Астрономия

Работа с данными в астрономии

Электронное издание

Н. А. Архипова, А. С. Лужнов

Центр педагогического мастерства

Ассоциация победителей олимпиад

астрономия

ЦЕНТР ПЕДАГОГИЧЕСКОГО МАСТЕРСТВА

Н. А. Архипова, А. С. Лужнов

Работа с данными в астрономии

Электронное издание

Москва
Издательство МЦНМО

УДК 52
ББК 22.6я2
А87

Архипова Н. А., Лужнов А. С.
Работа с данными в астрономии.
Электронное издание.
М.: МЦНМО, 2025.
98 с.
ISBN 978-5-4439-3383-2

Важнейшей задачей современного астрономического образования является внедрение в программу обучения методов работы с астрономическими данными и многочисленными каталогами астрономических данных, методам математической обработки данных, основам анализа данных, а также повторению ключевых классических работ по астрономии и астрофизике, но на основе современных наблюдательных астрономических данных.

Эти задачи помогут учащимся более глубоко изучить классические работы и самостоятельно проверить их результаты, более эффективно и продуктивно подключиться в дальнейшем к исследованиям в области астрономии на современном уровне, а также подготовиться к нестандартным заданиям различных олимпиад по астрономии.

Сведения об авторах:

Архипова Наталья Анатольевна
к.ф.-м.н., доцент кафедры ФТМ ИПТИП РТУ МИРЭА и факультета физики НИУ ВШЭ.
e-mail: arkhipova_n@mirea.ru, naarkhipova@hse.ru

Лужнов Алексей Сергеевич
бакалавр, преподаватель в Лицее НИУ ВШЭ,
специалист по учебно-методической работе в ЦПМ.
e-mail: alexei.luzhnov@apo-team.ru

Издательство Московского центра
непрерывного математического образования
119002, Москва, Большой Власьевский пер., 11
тел. (499) 241–08–04
http://www.mccme.ru

ISBN 978-5-4439-3383-2

ОГЛАВЛЕНИЕ

Введение

На сегодняшний момент накоплено большое количество наблюдательных данных в астрономии и астрофизике, что связано с работой огромного количества современных космических и наземных обсерваторий и телескопов, работающих в широком диапазоне длин волн, а также улучшением качества приемников информации и ростом времени работы телескопов в режиме обзоров небесной сферы. В связи с этим возникла задача преподавания современной астрономии в области работы с астрономическими данными, так как изучение только классического курса астрономии не дает представления о современной работе астрономов и тенденциях развития науки.

Поэтому важнейшей задачей современного астрономического образования является внедрение в программу обучения методов работы с астрономическими данными и многочисленными каталогами астрономических данных, методам математической обработки данных, основам анализа данных, а также повторению ключевых классических работ по астрономии и астрофизике, но на основе современных наблюдательных астрономических данных. Эти задачи помогут учащимся более глубоко изучить классические работы и самостоятельно проверить их результаты, более эффективно и продуктивно подключиться в дальнейшем к исследованиям в области астрономии на современном уровне, а также подготовиться к нестандартным заданиям различных олимпиад по астрономии.

В рамках нашего учебного пособия будут описаны некоторые практические работы по обработке и анализу астрономических данных и работы с астрономическими каталогами, в том числе с каталогами известных космических проектов. Но стоит отметить, что данные методики неограничены только астрономией или физикой, поскольку любая современная наука от географии до литературы включает в себя анализ данных, в том или ином виде.

Глава 1. Основы работы Python

В ходе изучения основ работы с данными и астрономическими каталогами нам потребуется немного погрузится в программирование. Наиболее удобным языком программирования для научной работы является Python. Мы советуем воспользоваться репозиторием Anaconda[1] для установки интерактивного блокнота (Jupyter, Spyder или др.) и установки основных библиотек для дальнейшей работы: numpy, matplotlib, csv. Также нам потребуется некоторые дополнительные библиотеки для работы с астрономическими каталогами: astropy, pyvo, astroquery, PyAstronomy. В ходе реализации любого кода достаточно в начале сессии блокнота запустить все необходимые библиотеки для работы в первой строке с помощью функции import и сократить их название с помощью функции as:

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

Далее мы рассмотрим основные команды в каждой из библиотек, которые могут пригодится в дальнейшей работы.

Основы работы с массивами и циклами

Создание массива данных и печать объекта:

```
a = [1, 4, 7]
print(a) #с помощью хэштега можно оставлять комментарии, они не влияют на код

>>> 1, 4, 7
```

Вызов определенного элементного массива:

```
print(a[1]) #счет начинается с 0, поэтому 1 - это 2-й элемент массива

>>> 4
```

Численные элементы массива можно суммировать между собой:

```
print(a[1]+a[2])

>>> 11
```

Символьные элементы массива тоже можно суммировать между собой:

[1] https://www.anaconda.com/

```
d = ['1', '4', '7']
print(d[1]+d[2])

>>> 47
```

Двумерный массив и вызов элемента 2-ой строки 3-го столбца:

```
b = [2, 6, 9]
c = [a, b]
print(c)
print(c[1][2])

>>> [[1, 4, 7], [2, 6, 9]]
>>> 9
```

Сумма арифметической прогрессии с первым членом равным a, шагом k и числом элементов n:

```
a = 1
k = 3
n = 10
s = 0

for i in range(0, n):
    s += a + k*i

print(s)

>>> 145
```

Цикл, который рисует “ёлочку” из единиц размером n:

```
n = 5
d = '1'

print(d)
for i in range(0, n-1):
    d += '1'
    print(d)

for i in range(1,n):
    print('1'*(n-i))

>>> 1
>>> 11
>>> 111
>>> 1111
>>> 11111
>>> 1111
```

```
>>> 111
>>> 11
>>> 1
```

Numpy

Библиотека numpy одна из наиболее распространенных библиотек для работы с многомерными массивами и различными математическими функциями. Большая часть задач, которые будут разбираться в данном пособии будут так или иначе задействовать операции с данной библиотекой. Рассмотрим некоторые примеры.

Преобразование градусной меры угла в радианную меру:

```
np.radians(30)

>>> 0.5235987755982988
```

Создание массива, заполненного равномерно числами от 4 до 15 с шагом 1

```
x = np.arange(4, 15, 1)

print(x)

>>> [ 4  5  6  7  8  9 10 11 12 13 14]
```

Определение среднеарифметического значения данного массива:

```
print(np.mean(x))

>>> 9.0
```

Создание массива из 10 элементов, заполненного случайными числами от 2 до 5:

```
print(np.random.uniform(2.0, 5.0, 10))

>>> [2.17515896 3.40476591 4.60322291 4.40105342 4.72247378 2.41232706
 2.32073319 2.95868821 3.58350452 4.81851736]
```

Создание массива из 10 нулей:

```
print(np.zeros(10))
```

```
>>> [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

Основные математические операции:

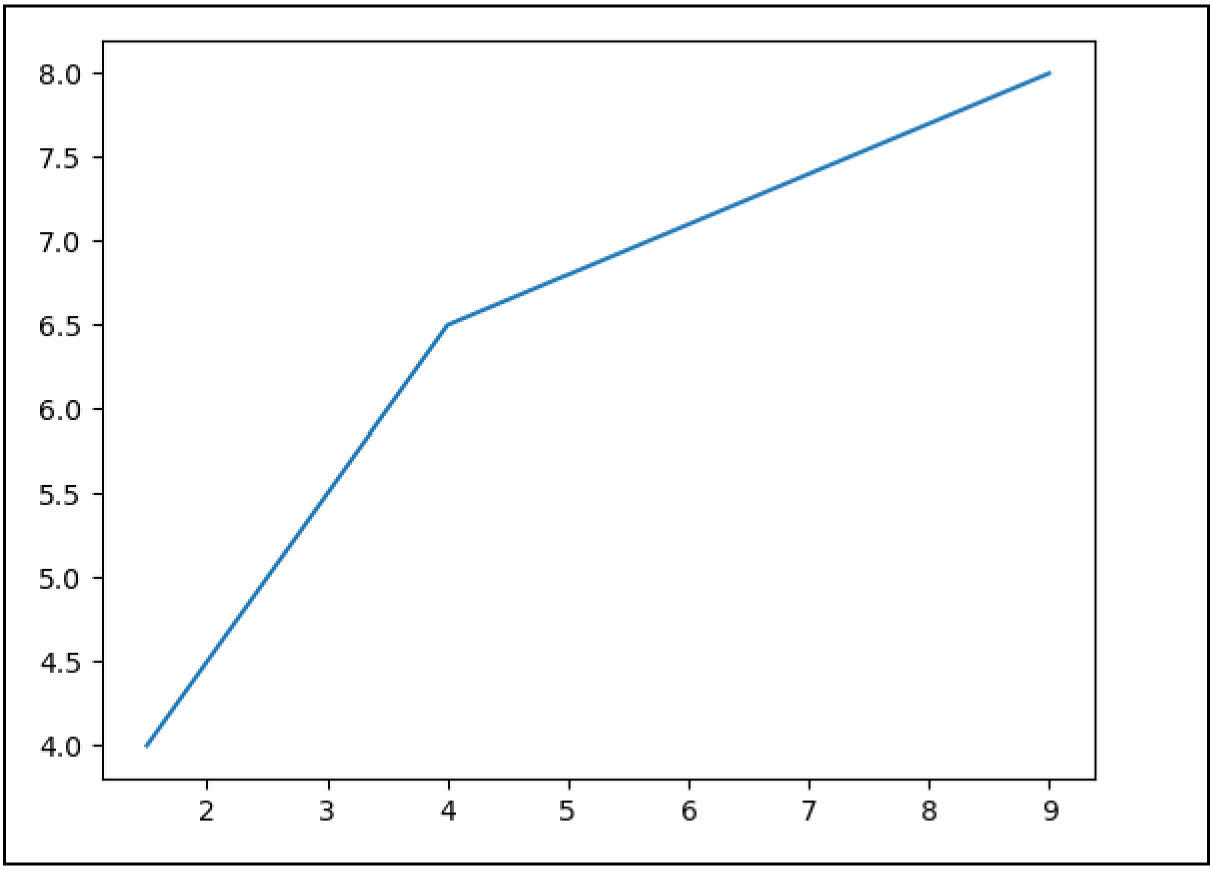
```
t = [0, 1, np.pi, 4, 10]
y = np.array(t) # меняем тип массива с 'list' на 'numpy.ndarray'

print(np.max(y))
print(np.min(y))
print(np.cos(y))
print(np.sin(y))
print(np.tan(y))
print(np.exp(y))
print(np.sqrt(y))

>>> 10.0
>>> 0.0
>>> [ 1.          0.54030231 -1.         -0.65364362 -0.83907153]
>>> [ 0.00000000e+00  8.41470985e-01  1.22464680e-16 -7.56802495e-01
 -5.44021111e-01]
>>> [ 0.00000000e+00  1.55740772e+00 -1.22464680e-16  1.15782128e+00
  6.48360827e-01]
>>> [1.00000000e+00 2.71828183e+00 2.31406926e+01 5.45981500e+01
 2.20264658e+04]
>>> [0.         1.         1.77245385 2.         3.16227766]
```

Matplotlib

Библиотека matplotlib наиболее популярна для графического отображения любых данных. Особенностью построения графиков в Python является то, что все графики строятся “по точкам”. То есть для построения любой функции необходимо создать начальный массив значений аргумента, по которым дальше считаются и наносятся значения функций.

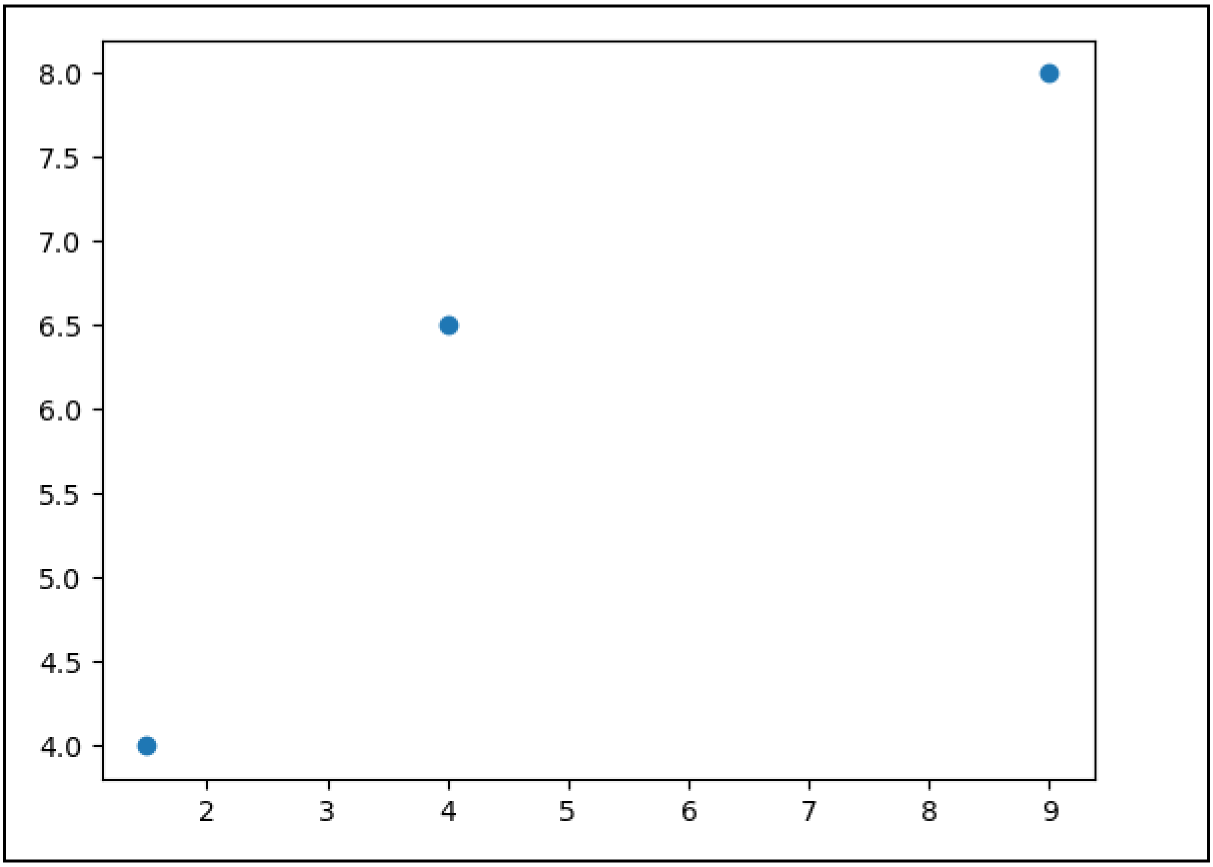
Построение функции по точкам:

```
a = [1.5, 4, 9]
b = [4, 6.5, 8]

plt.plot(a, b)

>>>
```



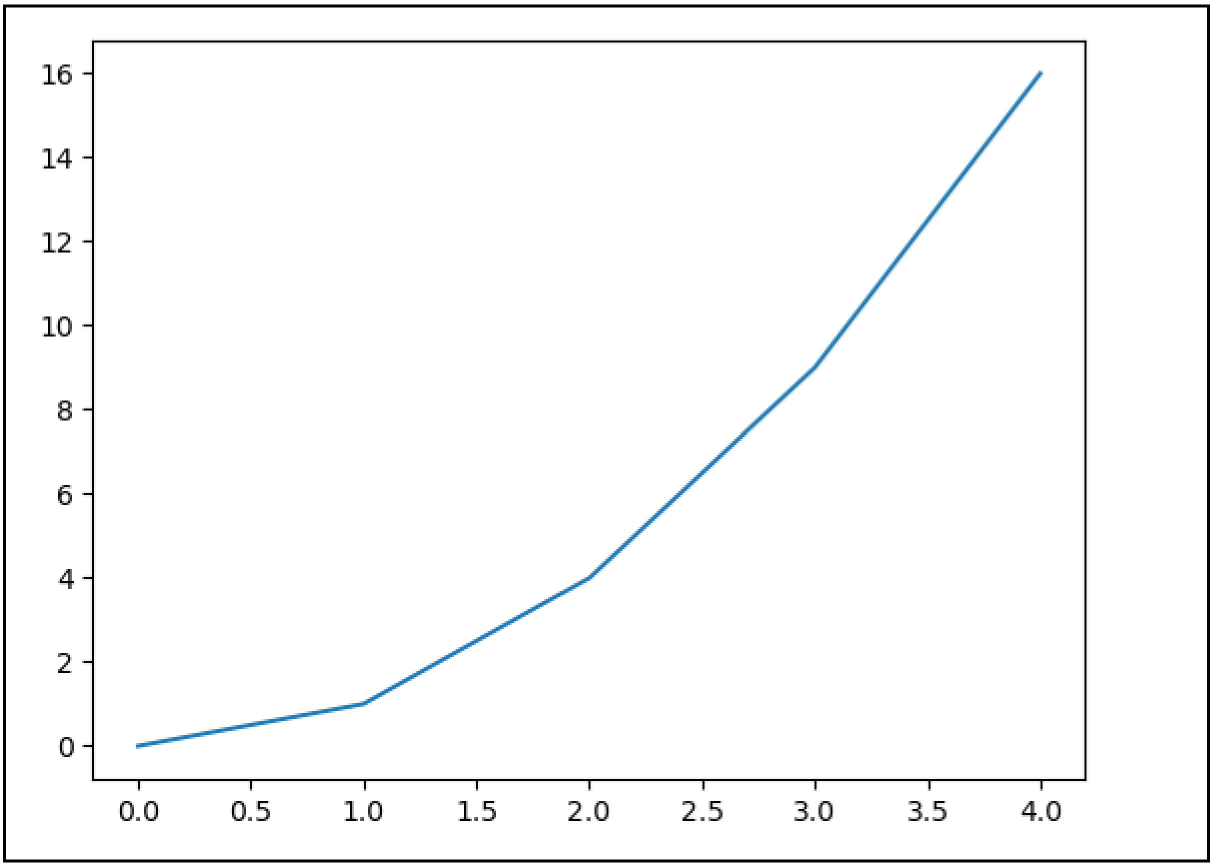
Отображение точек на графике:

