возможно (legacy mode):
 - Для задачи динамического программирования - Octave/C++/Python/Java
 - Для линейного программирования – Octave (glpk) или GLPK (через GUSEK)
- Почему это так?
 - Вы решали задачи линейного программирования, транспортные задачи, "руками" в соответствующих курсах
 - В рамках практических работ будет полезно познакомиться с тем, что представляет собой мир **ПО для оптимизации** и "пощупать" его руками
 - Python. Набирающая популярность инфраструктура для *анализа данных*
 - + язык достаточно широкого назначения (чем выгодно отличается от Octave/R и пр.)

Программное. Пояснения

- В мире оптимизации существует два класса приложений:
 - 1) **программные средства для моделирования** - с их помощью вы на понятном, удобном синтаксисе записываете математическую модель вашей задачи.
 - 2) **солверы**. Они либо встраиваемые (линкуются как библиотеки), либо считывают файлы какого-либо из проприетарных или распространенных форматов (mps).



План

- Постановка задачи линейного программирования (ЗЛП) (почти как в индивидуальных заданиях)
- Решение ЗЛП в Python
- Решение ЗЛП в GNU Octave
- Решение ЗЛП в GLPK/MathProg

Постановка задачи линейного
программирования (почти как в
индивидуальных заданиях)

Условие

- Рацион стада крупного рогатого скота включает **пищевые продукты А, В, С, D, Е.**
- В сутки одно животное должно съесть **не менее А1 [кг] продукта А, В1 [кг] продукта В, С1 [кг] продукта С, D1 [кг] продукта D, Е1 [кг] продукта Е.** Однако в чистом виде указанные продукты не производятся.
- Они содержатся в кормовых культурах **К-1, К-2, К-3, К-4.** Известно **процентное содержание** продуктов в **килограмме** кормовой культуры.
- Известны **суточные нормы А1-Е1 (в кг).**
- Определить **максимальное поголовье скота,** которое можно содержать, используя имеющийся запас культур.

Решение

- *Параметры*

- Будем использовать i для индексирования продуктов, j – для индексирования культур.
- Процентное содержание (r_{ij}) , $i \in \{1..5\}$, $j \in \{1..4\}$.
- Суточные нормы продуктов d_i , $i \in \{1..5\}$ [кг].
- Запас культур s_j [т].

- *Переменные*

- n – количество животных.
- $x_j \in \{1..4\}$ – количество культуры j -того типа, которую мы используем в кормовом плане [т].

- *Тогда*

- (ЦФ) $n \rightarrow \max$
- $x_1 r_{11} + x_2 r_{12} + x_3 r_{13} + x_4 r_{14} \geq 0,365 d_1 n$ - слева количество первого продукта
- ...
- $x_1 \leq s_1$ (согласовать единицы измерения!)

Линейные задачи. Таблица

	x1	x2	x3	x4	n	Неравенство	Правая часть
f	0	0	0	0	1	-	max
y1	r_{11}	r_{12}	r_{13}	r_{14}	$-0,365 \cdot d_1$	\geq	0
...
y6	1	0	0	0	0	\leq	s_1
...	\leq	...

Решение задачи ЛП в Python

Конфигурирование среды

1. Установить менеджер пакетов, зависимостей и окружений – Conda (Miniconda)
 - Особенно важно для Windows
 - Вариант: весь дистрибутив Anaconda (но это overkill)
2. Сконфигурировать окружение:

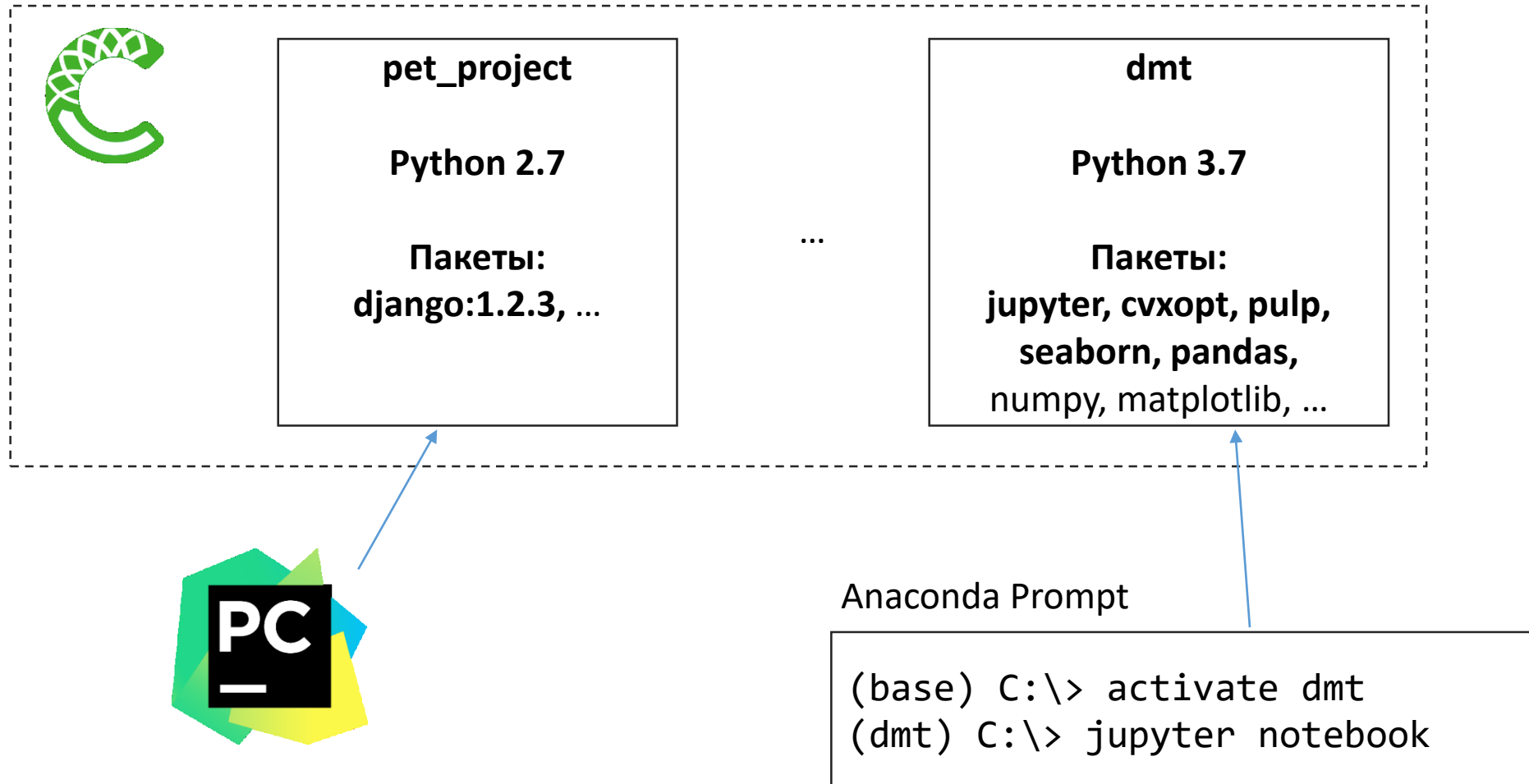
```
conda create --name dmt python=3.7
conda activate dmt
conda install jupyter cvxopt seaborn pandas
jupyter notebook
```

или
 - создать проект в своей любимой IDE для Python (Pycharm?) указав, что он должен выполняться в соответствующем окружении

Больше информации на <https://conda.io/>



Менеджер пакетов, зависимостей и окружений Conda



Больше информации на <https://conda.io/>

Docker

```
$ docker pull jupyter/minimal-notebook
```

```
$ docker run -it --rm -p 8888:8888 jupyter/minimal-notebook
```