```
plt.scatter(a, b)

>>>
```



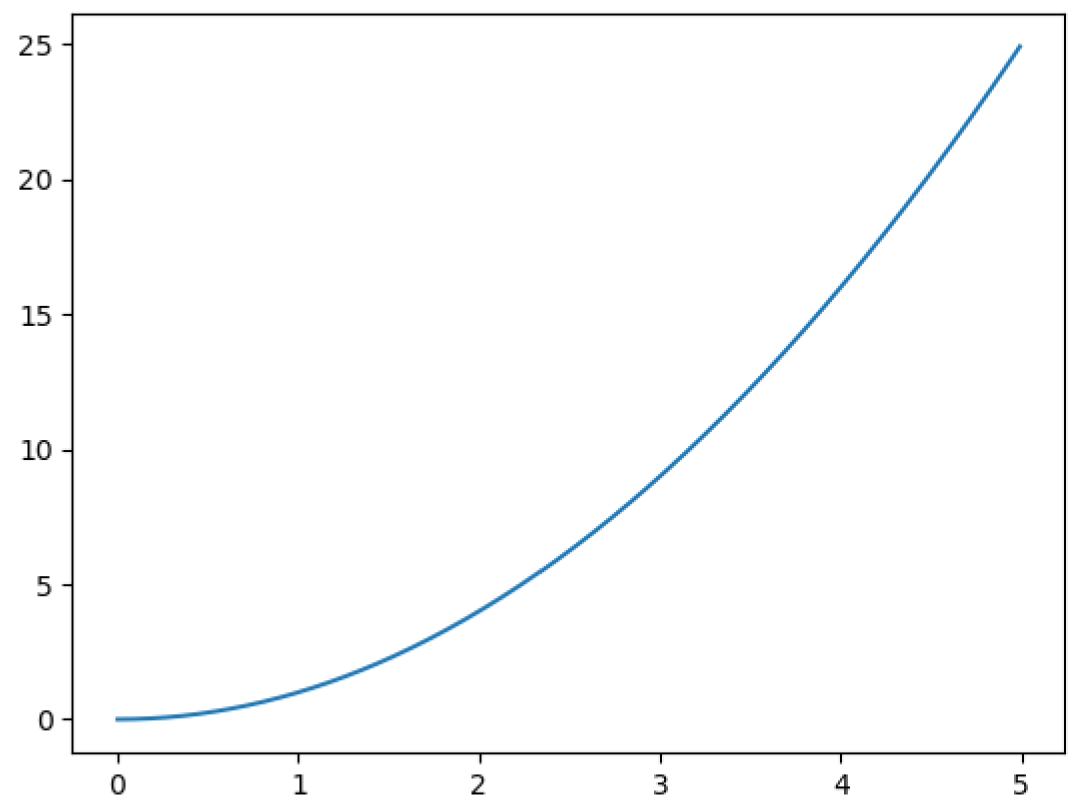
Построение графика параболы в пределах по аргументу от 0 до 5 с шагом 1:

```
x = np.arange(0.0, 5, 1)
y = x**2

plt.plot(x, y)

>>>
```



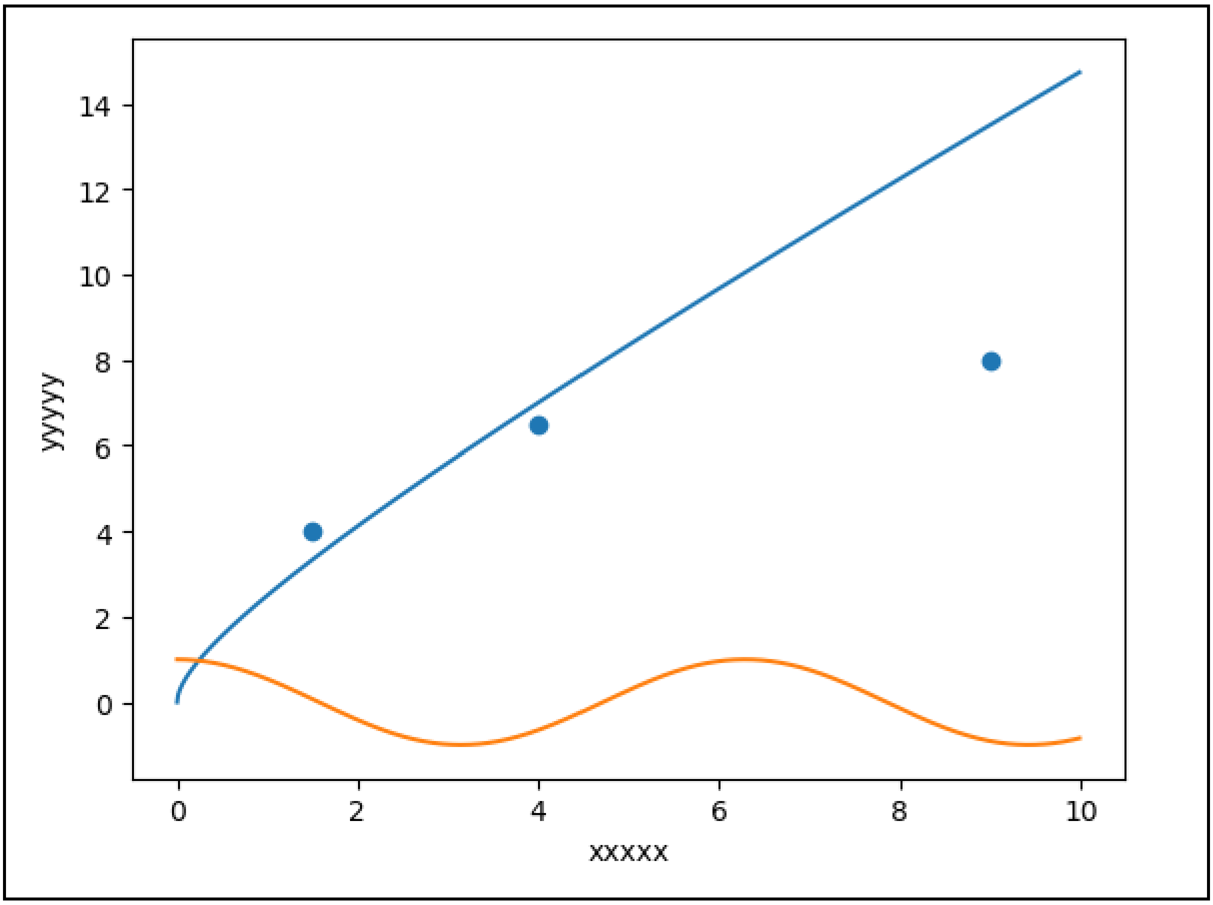
Построение графика параболы в пределах по аргументу от 0 до 5 с шагом 0.01:

```
x = np.arange(0.0, 5, 0.01)
y = x**2

plt.plot(x, y)

>>>
```



Построение нескольких функций и отображение точек на одном графике:

```
x = np.arange(0.0, 10, 0.01)
y = 1.5*x**0.5 + x
z = np.cos(x)

plt.plot(x, y)
plt.plot(x, z)
plt.scatter(a, b)

plt.xlabel('xxxxx') #подпись по горизонтальной оси
plt.ylabel('yyyyy') #подпись по вертикальной оси

>>>
```



CSV

Основная часть таблиц данных, которые используются в науке, хранят в форме “.csv”. Данный формат работает с офисными программами (Microsoft Excel, LibreOffice Calc и др.) и легко читается и форматируется в Python с помощью библиотеки csv.

Чтение файла data_test.csv:

```
x = []
#обратите внимание, что местоположение файла может отличаться у вас
with open('Downloads/data_test.csv') as f:
    reader = csv.reader(f)
    for row in reader:
        x.append(row)
print(x)

>>> [['1', '2'], ['3', '4'], ['4', '13.6'], ['14.27', '15.981'], ['51', '376']]
```

В данном случае он воспринимает все значения нашего массива, как символьные элементы. Для преобразования их в численные можем воспользоваться функцией float:

```
x = np.array(x)
n = len(x[0])
k = len(x[:, 0])

y = [[0.0]*n]*k
y = np.array(y)

for i in range(0,k):
    for j in range(0, n):
        y[i][j] = float(x[i][j])

print(y)

>>>
[[  1.      2.   ]
 [  3.      4.   ]
 [  4.     13.6  ]
 [ 14.27   15.981]
 [ 51.    376.   ]]
```

Создание и сохранение массива в новый файл data_test_new:

```
data = ['55', '66']

with open('Downloads/data_test_new.csv', 'w', newline='') as f:
    csv.writer(f).writerows(data)
```

Глава 2. Нахождение зависимостей

Одной из важнейших задач современной науки является установка законов, соответствующим наблюдениям и экспериментам. На данный момент многие области астрономии ещё не имеют фундаментальных теорий, как в классической физике, поэтому многие законы выведены только эмпирическим путем из анализа большого числа наблюдений. Но как связать между собой набор наблюдаемых величин и построить между ними зависимость?

Прямая пропорциональность

Поставим такую задачу:

«Допустим, что есть некая зависимость $y(x) = kx$. У нас имеется набор наблюдаемых величин $\{x_i\,; y_i\}$ и мы хотим получить коэффициент этой зависимости, чтобы можно было предсказать значение $y(x_0)$, которое является не экспериментальным».

Здесь необходимо сразу уточнить, что мы можем экстраполировать нашу зависимость исключительно на некоторый диапазон соизмеримый с масштабами наших наблюдений.

В школе на уроках физики вы проводили лабораторные работы и сталкивались с данной задачей. Для нахождения k вас просили построить график $y(x)$, нанести все экспериментальные точки с крестами погрешностей и далее провести прямую проходящую через большинство крестов погрешностей на глаз.

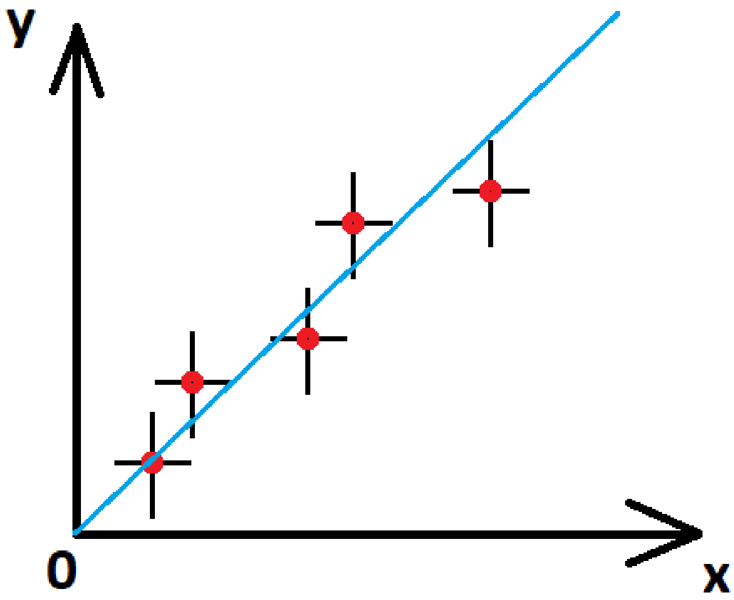


Рисунок 2.1. Построение прямой через кресты погрешностей

Проблема данного метода заключается в неточности понятия «на глаз», так как всегда можно провести прямую или чуть ниже или чуть выше.

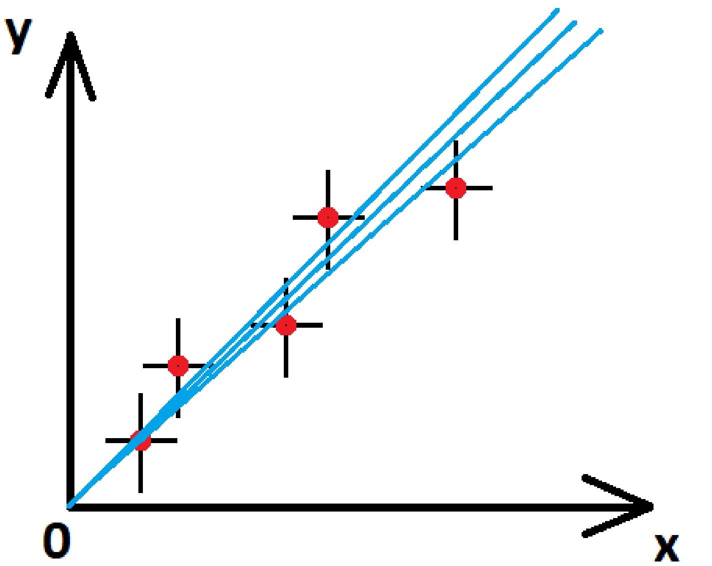


Рисунок 2.2. Построение нескольких прямых через кресты погрешностей

Так какую прямую нужно выбрать? Это одна из важнейших проблем статистического анализа. Необходимо создать критерий «правильности» прямой. Данным критерием будет служить минимум суммы квадратов расстояний от всех экспериментальных точек до теоретической прямой. Он достаточно удобен для расчетов и лучше учитывает возможные выбитые точки.

Давайте вспомним, как определяется расстояние (d) от точки $(x_0\ ;\ y_0)$ до прямой $y(x) = kx$.

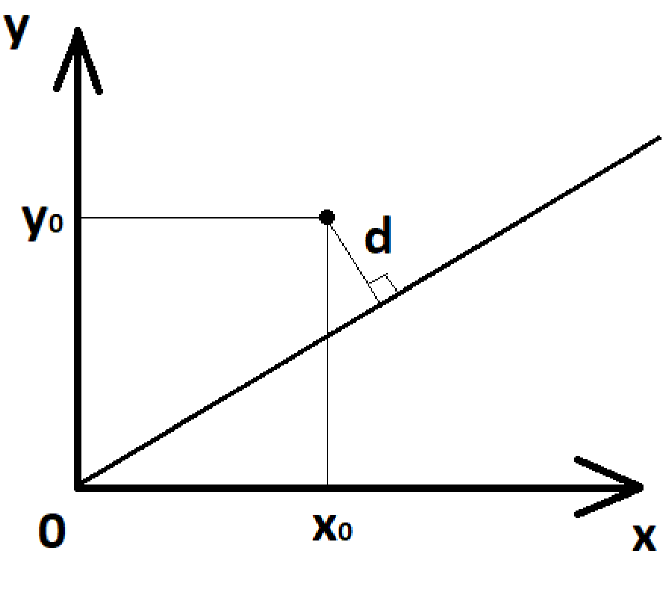


Рисунок 2.3. Определение расстояния от точки до прямой

Из подобия прямоугольных треугольников с вертикальным углом получаем:

$$\frac{d}{x_0} = \frac{y_0 - kx_0}{\sqrt{{x_0}^2 + k^2{x_0}^2}}$$

$$d = \frac{y_0 - kx_0}{\sqrt{1+k^2}}$$

Теперь запишем наш критерий «правильности»:

$$\sum_{i=1}^{N} {d_i}^2 = \sum_{i=1}^{N} (\frac{y_i - kx_i}{\sqrt{1+k^2}})^2 = \frac{1}{1+k^2}\sum_{i=1}^{N} ({y_i}^2 - 2ky_ix_i + k^2{x_i}^2) =$$

$$= \frac{1}{1+k^2}[\sum_{i=1}^{N} {y_i}^2 - 2k\sum_{i=1}^{N} x_iy_i + k^2\sum_{i=1}^{N} {x_i}^2]$$

Получили квадратичную зависимость по k с ветвями вверх, значит минимум параболы определяется её вершиной.

$$k = \frac{\sum_{i=1}^{N} x_iy_i}{\sum_{i=1}^{N} {x_i}^2}$$

Данный метод называется *метод наименьших квадратов* или сокращенно МНК.

Линейная зависимость

В случае если наша зависимость имеет не прямую пропорциональность, а линейный вид $y(x) = ax + b$, то идея определения параметров *a* и *b* не меняется. Но теперь наш критерий правильности будет зависеть сразу от двух величин:

$$\sum_{i=1}^{N} {d_i}^2 = \sum_{i=1}^{N} (\frac{y_i - bx_i - a}{\sqrt{1+b^2}})^2$$

И необходимо минимизировать наш критерий сразу по двум параметрам *a* и *b*:

$$\frac{d}{da}[\sum_{i=1}^{N} (\frac{y_i - bx_i - a}{\sqrt{1+b^2}})^2] = 0$$

$$\frac{d}{db}[\sum_{i=1}^{N} (\frac{y_i - bx_i - a}{\sqrt{1+b^2}})^2] = 0$$

Получим, что:

$$a = \frac{N\sum_{i=1}^{N} x_iy_i - (\sum_{i=1}^{N} x_i)(\sum_{i=1}^{N} y_i)}{N\sum_{i=1}^{N} {x_i}^2 - (\sum_{i=1}^{N} x_i)^2}$$

$$b = \frac{\sum_{i=1}^{N} y_i - a\sum_{i=1}^{N} x_i}{N}$$

Пример реализации в Python

```
def MNK(data):
    sum_xy = 0
    sum_x = 0
    sum_y = 0
    sum_x2 = 0
    N = len(data)
    for (i, _) in enumerate(data[0]):
        sum_xy += data[0][i]*data[1][i]
        sum_x += data[0][i]
        sum_y += data[1][i]
        sum_x2 += data[0][i]**2
    a = (N*sum_xy-sum_x*sum_y)/(N*sum_x2-sum_x**2)
    b = (sum_y-a*sum_x)/N
    return a, b
```

Логарифмирование функции

К сожалению, линейные зависимости не так часто встречаются в астрофизике. Достаточно часто неизвестен даже характер пропорциональности. Рассмотрим теперь случай, когда наша зависимость имеет вид $y(x) = kx^b$ и попробуем свести данную задачу к уже имеющимся.

$$ln(y) = ln(kx^b) = ln(k) + b\ ln(x)$$

Получили линейную зависимость $ln(y)$ от $ln(x)$, где показатель степени легко определяется с помощью МНК. Данное преобразование принято называть - *логарифмированием функции*.

Теперь давайте посмотрим как данные математические теории можно применить на практических задачах астрономии с реальными наблюдательными данными.

Зависимость «большая полуось - период»

Найдите показатель зависимости между большой полуосью (a) планет Солнечной системы и их сидерическими (звездными) периодами (T), если данная зависимость описывается законом $T(a) = ka^n$

Таблица 1. Данные больших полуосей и периодов Солнечной системы

Планета	Большая полуось, а.е.	Период, год

Меркурий	0,387	0,241
Венера	0,723	0,616
Земля	1,000	1
Марс	1,52	1,887
Юпитер	5,20	11,857
Сатурн	9,54	29,4
Уран	19,19	84,02
Нептун	30,07	164,8

Важное отличие наблюдательных данных от математических теорий - это наличие размерностей. Поэтому обезразмерим наши параметры:

$$y = a/a_0 \text{ , где } a_0 = 1 \text{ а.е.}$$
$$x = T/T_0 \text{ , где } T_0 = 1 \text{ год}$$
$$k' = \frac{k{a_0}^n}{T_0}$$

Таблица 2. Преобразование переменных из Таблицы 1

y	x	ln(y)	ln(x)
0,387	0,241	-0,949330586	-1,422958345
0,723	0,616	-0,3243460568	-0,4845083154
1,000	1	0	0
1,52	1,887	0,4187103349	0,6349882664
5,20	11,857	1,648658626	2,47291841
9,54	29,4	2,255493485	3,380994674
19,19	84,02	2,95438931	4,431054866
30,07	164,8	3,403527997	5,104732617

Получим, что:

$$n = \frac{N\sum_{i=1}^{N} x_i y_i - (\sum_{i=1}^{N} x_i)(\sum_{i=1}^{N} y_i)}{N\sum_{i=1}^{N} {x_i}^2 - (\sum_{i=1}^{N} x_i)^2} = 1{,}499 \approx 1{,}5$$

$$k' = \frac{\sum_{i=1}^{N} y_i - b\sum_{i=1}^{N} x_i}{N} = 1{,}002 \approx 1$$

Получили что : $T(a) \sim a^{3/2}$

Фактически мы смогли экспериментально подтвердить III Закон Кеплера на основе анализа наблюдательных данных о планетах Солнечной системы.

Этот закон вместе с I и II был установлен эмпирическим путем Иоганном Кеплером еще в 1619 году (в 1609 года он сформулировал свои законы только для орбиты Марса, а в 1619 уже расширил их на все планеты Солнечной системы) на основе наблюдений Тихо Браге. Данные законы легли в основу Закона Всемирного Тяготения Исаака Ньютона и породили начало классической механики.

Зависимость «масс - светимость»

Также по наблюдательным данным можно найти показатель зависимости между светимостью (L) звезд главной последовательности и их массой, если данная зависимость имеет вид $L = kM^n$.

Таблица 3. Данные светимостей и масс звезд главной последовательности

Название	Светимость (L/Lo)	Масса (M/Mo)
Регул	130	3,50
Сириус А	63	2,60
Фомальгаут	40	2,20
Альтаир	24	1,90
Процион	4,0	1,35
α Центавра А	1,45	1,08
Солнце	1,0	1,0
μ Кассиопеи	0,7	0,95
τ Кита	0,44	0,85
Поллукс	0,36	0,83
ε Эридана	0,28	0,78
α Центавра В	0,18	0,68

Где Lo и Mo - светимость и масса Солнца, соответственно.

Получим, что:

$$n = \frac{N\sum_{i=1}^{N} x_i y_i - (\sum_{i=1}^{N} x_i)(\sum_{i=1}^{N} y_i)}{N\sum_{i=1}^{N} {x_i}^2 - (\sum_{i=1}^{N} x_i)^2} = 4{,}2 \approx 4$$

$$k' = \frac{\sum_{i=1}^{N} y_i - b\sum_{i=1}^{N} x_i}{N} = 1{,}002 \approx 1$$

И получим классическую зависимость “масса-светимость” для маломассивных звезд главной последовательности на диаграмме Грецшпрунга-Рассела: $L \sim M^4$.

Данное задание имеет особую актуальность, так как подобное задание было предложено на теоретическом туре московской астрономической олимпиады в 2016 учебном году[2] (7 задача).

Домашнее задание

Изучите зависимость светимости радиопульсара от частоты излучения. Обратите внимание, что данная зависимость будет зависеть от диапазонов частоты излучения, поэтому сделайте деление на частоты: от 50 до 200 МГц, от 0,5 до 10 ГГц и от 5 ГГц.

Данные по радио-пульсарам доступны по ссылке:
https://www.atnf.csiro.au/research/pulsar/psrcat/

[2] https://mosastro.olimpiada.ru/upload/files/Moscow_2016_solutions_10-11.pdf

Глава 3. Ошибки измерений

В рамках изучения физики в школе учащихся знакомят с некоторыми видами погрешностей, в основном с погрешностями измерительных приборов. Часто данная задача является только “навесом” к основной задаче измерения, потому что объекты изучения всегда подобраны наилучшим образом. В реальной науке такое скорее является исключением.

Рассмотрим такой пример: при измерении толщины иголки с помощью обычной линейки значение диаметра составит 1 мм, а погрешность линейки составит цену деления, то есть тоже 1 мм. Любой добросовестный ученик запишет итоговый ответ d = (1±1) мм. В реальности данный ответ означает, что есть некоторая вероятность того, что толщина иголки равна 0 и её в целом не существует. Но она точно есть! Значит для правильного определения толщины иголки нужно совершенствовать измерительный прибор или методику определения. Например, взять микрометр, или измерить с помощью метода рядов несколько иголок, или применить метода накатывания. В астрономии задача усложняется тем, что мы не можем влиять на объекты изучения, а совершенствование приборов может занимать десятки (а в некоторых случаях сотни или тысячи) лет.

Как еще один пример можно рассмотреть факт доказательства гелиоцентризма. Ученые Древней Греции предполагали, что в центре Солнечной системы может располагаться Солнце, тогда при наблюдении любой звезды должно меняться её направление, по отношению к Солнцу (см. рисунок). Они измерили данные углы и получили, что углы совпадают, следовательно в центре Солнечной системы располагается Земля. Астрономы древности не могли знать, что расстояние до звезд настолько велико, что разница этих углов (2π, где π – параллакс) составляет тысячные градуса, а погрешность их измерительных приборов составляла 0,5°-1°.

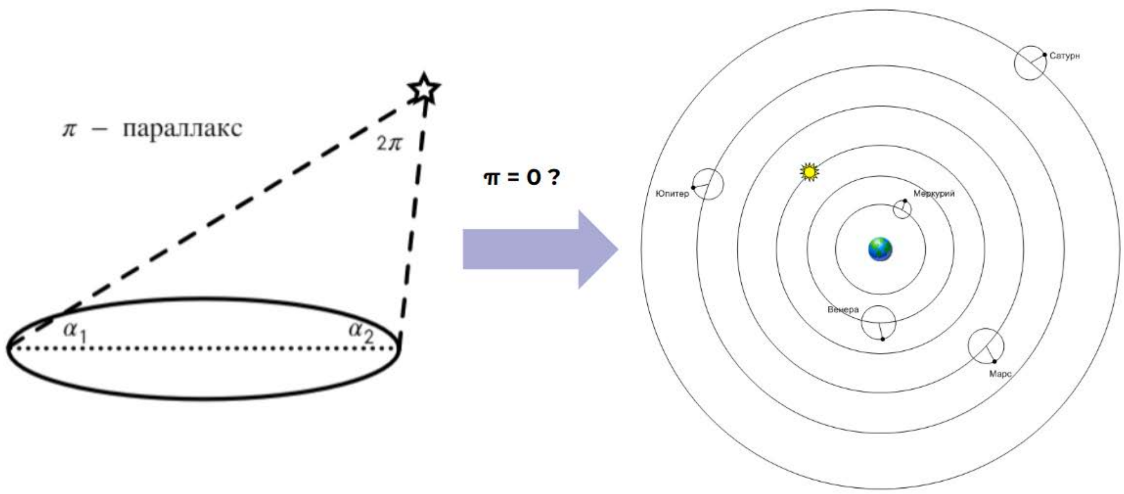


Рисунок 3.1. Доказательство геоцентризма

Данную теорию сломала фактически случайность, а именно наличие внутренней планеты. Галилео Галилей смог обнаружить фазы Венеры, которые никак не соответствовали геоцентризму и вернули к идее гелиоцентризма. Это дало огромный толчок к совершенствованию измерительных приборов в астрономии для поиска того самого параллакса. Уже спустя 2 века Василий Струве на Пулковской обсерватории смог обнаружить параллакс Веги и поставить финальную точку в гелиоцентризме.

Несмотря на значимость понимания мироустройства, данная ошибка мало влияла на измерения других ученых, в отличии от ошибки измерения года и как следствие создание Юлианского календаря. В данном случае систематическая ошибка накапливалась на протяжении множества столетий и значительно влияла на самые базовые астрономические наблюдения. Тот же день весеннего равноденствия, от которого отсчитываются многие астрономические параметры, смещался более, чем на неделю к Средним векам.

Как можно увидеть из этих примеров, нельзя недооценивать значимость ошибок измерений. Желание астрономов строить все более крупные и дорогие обсерватории, делать все более качественные измерительные приборы, обусловлено желанием узнать, как можно больше о нашей Вселенной. Малая погрешность измерительного прибора может скрыть от объектива наличие планеты пригодной для жизни, а неучтенная систематическая ошибка через сотни лет может привести к непредвиденным последствиям в работе навигационных спутников.

Систематические и случайные ошибки измерений

Теперь рассмотрим основные типы и учеты систематических и случайных ошибок измерений. Разница между систематическими и случайными носит условный характер и зависит от условий проведения эксперимента и методов расчетов.

Систематические ошибки характеризуются невозможностью их устранения путем усреднения данных. К ним относится:

- *Инструментальная погрешность прибора*
  У любого измерительного прибора есть класс точности и детально прописанная система инструментальной погрешности, которую определяет производитель. Данная погрешность может меняться в зависимости, как от диапазона измерений, так и от условий использования.

- *Калибровочная*
  В физике или астрономии сложно найти точку отсчета для любой величины. В случае угловых координат она может быть связана с неточностью времени или смещением в процессе измерений. А для яркости может вноситься вклад от общего фона. Подробнее про учёт фона мы расскажем в следующих главах.

- *Ошибки округления*
  Основная часть расчетов в науке происходит на компьютере. В нём есть ограничение по хранению информации от каждого числа. Например для 64-битной архитектуры нельзя одновременно хранить значения чисел, которые отличаются более чем в 10^20 раз. В случае астрономии многие параметры меняются экспоненциально и их значения могут выходить за эти пределы. Это может в некоторых случаях накапливаться и значительно портить итоговые результаты. Например, если вы попросите компьютер просуммировать ряд

чисел [10^-21; …; 10^-21; 1] сначала слева-направо, то вы получите число большее 1, а если справа-налево, то вы получите 1.

```
x = 10**(-21)
n = 10**6
q = 0

for i in range(0, n):
    q += x

print(q+1)

>>>  1.000000000000001

x = 10**(-21)
n = 10**6
q = 1

for i in range(0, n):
    q += x

print(q)

>>> 1.0
```

Случайная погрешность характеризуется возможностью её устранения путем усреднения данных. То есть при повторения эксперимента N раз и усреднением по N случайная погрешность будет иметь обратную зависимость от N по закону больших чисел. Часто случайные погрешности являются сочетанием влияния многих факторов.

Если эти факторы независимы друг от друга и оказывают малое воздействие на объект изучения, то можно принять их воздействие «нормальным», то есть функция их распределения подчиняется распределению по Гауссу. И для данного случая применимо правило «трёх сигм», согласно которому значение любой функции с вероятностью 99,73% лежит в пределе [x-3σ; x+3σ].

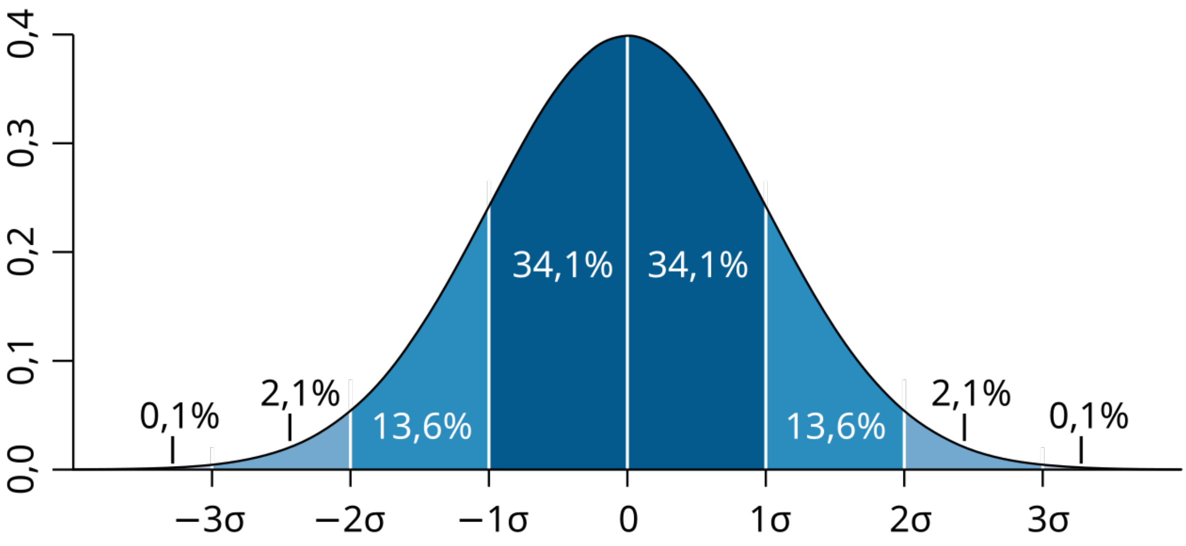


Рисунок 3.2. Гауссовское распределение и правило «трёх сигм». Иллюстрация из Википедии.

Среднеквадратичное отклонение (σ) рассчитывается следующим образом:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N} \quad (x_i - \underline{x})^2}{N-1}}$$

Где x_i - значение i-го измерения, $\underline{x}$ - среднеарифметическое значение величины, а N - число измерений. В большинстве случаев N достаточно велико и формула упрощается до:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N} \quad (x_i - \underline{x})^2}{N}}$$

Пример реализации в Python

Построение гауссовского распределения:

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

x = np.random.normal(1000, 50, 100000)
plt.hist(x, bins=100)
```