Внутри контейнера:

```
$ conda install cvxopt pulp seaborn pandas
```

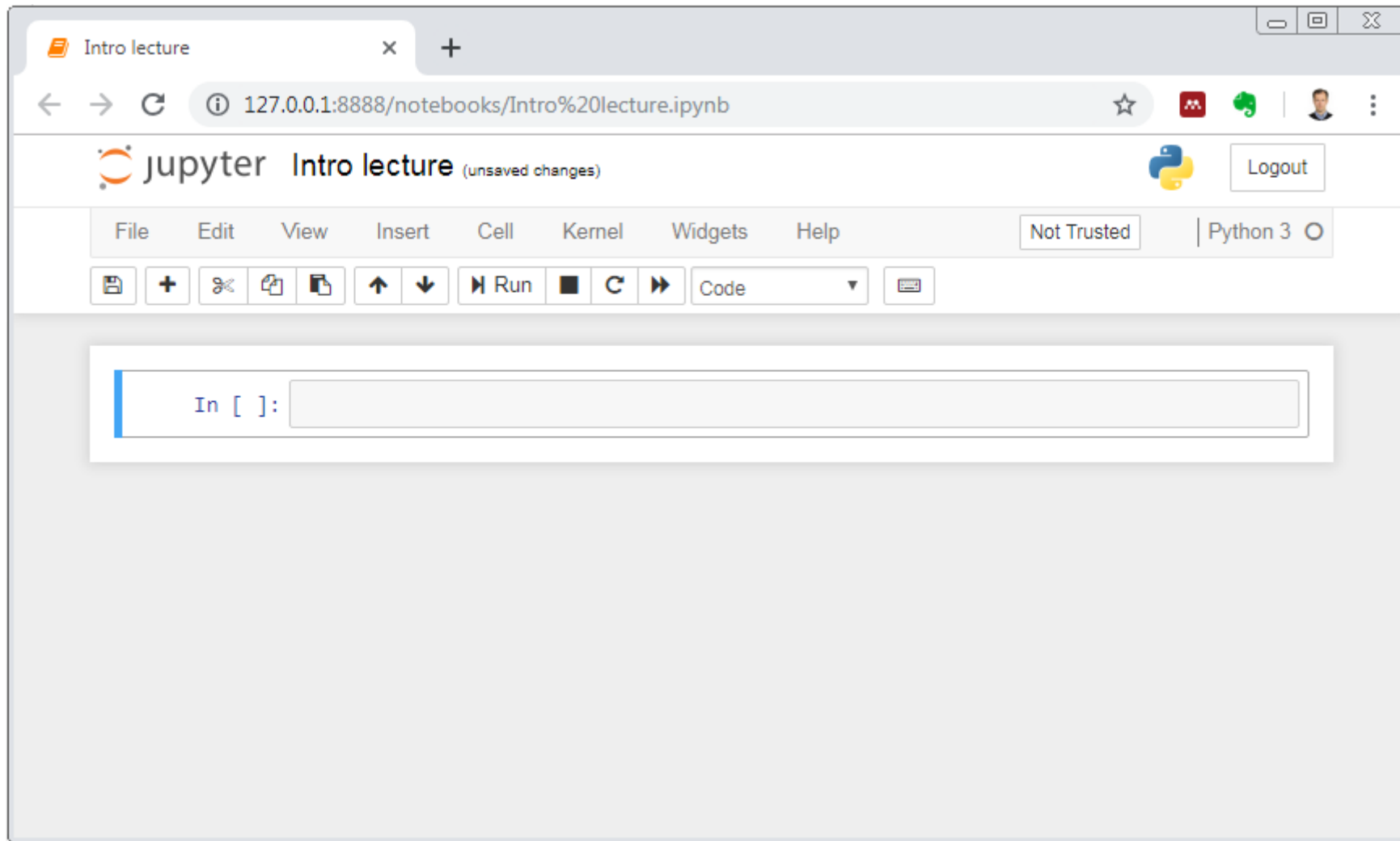
Google Colaboratory



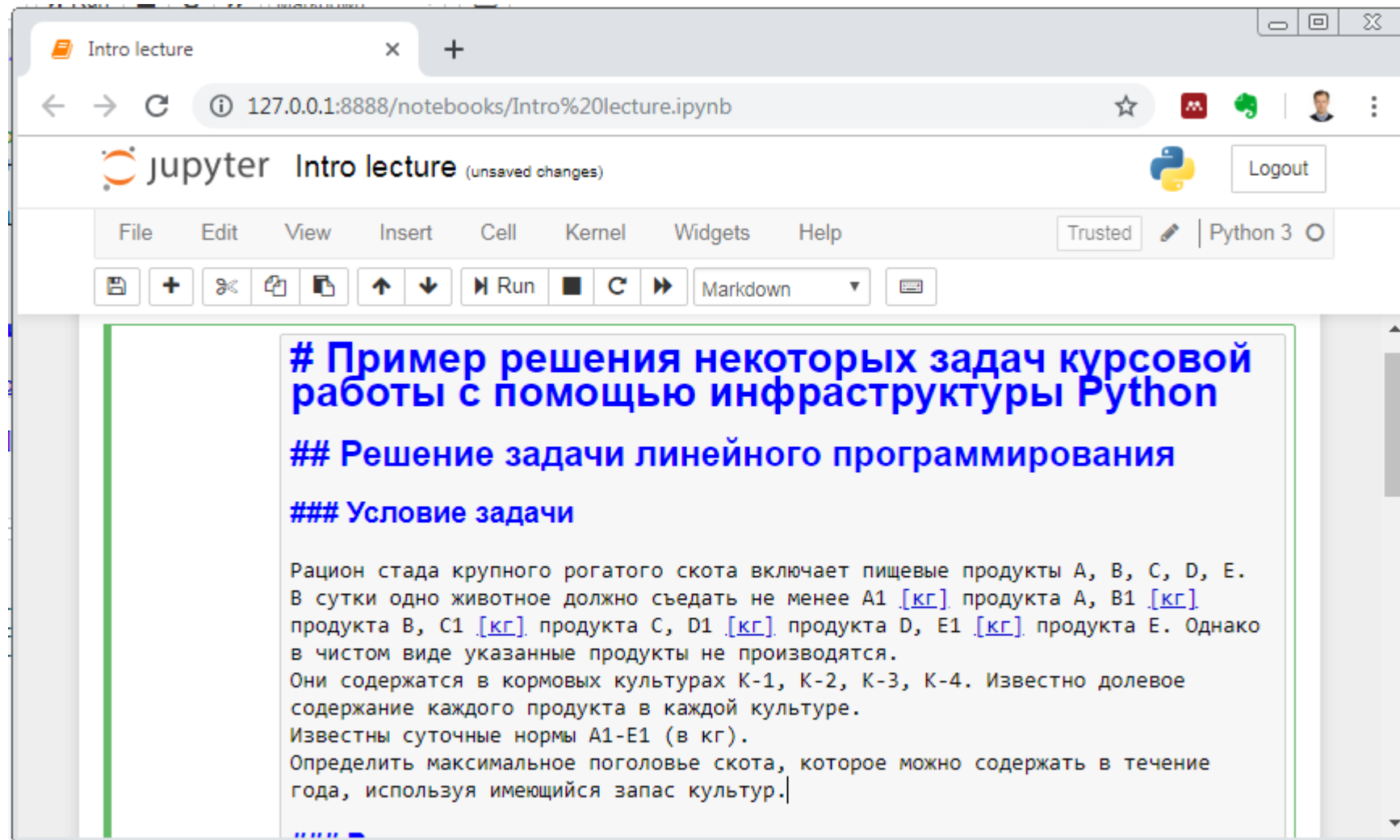
Некоторые библиотеки Python для решения задач ЛП

Библиотека	Матричный интерфейс	Моделирование	MILP	Анализ чувствительности	Пакет conda	Внешние солверы
CVXOPT	+	+/-	+	+	+	+
PuLP	-	+	+	-	+	+
scipy.optimize.linprog	+	-	-	-	+	-
Pyomo	-	+	+	?	+	?