Результат:

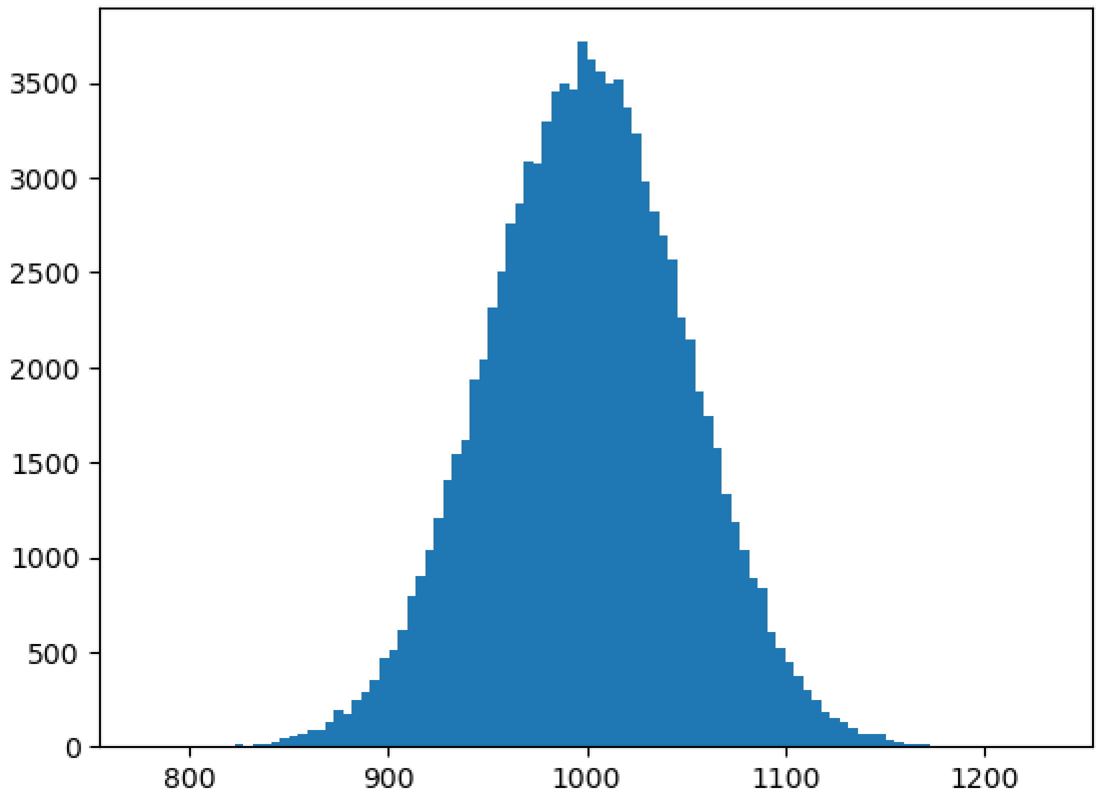


Рисунок 3.3. Гауссовское распределение с нормальным значением 1000 и σ=50

Определение среднеквадратичного отклонения для нормального распределения:

```
z = 0
N = 0
n = len(x)

for i in range(0, n):
    N += 1
    z += x[i]

z_mean = z/N
print(z_mean)

sum_sigma = 0
for i in range(0, n):
    sum_sigma += (x[i]-z_mean)**2

sigma = np.sqrt(sum_sigma/(N-1))
print(sigma)
```

В данном случае получим среднее значение случайной величины 1000 и среднеквадратичное отклонение 50 с некоторой погрешностью.

Метод Монте-Карло

Случайность величины иногда может играть на пользу при анализе данных. С помощью случайного распределения можно определять интегралы различных функций с достаточно высокой точностью при наименьшей затрате вычислительной мощности. Данный метод получил свое название в честь столицы Монако - самого известного игорного города наравне с Лас-Вегасом. Суть метода заключается в следующем: если параметры некоторого процесса можно интерпретировать через вероятность случайного процесса, то можно с помощью генератора случайных чисел посчитать параметры данного процесса. Самым простым примером является подсчет определенного интеграла через вероятность попадания точки “над функцией” и “под функцией”.

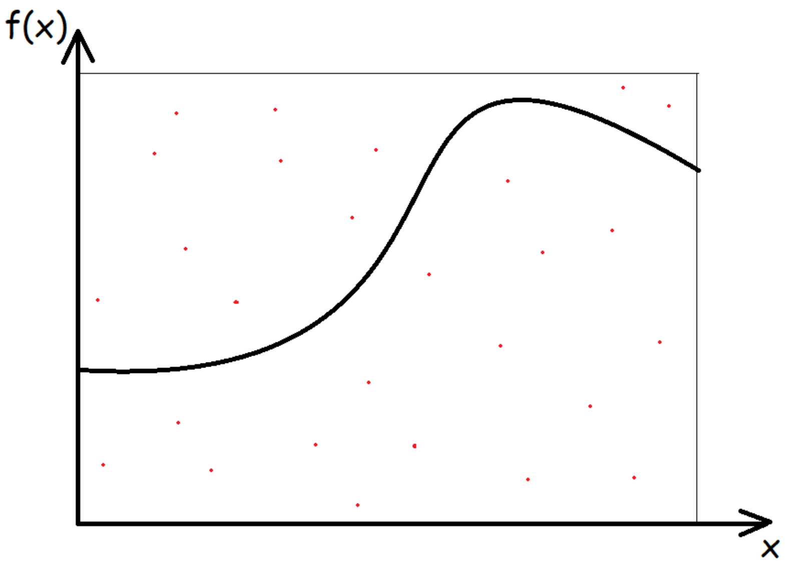


Рисунок 3.4. Иллюстрация к методу Монте-Карло

В данном случае площадь под графиком можно выразить, как отношение числа точек “под функцией” к суммарному числу точек и умножив на выделенную площадь бросания. Пример реализации:

```
N = 100
x_max = 10.0
x = np.arange(0.0, x_max, x_max/N)
y = x**2
t = np.random.uniform(0.0, max(y), N)

plt.plot(x, y)
plt.scatter(x, t)

k = 0

for i in range(0, N):
    if t[i]<y[i]:
        k += 1
```

```
S = (k/N)*(x_max*max(y))
print(S)
```

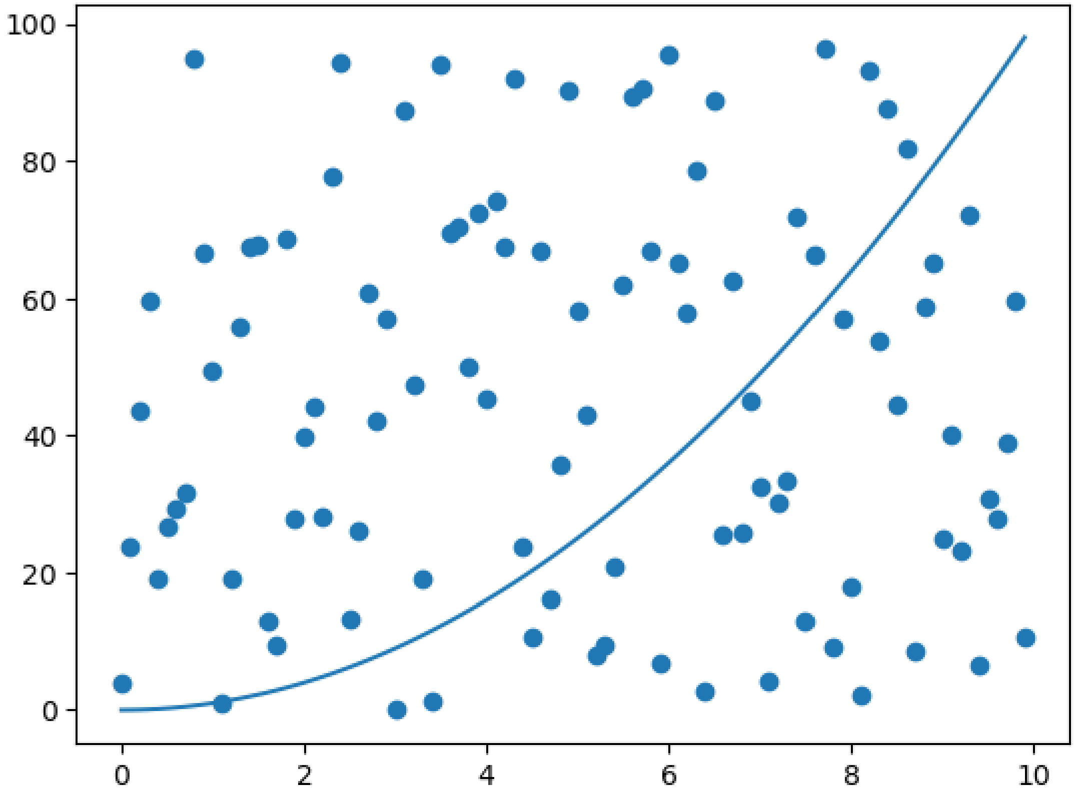


Рисунок 3.5. Реализация метода Монте-Карло для параболической функции.

Несложно заметить, что интеграл заданной функции будет иметь значение:

$$\int_0^{10} x^2 dx = \frac{x^3}{3} = 333{,}33$$

При значении N = 100 получаем значение интеграла 362.637, то есть погрешность 8,8%. При N = 1000 получится значение 335.33, то есть погрешность 0,6%. При N = 10000 получим значение 334.03, то есть погрешность 0,21%. Получается, что при увеличении N данный метод дает более точный результат.

Примечание: при самостоятельном подсчете могут получиться иные значения, потому что используется команда рандомайзера.

Во многих задачах физики, математики, астрономии, экономики, теории управлений и других, когда поведение функции задается сложными аналитическими функциями, то стохастический метод подсчета интеграла является более предпочтительным, чем классические методы. Пример использования метода Монте-Карло в астрофизических измерениях мы рассмотрим в следующих главах.

Задача Бюффона

Одним из классических примеров метода Монте-Карло, доступных для учащихся является задача Бюффона о падении иглы. Задача ставится следующим образом:

«Определите число Пи из теории вероятности пересечения линии и зубочистки при падении. Для этого разметьте лист бумаги на полоски с шириной, соответствующей длине зубочистки. Возьмите стопкой 10 зубочисток (или иголок) и с высоты 25-40 см уроните их на лист. Посчитайте число пересечений зубочисток и размеченных линий. Повторите эксперимент N раз».

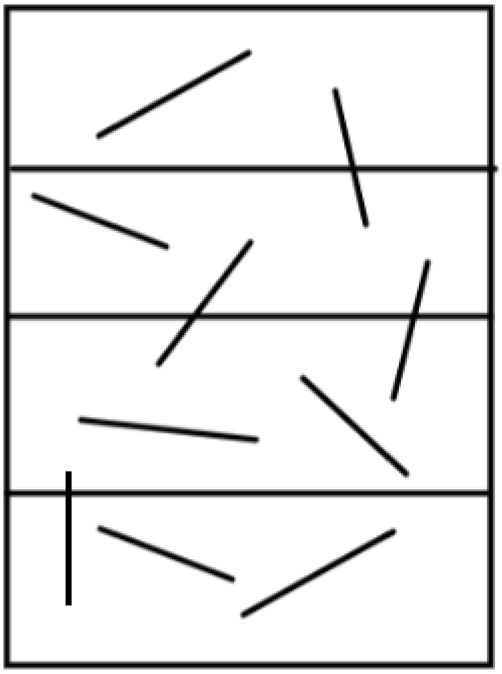


Рисунок 3.6. Пример проведения опыта Бюффона по бросанию зубочисток. Число пересечений на данном рисунке равно 4.

После проведения данного эксперимента необходимо построить гистограмму распределения количества пересечений зубочисток и вычислить среднее значение. Если поделить это число на 10, то получится экспериментальная вероятность пересечения зубочистки и линии $p_э$.

Если подойти к этой задачи теоретически, то вероятность пересечения при заданном угле будет задаваться |sin a|, как отношение длин. А вероятность, что зубочистка упадет в диапазоне Δa угла составляет Δa/2π. Итоговая теоретическая вероятность пересечения:

$$p = \int_0^{2\pi} \quad |sin\ a|\ \frac{d\ a}{2\pi} = \frac{2}{\pi}$$

Отсюда можно определить экспериментальное значение числа Пи без каких-либо измерительных приборов, а сугубо из теории вероятности. По сути с помощью стохастического метода определяется значение заданного определенного интеграла. В этой же задаче можно оценить погрешность проведенного эксперимента, с помощью среднеквадратичного отклонения:

$$\Delta\pi = \sqrt{\frac{\sum_{i=1}^{N} \quad (n_i - \underline{n})^2}{N}}$$

Корреляция данных

Теперь рассмотрим случай, когда есть множество параметров, от которых может зависеть величина. Представьте себе набор машин, которые есть на Земле. У этих машин есть множество параметров: масса, год выпуска, форма, цвет, прослушиваемая музыка во время езды, количество цилиндров в двигателе и многие другие. Все эти данные хранятся в единой таблице и ваша задача понять от каких параметров и каким образом будет зависеть, например, расход топлива на 100 километров езды. Становится очевидно, что например от цвета и музыки расход не должен зависеть, а от массы и количества цилиндров будет зависеть. Но будет ли зависеть от года выпуска или формы? Чтобы понять это потребуется разобраться с основами статистического анализа.

Допустим, что у нас есть две случайные величины X и Y для каждого измерения. Первым шагом необходимо построить X(Y) и посмотреть есть ли какая-то очевидная зависимость между X и Y:

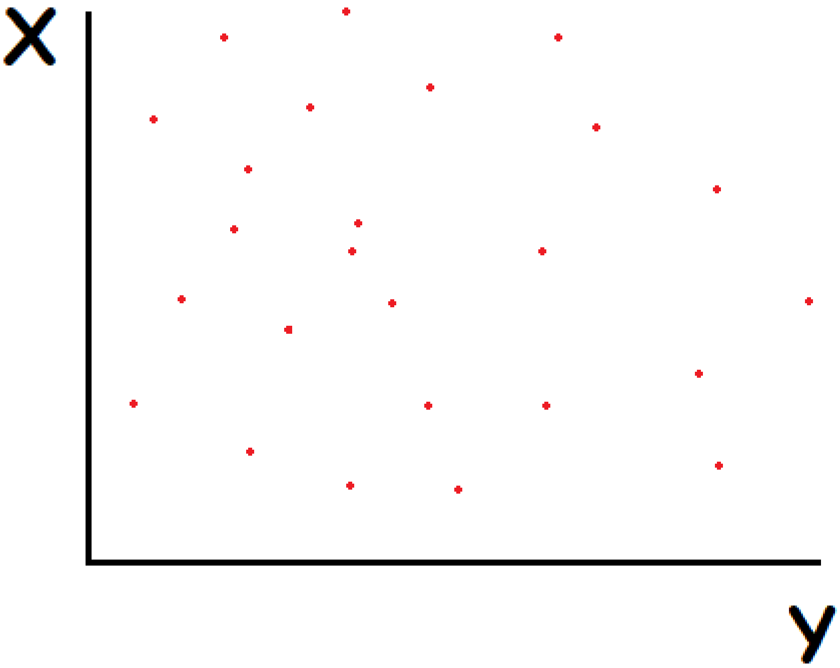


Рисунок 3.7. Отсутствие корреляции между X и Y

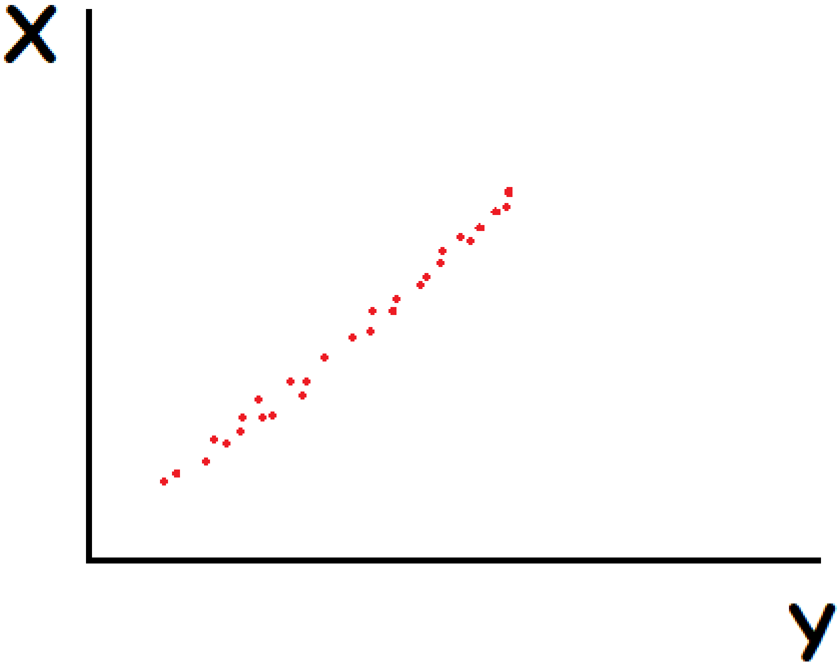


Рисунок 3.8. Наличие корреляции между X и Y

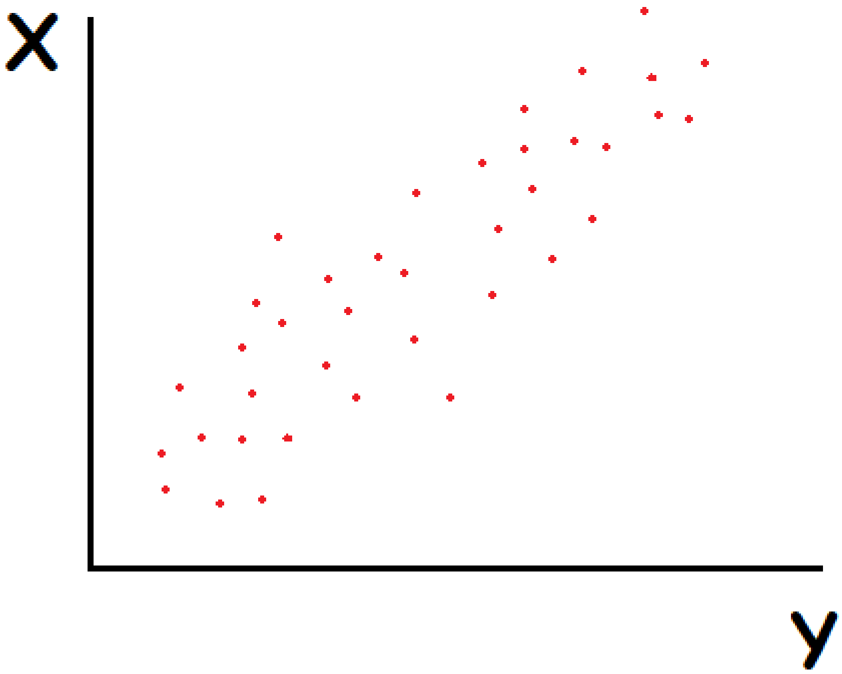


Рисунок 3.9. Вероятная зависимость X и Y

Так как понять, что какая-то случайная величина X зависит от случайной величины Y? Для этого нам потребуется задать параметр, который говорит о том зависят величины друг от друга или нет. В качестве такого параметра используется линейный коэффициент корреляции или коэффициент корреляции Пирсона:

$$r_{XY} = \frac{\sum_{i=1}^{N} (x_i - x_{ср})(y_i - y_{ср})}{\sqrt{\sum_{i=1}^{N} |x_i - x_{ср}| \sum_{i=1}^{N} |y_{i_i} - y_{ср}|}}$$

$$x_{ср} = \frac{\sum_{i=1}^{N} x_i}{N}$$

$$y_{ср} = \frac{\sum_{i=1}^{N} y_i}{N}$$

Где N - число случайных величин X и Y.

Далее в зависимости от числа параметров от которых зависит данная система и уровня значимости определяется приемлемый диапазон критерия Пирсона. На базе Python есть встроенные функции для сравнения вашего критерия с приемлемым диапазоном. Более подробно о данном критерии и характерных значениях можно узнать в рамках изучения статистического анализа.

Глава 4. ПЗС-матрица

Все современные научные астрономические фотографии неба осуществляются с помощью ПЗС-матриц (сокр. от «прибор с зарядовой связью»), которые состоят из матрицы субпикселей. Общая идея субпикселей вытекает из теории фотоэффекта и принципа работы полупроводникового диода. При попадании фотонов на чувствительный электрод из него вырываются электроны, которые создают разницу потенциалов на диоде. Далее данный сигнал считывается с каждого субпикселя и передается в компьютер. В результате в памяти компьютера хранится матрица значений, которые соответствуют интенсивности освещения на каждом участке ПЗС-матрицы.

Определение центра звезды

Так как единицы измерения сигнала не играют особой роли, для небольшого упрощения все значения из обучающих примеров будут представлены в целых числах. При обучающей работе можно их интерпретировать как просто число фотонов, которые падают на субпиксель ПЗС-матрицы.

Рассмотрим такую задачу:

«Перед вами фотография некоторого участка неба в виде матрицы значений яркости ПЗС-матрицы. В первой строке и столбце указаны условные координаты на небе. Необходимо определить положение центра звезды на фотографии в условных координатах.»

Вначале рассмотрим такой пример фотографии:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	9	10	6	12	8	10	9	9	14	10	11	9	7	6	7	7	10	7	13	8
2	14	12	12	9	12	9	7	7	10	6	6	13	6	14	14	7	15	8	10	9
3	14	13	11	11	12	11	11	12	7	9	14	7	8	10	12	15	13	9	9	10
4	8	13	13	14	14	12	11	8	11	14	8	11	15	9	7	10	10	8	13	15
5	8	11	15	9	7	10	10	8	7	12	12	14	10	7	11	13	8	7	13	12
6	12	14	10	7	11	13	8	7	10	11	11	9	13	13	6	6	9	13	14	7
7	11	9	13	13	6	6	9	13	12	14	10	15	6	8	10	10	9	6	7	10
8	10	15	6	8	10	10	8	11	15	9	7	10	10	8	13	12	11	13	9	10
9	9	15	10	11	13	12	12	14	10	7	11	13	8	7	15	7	9	9	13	7
10	7	8	7	7	15	7	11	9	13	13	6	6	9	13	14	12	7	14	13	9
11	13	11	15	11	14	12	400	15	6	8	10	10	9	6	14	10	6	12	9	7
12	9	9	7	13	14	10	9	15	10	11	13	12	11	13	12	8	14	9	8	9
13	11	12	11	15	12	8	7	8	7	7	15	7	9	9	14	9	12	8	14	14
14	11	11	11	13	14	9	13	11	15	11	14	12	7	14	14	7	10	12	7	15
15	8	11	10	12	14	7	9	9	7	13	14	10	6	12	15	11	12	6	12	6
16	10	9	9	7	8	10	11	12	11	15	12	8	14	9	12	12	12	8	9	10
17	13	9	8	14	12	14	11	11	11	13	14	9	12	8	14	14	10	7	6	8
18	10	11	11	14	8	9	8	11	10	12	14	7	10	12	9	13	14	7	15	10
19	11	14	14	8	13	7	13	7	15	6	15	8	8	14	9	13	10	14	12	11
20	13	7	12	7	10	11	11	12	9	9	9	14	7	7	9	8	10	14	11	7

Рисунок 4.1. Матрица значений яркостей ПЗС-матрицы

Не сложно заметить, что определенная ячейка матрицы сильно выделяется значением по сравнению с остальными, значит это и есть звезда. В данном случае положение центра звезды будет (7; 11), но точнее чем один пиксель координаты получить невозможно.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	9	10	6	12	8	10	9	9	14	10	11	9	7	6	7	7	10	7	13	8
2	14	12	12	9	12	9	7	7	10	6	6	13	6	14	14	7	15	8	10	9
3	14	13	11	11	12	11	11	12	7	9	14	7	8	10	12	15	13	9	9	10
4	8	13	13	14	14	12	11	8	11	14	8	11	15	9	7	10	10	8	13	15
5	8	11	15	9	7	10	10	8	7	12	12	14	10	7	11	13	8	7	13	12
6	12	14	10	7	11	13	8	7	10	11	11	9	13	13	6	6	9	13	14	7
7	11	9	13	13	6	6	9	13	12	14	10	15	6	8	10	10	9	6	7	10
8	10	15	6	8	10	10	8	11	15	9	7	10	10	8	13	12	11	13	9	10
9	9	15	10	11	13	12	12	14	10	7	11	13	8	7	15	7	9	9	13	7
10	7	8	7	7	15	7	11	9	13	13	6	6	9	13	14	12	7	14	13	9
11	13	11	15	11	14	12	400	15	6	8	10	10	9	6	14	10	6	12	9	7
12	9	9	7	13	14	10	9	15	10	11	13	12	11	13	12	8	14	9	8	9
13	11	12	11	15	12	8	7	8	7	7	15	7	9	9	14	9	12	8	14	14
14	11	11	11	13	14	9	13	11	15	11	14	12	7	14	14	7	10	12	7	15
15	8	11	10	12	14	7	9	9	7	13	14	10	6	12	15	11	12	6	12	6
16	10	9	9	7	8	10	11	12	11	15	12	8	14	9	12	12	12	8	9	10
17	13	9	8	14	12	14	11	11	11	13	14	9	12	8	14	14	10	7	6	8
18	10	11	11	14	8	9	8	11	10	12	14	7	10	12	9	13	14	7	15	10
19	11	14	14	8	13	7	13	7	15	6	15	8	8	14	9	13	10	14	12	11
20	13	7	12	7	10	11	11	12	9	9	9	14	7	7	9	8	10	14	11	7

Рисунок 4.2. Звезда размером один пиксель на ПЗС-матрице

Теперь рассмотрим более сложный пример, когда фотография будет занимать не один пиксель ПЗС-матрицы, а несколько, в связи с увеличением концентрации субпикселей.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	9	10	6	12	8	10	9	9	14	10	11	9	7	6	7	7	10	7	13	8
2	14	12	12	9	12	9	7	7	10	6	6	13	6	14	14	7	15	8	10	9
3	14	13	11	11	12	11	11	12	7	9	14	7	8	10	12	15	13	9	9	10
4	8	13	13	14	14	12	11	8	11	14	8	11	15	9	7	10	10	8	13	15
5	10	7	9	11	11	7	8	13	7	12	12	14	10	7	11	13	8	7	13	12
6	15	6	7	11	56	42	13	9	10	11	11	9	13	13	6	6	9	13	14	7
7	14	11	11	241	312	284	192	39	12	14	10	15	6	8	10	10	9	6	7	10
8	7	13	328	410	530	461	305	218	36	7	9	15	10	11	13	12	11	13	9	10
9	6	175	408	632	789	631	482	254	234	12	7	8	7	7	15	7	9	9	13	7
10	44	231	473	725	845	732	501	309	121	7	13	11	15	11	14	12	7	14	13	9
11	34	183	412	680	701	677	452	278	88	7	9	9	7	13	14	10	6	12	9	7
12	78	131	349	493	521	478	342	165	11	12	11	12	11	15	12	8	14	9	8	9
13	9	76	222	267	320	298	187	55	11	7	11	11	11	13	14	9	12	8	14	14
14	13	12	118	157	182	145	110	6	9	9	8	11	10	12	14	7	10	12	7	15
15	13	6	14	45	59	38	14	13	6	11	7	6	7	7	15	11	12	6	12	6
16	10	9	9	7	8	10	12	9	8	14	13	13	14	11	12	12	12	8	9	10
17	13	9	8	14	12	14	13	8	14	13	12	7	13	9	14	14	10	7	6	8
18	10	11	11	14	8	9	8	12	13	12	13	9	6	14	9	13	14	7	15	10
19	11	14	14	8	13	7	13	7	15	6	15	8	8	14	9	13	10	14	12	11
20	13	7	12	7	10	11	11	12	9	9	9	14	7	7	9	8	10	14	11	7

Рисунок 4.3. Матрица значений яркостей ПЗС-матрицы для звезды большего размера

В данном случае звезда «размазана» уже на несколько элементов массива.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	9	10	6	12	8	10	9	9	14	10	11	9	7	6	7	7	10	7	13	8
2	14	12	12	9	12	9	7	7	10	6	6	13	6	14	14	7	15	8	10	9
3	14	13	11	11	12	11	11	12	7	9	14	7	8	10	12	15	13	9	9	10
4	8	13	13	14	14	12	11	8	11	14	8	11	15	9	7	10	10	8	13	15
5	10	7	9	11	11	7	8	13	7	12	12	14	10	7	11	13	8	7	13	12
6	15	6	7	11	56	42	13	9	10	11	11	9	13	13	6	6	9	13	14	7
7	14	11	11	241	312	284	192	39	12	14	10	15	6	8	10	10	9	6	7	10
8	7	13	328	410	530	461	305	218	36	7	9	15	10	11	13	12	11	13	9	10
9	6	175	408	632	789	631	482	254	234	12	7	8	7	7	15	7	9	9	13	7
10	44	231	473	725	845	732	501	309	121	7	13	11	15	11	14	12	7	14	13	9
11	34	183	412	680	701	677	452	278	88	7	9	9	7	13	14	10	6	12	9	7
12	78	131	349	493	521	478	342	165	11	12	11	12	11	15	12	8	14	9	8	9
13	9	76	222	267	320	298	187	55	11	7	11	11	11	13	14	9	12	8	14	14
14	13	12	118	157	182	145	110	6	9	9	8	11	10	12	14	7	10	12	7	15
15	13	6	14	45	59	38	14	13	6	11	7	6	7	7	15	11	12	6	12	6
16	10	9	9	7	8	10	12	9	8	14	13	13	14	11	12	12	12	8	9	10
17	13	9	8	14	12	14	13	8	14	13	12	7	13	9	14	14	10	7	6	8
18	10	11	11	14	8	9	8	12	13	12	13	9	6	14	9	13	14	7	15	10
19	11	14	14	8	13	7	13	7	15	6	15	8	8	14	9	13	10	14	12	11
20	13	7	12	7	10	11	11	12	9	9	9	14	7	7	9	8	10	14	11	7

Рисунок 4.4. Выделение ячеек матрицы, принадлежащие звезде

Понятно, что центр звезды располагается примерно около 5-6 по горизонтали и 10-11 по вертикали. Также можно было бы определить положение центра звезды, как среднеарифметическое значений матрицы по обеим осям, но легко заметить, что чем ближе к краям, тем меньше значения яркости. Значит краевые пиксели должны вносить меньший вклад. Поэтому более честным будет расчет среднеарифметического с учетом веса:

$$x_ц = \frac{\sum_{i=1}^{N} \quad x_i m_i}{\sum_{i=1}^{N} \quad m_i}$$

$$y_ц = \frac{\sum_{i=1}^{N} \quad y_{i_i} m_i}{\sum_{i=1}^{N} \quad m_i}$$

Наиболее понятной и известной для учащихся должна быть аналогия с формулой центра масс. По сути в данной задаче мы вычисляем центр яркости, поэтому не удивительно, что в формуле центра масс происходит замена массы на яркость, то есть значение элемента в массиве. Если посчитать данные значения (в зависимости от уровня подготовки здесь можно воспользоваться калькулятором/Excel/Python), то получится положение центра звезды (5,18; 10,24).

Пример реализации в Python

```
data        #исходный массив star.csv
sum = 0
sum_xm = 0
sum_ym = 0

for i in range(1, n-1):
    for j in range(1, n-1):
        if star[i][j] > k:
            sum_xm += star[i][j]*j
            sum += star[i][j]
            sum_ym += star[i][j]*i

print(sum_xm/sum )
print(sum_ym/sum )
```

В данном случае критерий k, по которому определяется является элемент массива звездой или фоном неба, можно определить самостоятельно, исходя из визуального анализа массива. В случае более честного и программного анализа можно воспользоваться правилом трёх сигма для нормального распределения. Для этого нужно выделить участок неба без звезды и рассчитать среднеквадратичное отклонение для него:

```
z = 0
N = 0

for i in range(1, n-1):
    for j in range(10, n-1):
        N += 1
        z += star[i][j]

z_mean = z/N

sum_sigma = 0
for i in range(1, n-1):
    for j in range(10, n-1):
        sum_sigma += (star[i][j]-z_mean)**2

sigma = np.sqrt(sum_sigma/(N-1))

k = z_mean + 3*sigma
```

Главным выводом данной задачи является возможность определение положение центра звезды с точностью выше, чем один пиксель. В действительности именно использование фотографий и ПЗС-матриц позволило существенно увеличить точность определения координат многих объектов.