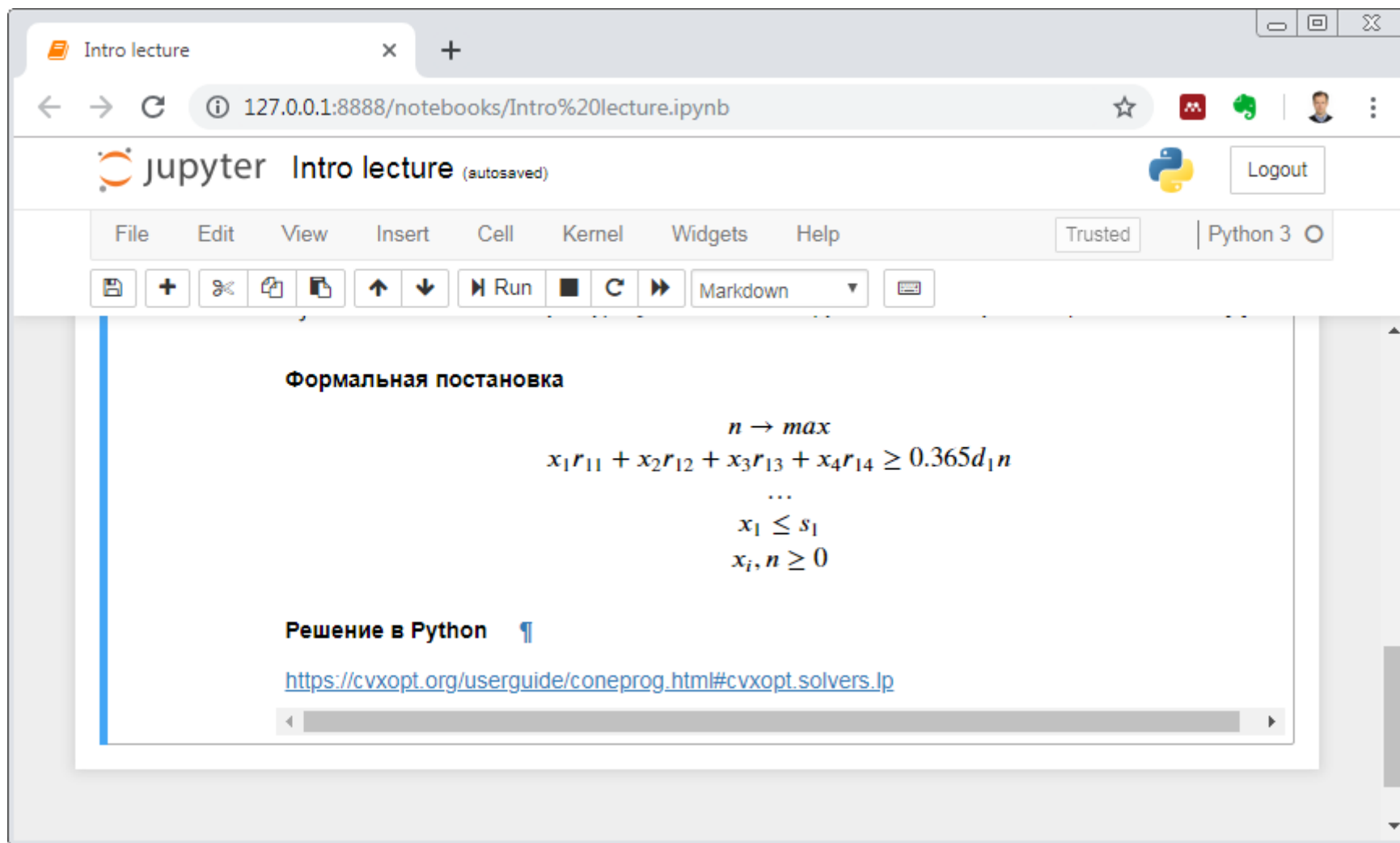
Использование Jupyter



Использование Jupyter



Использование Jupyter



Использование CVXOPT

`cvxopt.solvers.lp(c, G, h[, A, b[, solver[, primalstart[, dualstart]]]])`

$$\begin{array}{ll}\text{minimize} & c^T x \\ \text{subject to} & Gx + s = h \\ & Ax = b \\ & s \succeq 0\end{array}$$

$$\begin{array}{ll}\text{maximize} & -h^T z - b^T y \\ \text{subject to} & G^T z + A^T y + c = 0 \\ & z \succeq 0.\end{array}$$

Таким образом:

- c – коэффициенты переменных в целевой функции;
- G – коэффициенты переменных в ограничениях-неравенствах (\leq);
- h – правые части ограничений-неравенств (\leq);
- A – коэффициенты переменных в ограничениях-равенствах;
- b – правые части ограничений-равенств.

Использование CVXOPT: заполнение матриц

	x1	x2	x3	x4	n	Неравенство	Пр. часть
c	0	0	0	0	1	-	max (-1)
y1	r_{11}	r_{12}	r_{13}	R_{14}	$-0,365*d_1$	$\geq (-1)$	0
...
y6	1	0	0	0	0	\leq	s_1
...
y11	1	0	0	0	0	$\geq (-1)$	0
...

Использование CVXOPT: заполнение матриц

с – коэффициенты переменных в целевой функции

	x1	x2	x3	x4	n	Неравенство	Пр. часть
с	0	0	0	0	1	-	max (-1)
y1	r_{11}	r_{12}	r_{13}	R_{14}	$-0,365*d_1$	$\geq (-1)$	0
...
y6	1	0	0	0	0	\leq	s_1
...
y11	1	0	0	0	0	$\geq (-1)$	0
...