Фон неба

Теперь давайте учтем в предыдущей задаче такое понятие, как фон неба. В целом любой телескоп ограничен в светочувствительности и более далекие и слабые звезды или другие источники излучения будут создавать некоторую яркость даже в отсутствии видимых звезд. В случае наличия звезды эта яркость наложится на её излучение, поэтому при более честном подсчете центра звезды необходимо вычесть среднее значение фона неба из яркости звезды.

Определим среднее значение фона неба, как среднеарифметическое значение:

```
sum_sky = 0
sum_sky_N = 0

for i in range(1, n-1):
    for j in range(1, n-1):
        if star[i][j] < k:
            sum_sky_N += 1
            sum_sky += star[i][j]

fon = sum_sky/sum_sky_N
```

Теперь вычтем среднее значение фона неба из всех элементов массива звезды:

```
sum2 = 0
sumx = 0
sumy = 0

for i in range(1, n-1):
    for j in range(1, n-1):
        if star[i][j] > k:
            sumy += star[i][j]*i - fon
            sumx += star[i][j]*j - fon
            sum2 += star[i][j] - fon

print(fon)
print(sumx/sum2)
print(sumy/sum2)
```

Получим положение центра звезды (5,33; 10,57), что отличается от первоначального ответа. Можно также визуально построить данную звезду и отметить её центр.

```
plt.figure()
plt.imshow(star, cmap='bone')
plt.colorbar()
plt.scatter(sumx/sum2, sumy/sum2)
```

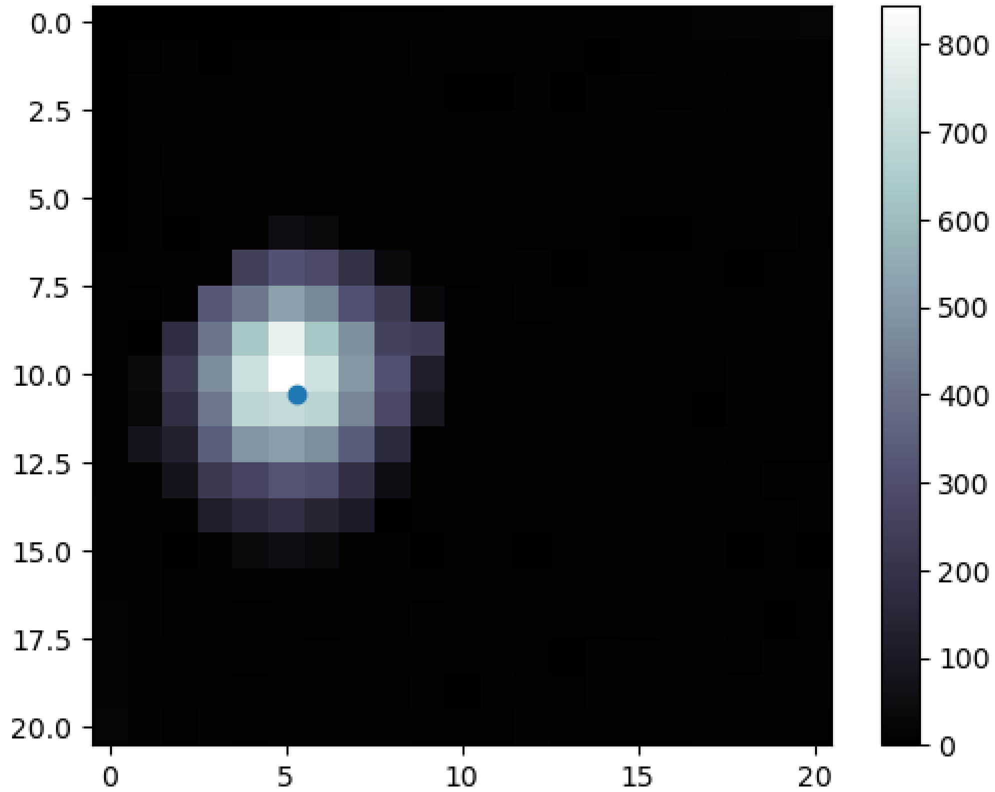


Рисунок 4.5. Построение изображения звезды в Matplotlib

Методика данной задачи является универсальной и применяется не только в астрономии, но и во многих других областях. Например можно вычислять более точным образом положение микроорганизмов в растворах и, как следствие, более точно определять их концентрацию. Также похожие методики используются в географии при определении распределении популяции определенных биологических видов.

Глава 5. Подсчет числа элементов

Задача о подсчете числа элементов на фотографии является одной из основной в астрономии и не только. Данная задача появляется при изучении распределении вещества и звезд внутри галактик, при анализе распределения галактик в структуре Вселенной. Также это способ определения числа Вольфа по солнечным пятнам, которое показывает уровень солнечной активности. Сложность задачи зависит от сгруппированности данных объектов и оптических ограничений, связанных с разрешающей способностью телескопа и учетом фона неба, для различения накладывающихся объектов друг на друга.

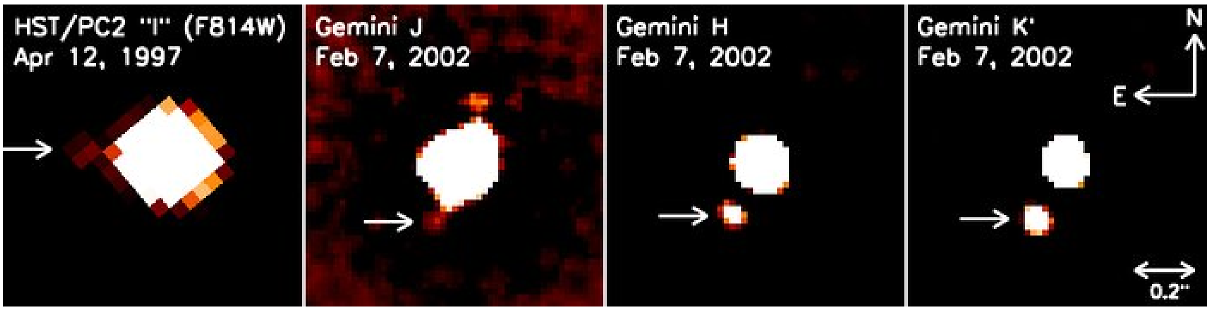


Рисунок 5.1.Фотография пары коричневый карлик-звезда, полученная в обсерватории Gemini. При более качественном анализе можно обнаружить вторую компоненту звезды.

Подсчет числа звёзд размером один пиксель

Для начала рассмотрим задачу по определению числа звезд и их положения, если они располагаются в одной ячейке матрицы. Для этого можно также определить критерий отличия звезды от фона, как в предыдущей задаче:

```
sky     #исходный массив sky.csv
z = 0
N = 0

for i in range(30, n-1):
   for j in range(30, n-1):
      N += 1
      z += sky[i][j]

z_mean = z/N

sum_sigma = 0
for i in range(30, n-1):
   for j in range(30, n-1):
      sum_sigma += (sky[i][j]-z_mean)**2

sigma = np.sqrt(sum_sigma/(N-1))
```

```
k = z_mean + 3*sigma
```

Далее определяем число звезд на данной “фотографии” из файла “sky.csv”:

```
stars = 0

for j in range(0, n-1):
    for i in range(0, n-1):
        if sky[i][j] > k:
            stars += 1

print("число звезд = ", stars)
```

В нашем примере получается 9 звезд. Также можно определить положения данных звезд на небе:

```
for i in range(0, n-1):
    for j in range(0, n-1):
        if sky[i][j] > k:
            print(i+1, j+1)
```

И построить их визуально:

```
plt.figure()
plt.imshow(sky, cmap='bone')
plt.colorbar()
```

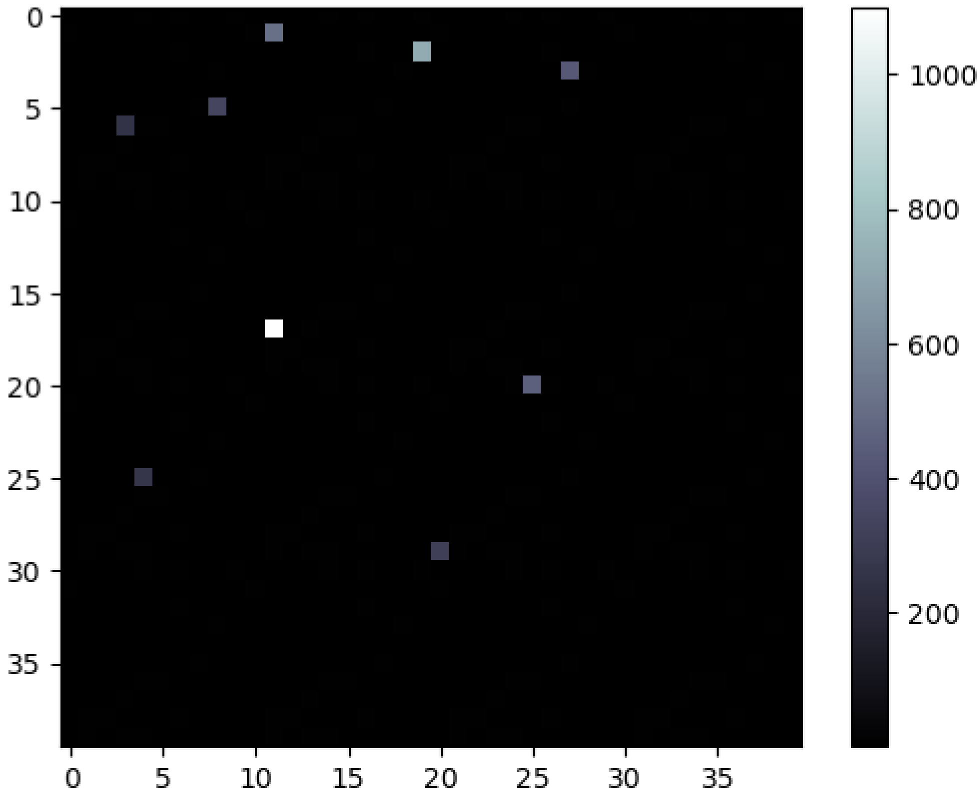


Рисунок 5.2. Построение звёзд из файла “sky.csv” в Matplotlib

Подсчет числа звёзд размером больше одного пикселя

Теперь усложним задачу и добавим несколько звёзд размером больше одного пикселя. В случае использования предыдущего кода, он будет выдавать завышенное значение звёзд. Чтобы избежать этого мы можем вспомнить, что наши циклы for проходят справа налево и сверху вниз. Поэтому при попадании на звезду мы должны сравнить соседние клетки и критерием k, который является средним значением фона фотографии. Если они тоже являются звездой, то мы должны уменьшить число звезд на один.

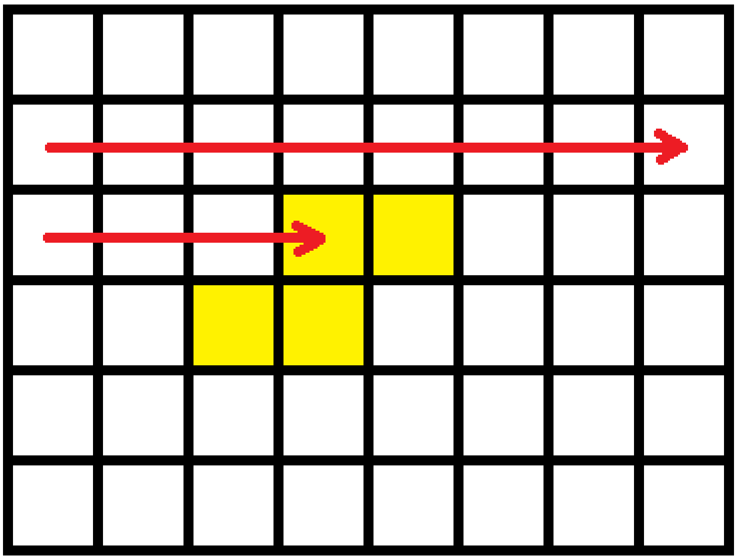


Рисунок 5.3. Алгоритм поиска начала звезды в Python

К моменту, когда подойдем к левой нижней границе звезды вычитание происходить уже не должно. Поэтому будем сравнивать только выделенные ячейки:

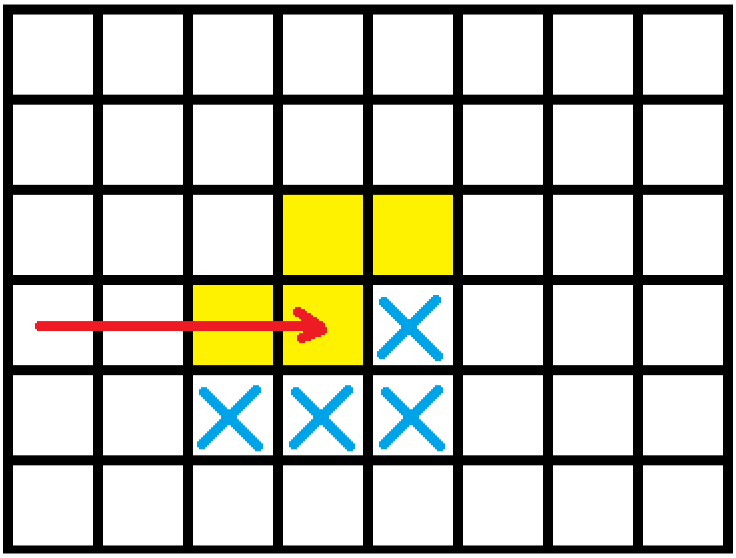


Рисунок 5.4. Алгоритм поиска конца звезды в Python

В результате получим следующий алгоритм:

```
sky     #исходный массив sky2.csv
stars = 0

for j in range(0, n-1):
   for i in range(0, n-1):
```

```
    if sky[i][j] > k:
        stars += 1
        if sky[i][j+1] > k or sky[i+1][j] > k or sky[i+1][j+1] > k or sky [j-1][i+1] > k:
            stars -= 1

print("число звезд = ", stars)
```

```
plt.figure()
plt.imshow(sky, cmap='bone')
plt.colorbar()
```

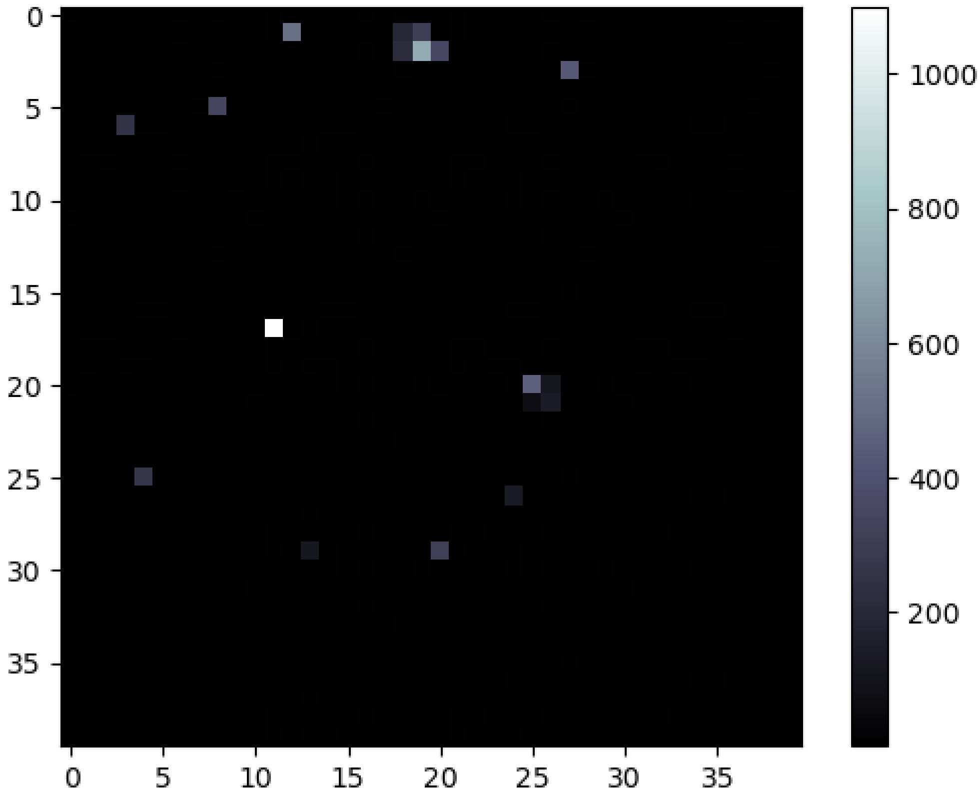


Рисунок 5.5. Построение звёзд из файла “sky2.csv” в Matplotlib

Также можно определить координаты звезд, объединив эту задачу с задачей поиска центра звезды из предыдущей главы. И получим координаты 11 звёзд:

Таблица 4. Координаты звёзд из файла “sky2.csv”

	Горизонтальная ось	Вертикальная ось
1	4	7
2	5	26

3	9	6
4	12	18
5	13	2
6	14	30
7	28	4
8	21	30
9	25	27
10	19,49	1,74
11	26,62	21,30

Глава 6. Fits-файлы

Одной из особенностей анализа космоса в астрономии является использование специального формата изображений – *.fits* (сокр. от «Flexible Image Transport System»). Помимо информации об изображении в данном формате добавляются дополнительные метаданные, включающие информацию о телескопе, координатах, происхождении и любой дополнительной информации, которую автор фотографии посчитал нужным добавить. Для работы с fits-файлами нам потребуется библиотека astropy в Python.

Примеры fits-файлов

Большая часть fits-файлов хранится в общем доступе на серверах NASA, Google, Amazon и крупнейших телескопов. Для начала попробуем построить скачанный файл HorseHead.fits с репозитория astropy, в котором хранится изображение туманности Конская Голова в созвездии Ориона:

```
import matplotlib.pyplot as plt
from astropy.io import fits

horsehead = fits.open('Downloads/HorseHead.fits')
horsehead.info()
image_data = horsehead[0].data

plt.figure()
plt.imshow(image_data, cmap='magma')
plt.colorbar()
```

```
No.    Name         Ver    Type        Cards   Dimensions    Format
  0  PRIMARY          1 PrimaryHDU      161   (891, 893)    int16
  1  er.mask          1 TableHDU         25   1600R x 4C    [F6.2, F6.2, F6.2, F6.2]
```

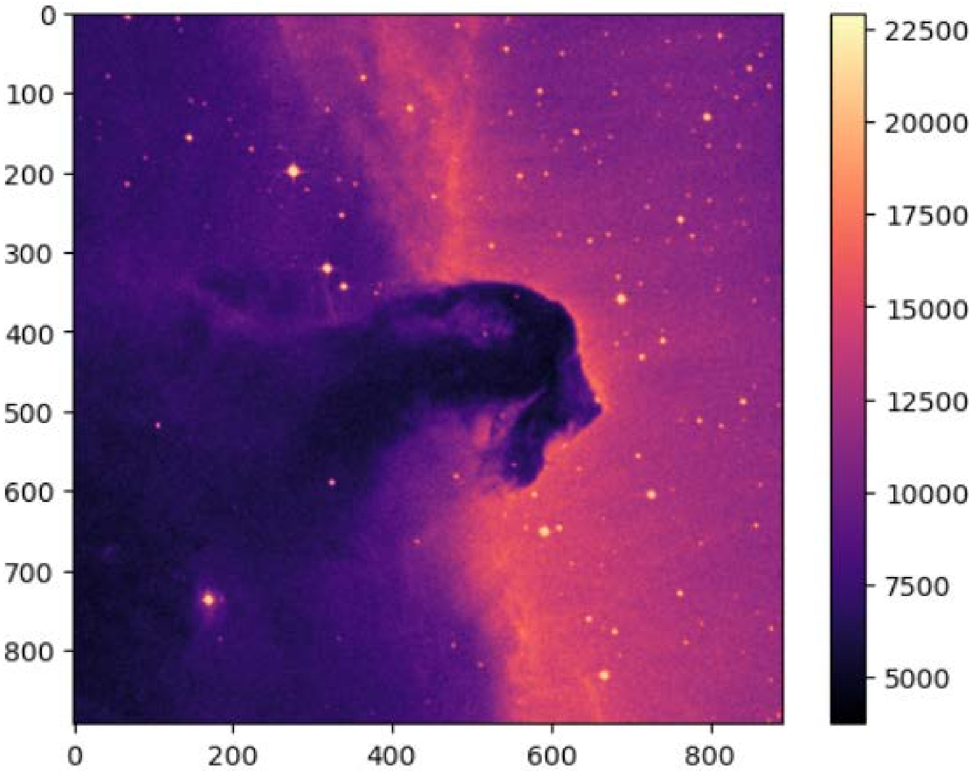


Рисунок 6.1. Построение файла HorseHead.fits в Python

Здесь нас интересует только первый массив PRIMARY, в котором хранится данное изображение. Второй массив представлен в виде 1600 строк и 4 столбцов. В качестве цвета градиента можно использовать любой другой из библиотеки matplotlib[3].

Также можно открывать fits-файлы напрямую из репозитория различных серверов. В качестве примера построим изображение из репозитория astropy шарового звездного скопления M67 в созвездии Рака:

```
from astropy.utils.data import get_pkg_data_filename
image_file = get_pkg_data_filename('photometry/M6707HH.fits')

fits.info(image_file)
image_data = fits.getdata(image_file, ext=0)
plt.figure()
plt.imshow(image_data, cmap='bone')
plt.colorbar()
```

[3] https://matplotlib.org/stable/users/explain/colors/colormaps.html

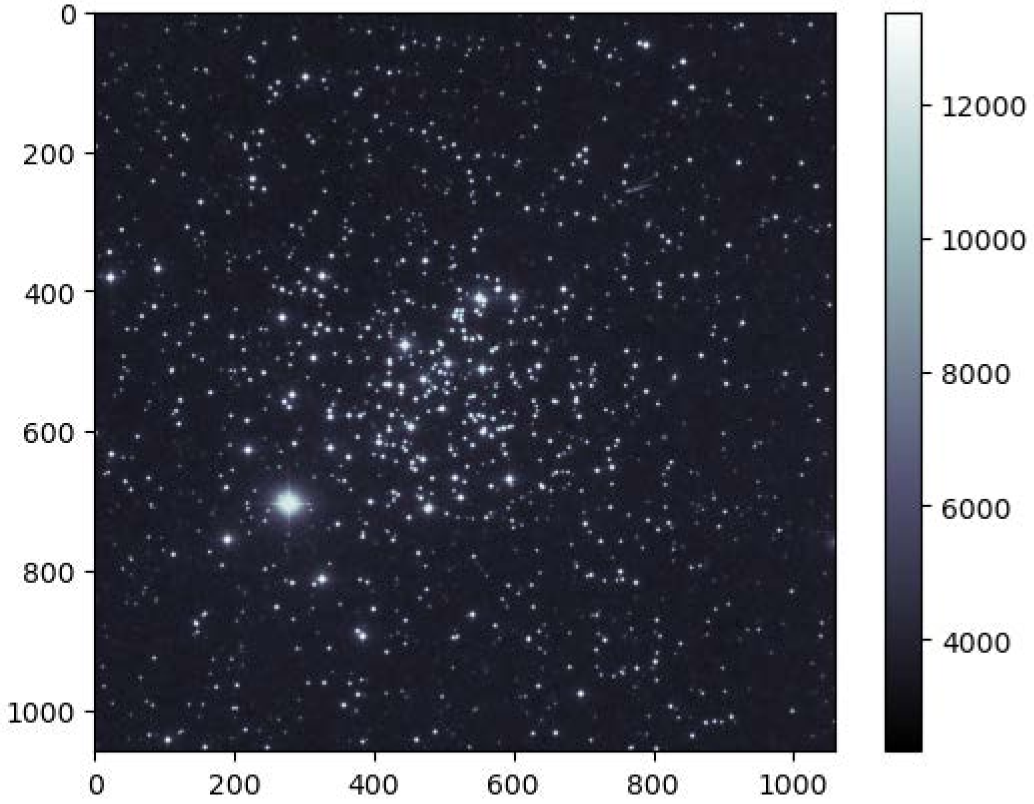


Рисунок 6.2. Построение скопления M67 в Python

Построение фрагмента фотографии в указанных координатах

Также если нас интересует конкретный фрагмент фотографии, то мы можем задать построение только указанных координат. Построим фрагмент изображения в координатах {100; 200} по горизонтальной и вертикальной оси с помощью функции *section*:

```
with fits.open(image_file, use_fsspec=True, fsspec_kwargs={"anon": True}) as hdul:
  cutout = hdul[0].section[100:200, 100:200]

plt.imshow(cutout, cmap='bone')
plt.colorbar()
```

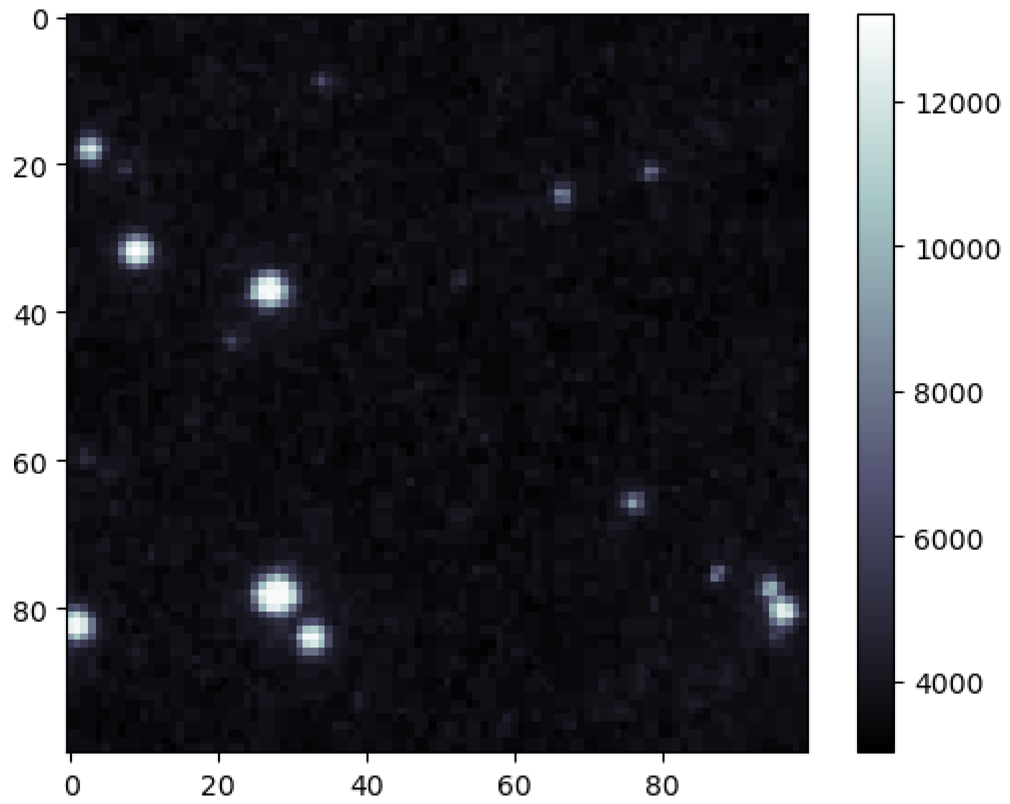


Рисунок 6.3. Построение заданного фрагмента fits-изображения

Для данного фрагмента можно реализовать алгоритм подсчета числа звёзд, который был описан в предыдущей главе.

```
import numpy as np
sky = cutout
n = np.count_nonzero(sky[1])
z = 0
N = 0

for i in range(0, n-1):
    for j in range(0, n-1):
        N += 1
        z += sky[i][j]

z_mean = z/N

sum_sigma = 0
for i in range(0, n-1):
    for j in range(0, n-1):
        sum_sigma += (sky[i][j]-z_mean)**2

sigma = np.sqrt(sum_sigma/(N-1))
```

```
k = z_mean + 3*sigma

stars = 0
for j in range(0, n-1):
   for i in range(0, n-1):
      if sky[i][j] > k:
         stars += 1
         if sky[i][j+1] > k or sky[i+1][j] > k or sky[i+1][j+1] > k or sky [j-1][i+1] > k:
            stars -= 1

print("число звезд = ", stars)
```

Алгоритм выдает 18 звёзд на фрагменте. Визуально здесь можно распознать 16 звёзд, оставшиеся 2 плохо различимы на общем фоне.

Анализ больших файлов

Также особенностью fits-форматов является возможность их удаленного фрагментированного анализа со сторонних серверов с невысокой загрузкой компьютера. Файл *“j8pu0y010_drc.fits”* получен с телескопа Хаббла[4] и весит свыше 256 Мб, но при этом время выполнения операций будет даже на не мощном компьютере достаточно малым. Изучим данный файл.

```
url =
"https://mast.stsci.edu/api/v0.1/Download/file/?uri=mast:HST/product/j8pu0y010_drc.fits"

fits.info(url)
```

```
No.    Name         Ver    Type      Cards   Dimensions   Format
  0  PRIMARY          1 PrimaryHDU     839   ()      
  1  SCI              1 ImageHDU        81   (4223, 4421)   float32   
  2  WHT              1 ImageHDU        44   (4223, 4421)   float32   
  3  CTX              1 ImageHDU        37   (4223, 4421)   int32   
  4  HDRTAB           1 BinTableHDU    593   8R x 292C   [9A, 3A, K, D, D, D, D, D, D, D, D, D, D, D, D, D, K, 8A, 9A, 7A, 18A, 4
A, D, D, D, D, 3A, D, D, D, D, D, D, D, D, D, D, D, D, K, 8A, 23A, D, D, D, D, K, K, K, 8A, K, 23A, 9A, 20A, K, 4A, K, K, K, K,
K, K, 23A, D, D, D, D, K, K, 3A, 3A, 4A, 4A, L, D, D, D, 3A, 1A, K, D, D, D, 13A, 3A, 4A, 4A, 12A, 12A, 23A, 8A, 23A, 10A, 10A,
D, D, 3A, 3A, 23A, 4A, 8A, 7A, 23A, D, K, D, 6A, 9A, 8A, D, D, L, 4A, 18A, 3A, K, 7A, 5A, 3A, D, 13A, 8A, 4A, 3A, L, K, L, K,
L, K, K, D, D, D, D, D, D, 3A, 1A, D, 23A, D, D, D, 3A, 23A, L, 1A, 3A, 1A, D, 3A, 6A, K, D, D, D, D, D, D, D, D, D, D, 23A, D,
D, D, D, 3A, D, D, D, 1A, K, K, K, K, K, K, 23A, K, 5A, 7A, D, D, D, D, D, D, D, D, D, D, D, D, D, D, D, D, D, 12A, D, 24A, 23
A, D, 3A, 8A, D, K, D, D, 6A, 4A, D, 4A, K, D, K, 8A, D, D, D, D, D, 23A, 23A, D, 8A, D, 29A, D, 3A, D, L, D, D, 3A, 8A, 8A, 2
A, D, 8A, K, 1A, 1A, 1A, 1A, D, D, D, D, D, D, 4A, D, 4A, D, 4A, K, 4A, 3A, 1A, L, K, K, 1A, D, D, D, D, K, 3A, L, L, 6A, L, D,
D, 11A, D, D, 3A, 8A, 1A, D, K, D, L, 30A, L, 5A]
```

Видим, что данный файл содержит 3 изображения (science image ['SCI'], weight image ['WHT'] и context image ['CTX']) в формате 4k качества одного и того же участка неба. Подробная расшифровка условных обозначений в fits-файлах представлена в документации Хаббла[5]. Построим фрагмент указанного изображения в первом спектре.