Использование CVXOPT: заполнение матриц

G – коэффициенты переменных в ограничениях-неравенствах

	x1	x2	x3	x4	n	Неравенство	Пр. часть
c	0	0	0	0	1	-	max (-1)
y1	r_{11}	r_{12}	r_{13}	R_{14}	$-0,365*d_1$	$\geq (-1)$	0
...
y6	1	0	0	0	0	\leq	s_1
...
y11	1	0	0	0	0	$\geq (-1)$	0
...

Использование CVXOPT: заполнение матриц

h – значения правых частей ограничений-неравенств

	x1	x2	x3	x4	n	Неравенство	Пр. часть
c	0	0	0	0	1	-	max (-1)
y1	r_{11}	r_{12}	r_{13}	R_{14}	$-0,365*d_1$	$\geq (-1)$	0
...
y6	1	0	0	0	0	\leq	s_1
...
y11	1	0	0	0	0	$\geq (-1)$	0
...

Пример решения ЗЛП с помощью CVXOPT

```
from cvxopt import matrix, solvers
```

```
c = matrix([0, 0, 0, 0, -1], tc='d')
G = matrix([[ -0.05, -0.13, 0, 0, 0.365 * 1.5],
            [ -0.10, -0.20, -0.10, 0, 0.365 * 1.4],
            [ -0.04, -0.06, -0.15, 0, 0.365 * 1.0],
            [ 0, -0.10, -0.08, -0.15, 0.365 * 0.8],
            [ -0.15, 0, -0.14, -0.10, 0.365 * 0.6],
            [1, 0, 0, 0, 0],
            [0, 1, 0, 0, 0],
            [0, 0, 1, 0, 0],
            [0, 0, 0, 1, 0],
            [-1, 0, 0, 0, 0],
            [ 0, -1, 0, 0, 0],
            [ 0, 0, -1, 0, 0],
            [ 0, 0, 0, -1, 0],
            [ 0, 0, 0, 0, -1],
            ], tc='d')
```

```
h = matrix([0, 0, 0, 0, 0, 5, 6, 7, 8,
            0, 0, 0, 0, 0], tc='d')
```

```
solution = solvers.lp(c, G.T, h, solver='glpk')
print('Status: ', solution['status'])
print('x = \n', solution['x'])
print('Objective: ', solution['primal objective'])
```

Status: optimal

x =

[5.00e+00]

[6.00e+00]

[8.44e-01]

[0.00e+00]

[1.88e+00]

Objective: -1.881278538812786

Решение задачи ЛП в GNU Octave

Пример решения ЗЛП в Octave (1)

- Параметры функции **glpk**:
 - c – вектор-столбец коэффициентов целевой функции;
 - A – матрица с коэффициентами ограничений;
 - b – вектор-столбец со значениями свободных членов;
 - lb – вектор с нижними границами для каждой переменной (0);
 - ub – вектор с верхними границами для каждой переменной, если нет, то бесконечность;
 - $stype$ – массив символов с типами ограничений:
 - F – свободное ограничение (не ограничение вовсе);
 - U – \leq (upper bound)
 - S – $=$
 - L – \geq (lower bound)
 - D – $-b \leq Ax \leq b$
 - $vartype$ – строка с типами переменных:
 - C – непрерывная переменная;
 - I – дискретная переменная.
 - $sense$ – 1 – minimization (default), -1 – maximization.
 - $param$ – структура с дополнительными параметрами.
- Возвращает:
 - $hopt$, $fmin$, $status$, $extra$. Status должен быть 180 – найдено оптимальное решение. Если нет, то нужно анализировать.

Пример решения ЗЛП в Octave (2)

- Возвращает:
 - `xopt` – значения переменных;
 - `fopt` – значение целевой функции;
 - `errnum` – код ошибки:
 - 0 – нет ошибки;
 - 2 – вырожденная матрица коэффициентов;
 - 10 – задача не имеет допустимых решений;
 - 11 – двойственная задача не имеет допустимых решений;
 - `extra` – дополнительная информация о решении:
 - `lambda` – теневые цены;
 - `costs` - приведенные цены;
 - `status` (5 – оптимальное решение).
- В старых версиях (3.6.x):
 - `xopt`, `fopt`, `status`, `extra`;
 - `Status` = 180, если найдено единственное решение.

**Нужно совместно
анализировать
`errnum` и `status`!**

Пример решения ЗЛП в Octave (3)

	x1 [C]	x2 [C]	x3 [C]	x4 [C]	n [C]	Неравенство	b
c	0	0	0	0	1	-	max (-1)
y1	r_{11}	r_{12}	r_{13}	r_{14}	$-0,365*d_1$	$\geq (L)$	0
...
y6	1					$\leq (U)$	s_1

Пример решения ЗЛП в Octave (4)

```
1;
```

```
c = [0 0 0 0 1]';
```

```
A = [0.05 0.13 0      0      -0.365 * 1.5;  
      0.10 0.20 0.10 0      -0.365 * 1.4;  
      0.04 0.06 0.15 0      -0.365 * 1;  
      0      0.10 0.08 0.15 -0.365 * 0.8;  
      0.15 0      0.14 0.10 -0.365 * 0.6;  
      1  0  0  0 0;  
      0  1  0  0 0;  
      0  0  1  0 0;  
      0  0  0  1 0];
```

```
b = [0 0 0 0 0 5 6 7 8]';
```

```
[x_max, z_max, en] = glpk(c, A, b,  
                           zeros(5, 1), # lb  
                           [],          # ub  
                           "LLLLLUUUU", # ctype  
                           "CCCCC",    # vartype  
                           -1);        # sense
```

```
x_max
```

```
z_max
```

```
en
```

Решение задачи ЛП в GLPK/MathProg

GNU Linear Programming Kit

- GLPK – свободно распространяемый пакет для решения задач ЛП (в том числе, целочисленного ЛП и смешанного ЛП)
- Структура:
 - Библиотека
 - «Обертки» для разных языков
- Язык математического моделирования GMPL (GNU MathProg)
 - Подмножество AMPL

Элементы GMPL

- Модель описывается с помощью объектов следующих видов:
 - Множества
 - Параметры
 - Переменные
 - Ограничения
 - Цели
- Объекты задаются с помощью языковых конструкций (предложений)
 - Декларативные и функциональные
- Разделение модели и данных (разные секции или даже разные файлы)

Пример решения ЗЛП в GLPK (1)

ration.mod

```
set Products;

set Crops;

param r{i in Products, j in Crops} >= 0, <= 1, default 0;

param d{i in Products} >= 0;

param s{j in Crops} >= 0; # Запасы культур

var n >= 0; # Искомое количество голов скота

var x {j in Crops} >= 0; # Расход культуры [т]

maximize z: n;

# Ограничения

s.t. Ration{i in Products}: sum{j in Crops} r[i, j] * x[j] >= 0.365 * d[i] * n; # Рацион по i-тому продукту

s.t. Stock{j in Crops}: x[j] <= s[j]; # Ограничение по запасам j-той культуры

solve;

end;
```

Пример решения ЗЛП в GLPK (2)

```
data;
```

ration.dat

```
set Products := A B C D E;
```

```
set Crops := K1 K2 K3 K4;
```

```
param r : K1      K2      K3      K4 :=
```

```
    A    0.05  0.13  0      0
```

```
    B    0.1   0.2   0.1   0
```

```
    C    0.04  0.06  0.15  0
```

```
    D    0     0.1   0.08  0.15
```

```
    E    0.15  0     0.14  0.1;
```

```
param d A 1.5 B 1.4 C 1 D 0.8 E 0.6;
```

```
param s := [K1] 5 [K2] 6 [K3] 7 [K4] 8;
```

```
end;
```

Пример решения ЗЛП в GLPK (3)

- Из консоли:
 - `glpsol -m ration.mod -d ration.dat -o ration.solution`
- Из GUSEK:
 - GUSEK | File | New (сохранить файл .mod)
 - File | New (сохранить в той же директории файл с данными – имя должно совпадать с модельным, расширение - .dat)
 - Tools | Generate Output File on Go (должно быть отмечено)
 - Tools | Go (справа появится вывод консольной GLPK, и откроется новый файл out, содержащий ответ в форме, которую еще нужно научиться читать – но об этом в следующий раз)