[4] https://hubblesite.org/home
[5] https://hst-docs.stsci.edu/acsdhb/chapter-2-acs-data-structure/2-2-acs-file-structure

```
with fits.open(url, use_fsspec=True, fsspec_kwargs={"anon": True}) as hdul:
  cutout2 = hdul[1].section[1000:1500, 2000:2500]

plt.imshow(cutout2, cmap='bone')
plt.colorbar()
```

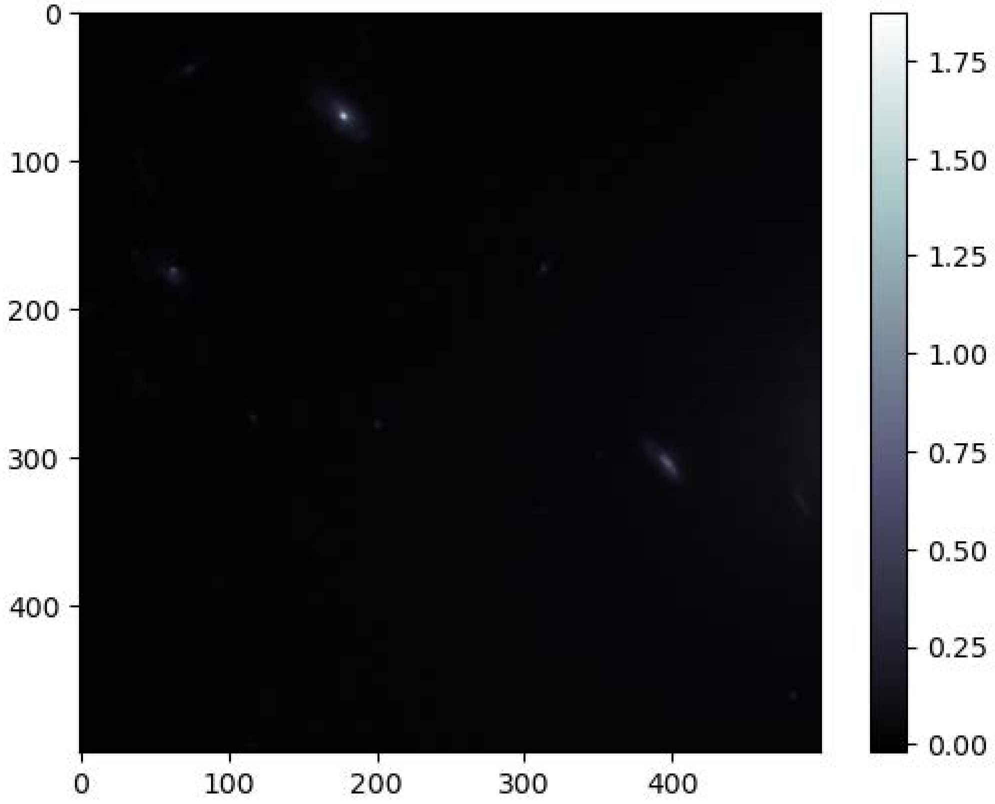


Рисунок 6.4. Построение изображения с телескопа Хаббл

Получаем изображение нескольких галактик. Если мы хотим посчитать их число, то придется вручную увеличить критерий k в нашем алгоритме, чтобы исключить некоторые не яркие фоновые звезды. При подстановке $k = 0.22$ получаем 15 галактик, что соответствует фотографии. Дальнейший анализ галактик будет представлен в следующей главе.

Глава 7. Изучение структуры объектов

В данной главе попробуем изучить какие функции доступны для изучения структур галактик и не только. Для начала вспомним, какие типы галактик существуют: эллиптические (E), линзовидные (S0), спиральные без перемычки (S), спиральные с перемычкой (SB) и неправильные (Irr).

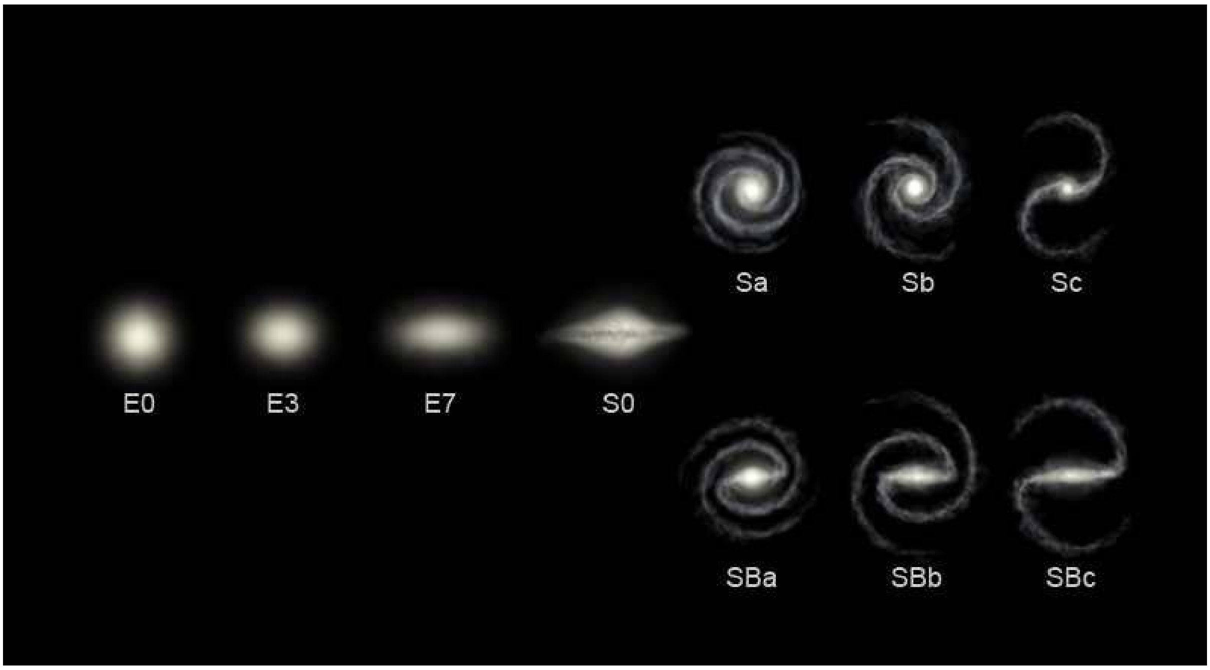


Рисунок 7.1. Классификация галактик по Хабблу

Классическая галактика обычно включает в себя ядро (компактный и массивный объект), которое окружено балджем, состоящим из множества звезд. Обычно данная часть будет самой яркой в галактике. Данная система окружена диском, который может включать в себя спиральные рукава.

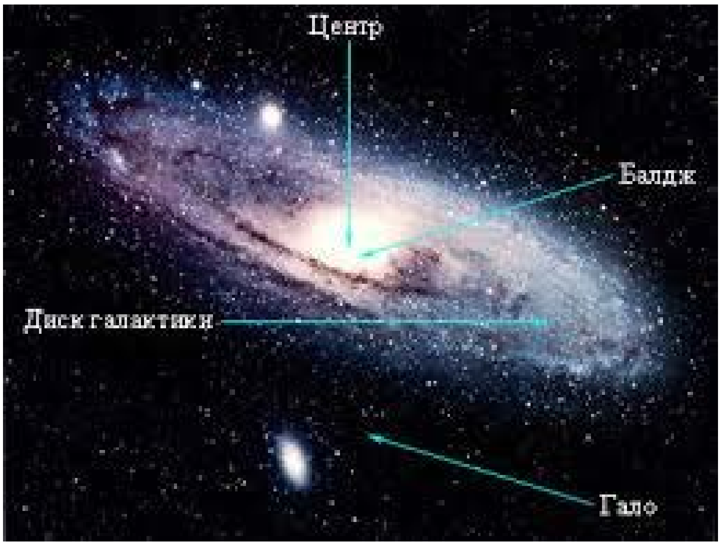


Рисунок 7.2. Структура галактики СДЕЛАТЬ СВОЮ КАРТИНКУ

Анализ структуры M31

Изучим галактику Андромеду (M31). Для этого построим ее fits-изображение:

```
import matplotlib.pyplot as plt
from astropy.io import fits
import numpy as np

image_file = 'Downloads/M31.fits'

M31 = fits.open(image_file)
M31.info()
image_data = M31[0].data

plt.imshow(image_data, cmap='bone')
plt.colorbar()
```

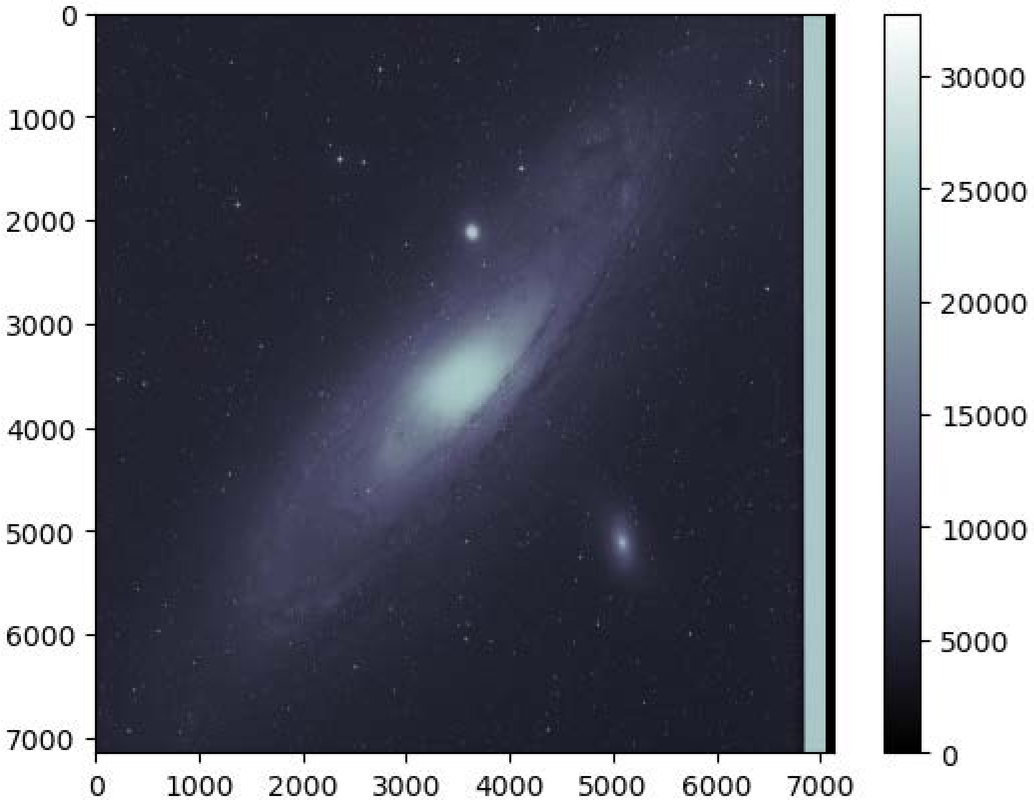


Рисунок 7.3. Построение M31 в Matplotlib

Заметим, что на фотографии присутствует полоса засветки. Необходимо её обрезать:

```
with fits.open(image_file, use_fsspec=True, fsspec_kwargs={"anon": True}) as hdul:
  cutout = hdul[0].section[0:6500, 0:6500]

plt.imshow(cutout, cmap='gray')
```

```
plt.colorbar()
```

В библиотеке matplotlib существует встроенная функция contour для выделения контуров в зависимости от значения яркости:

```
plt.imshow(cutout, cmap='gray')
plt.colorbar()
plt.contour(cutout, levels=1)
```

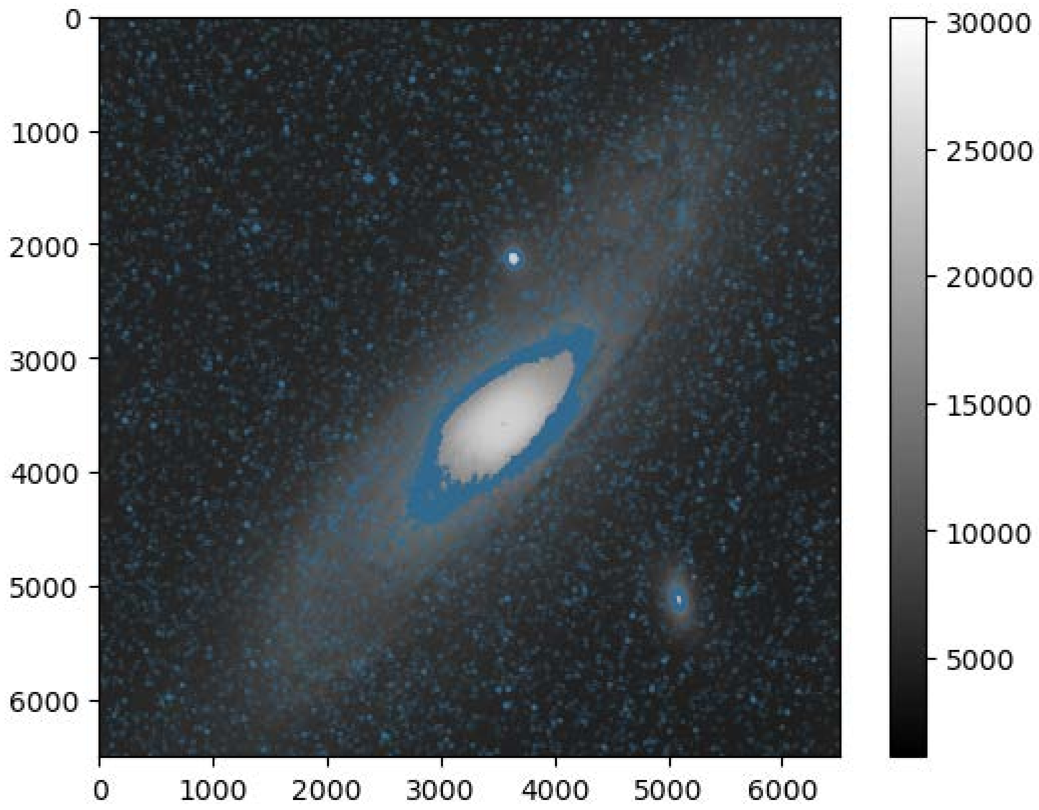


Рисунок 7.4. Выделение ядра M31 на фотографии

Как мы видим, в данном случае выделяется ядро Андромеды и соседних галактик, а также все звезды. Можно улучшить данное структурирование с помощью увеличения числа levels:

```
plt.imshow(cutout, cmap='gray')
plt.colorbar()
plt.contour(cutout, levels=3)
```

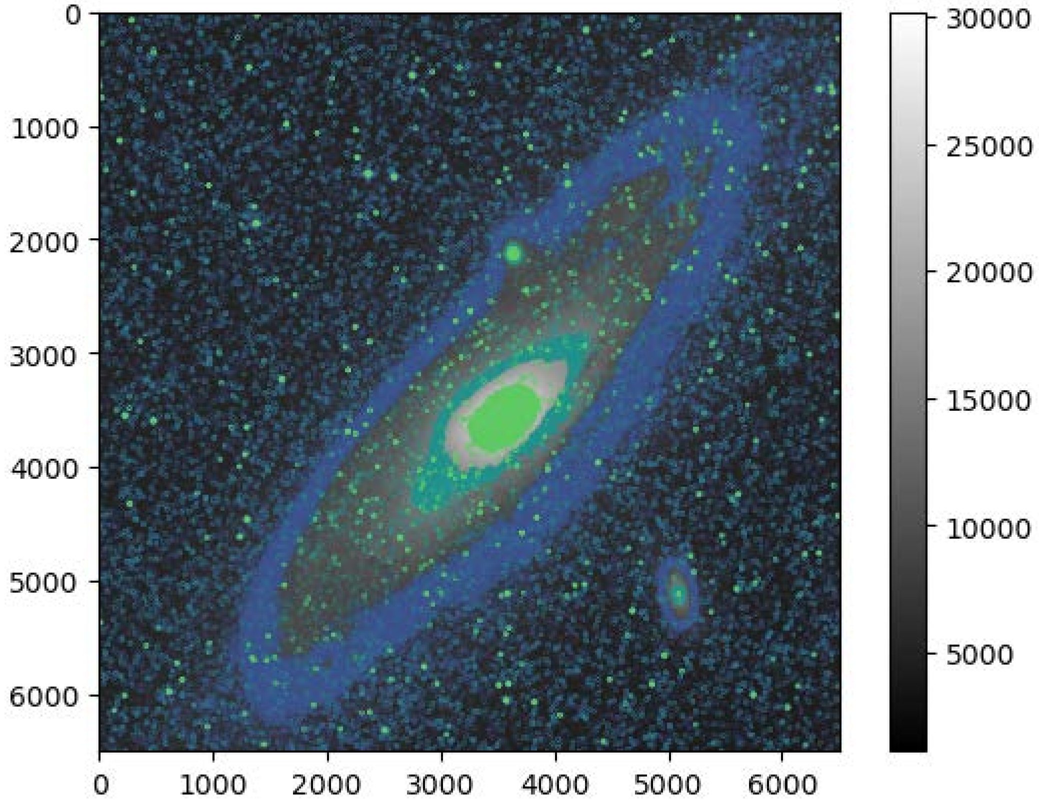


Рисунок 7.5. Выделение балджа, ядра и диска галактики Андромеды на фотографии

В данном случае выделяется уже 3 границы областей галактик: ядро, балдж и края диска. Для более детального изучения можно воспользоваться заливкой по граница с помощью похожей функции contourf:

```
plt.imshow(cutout, cmap='gray')
plt.colorbar()
plt.contourf(cutout, levels=3)
```

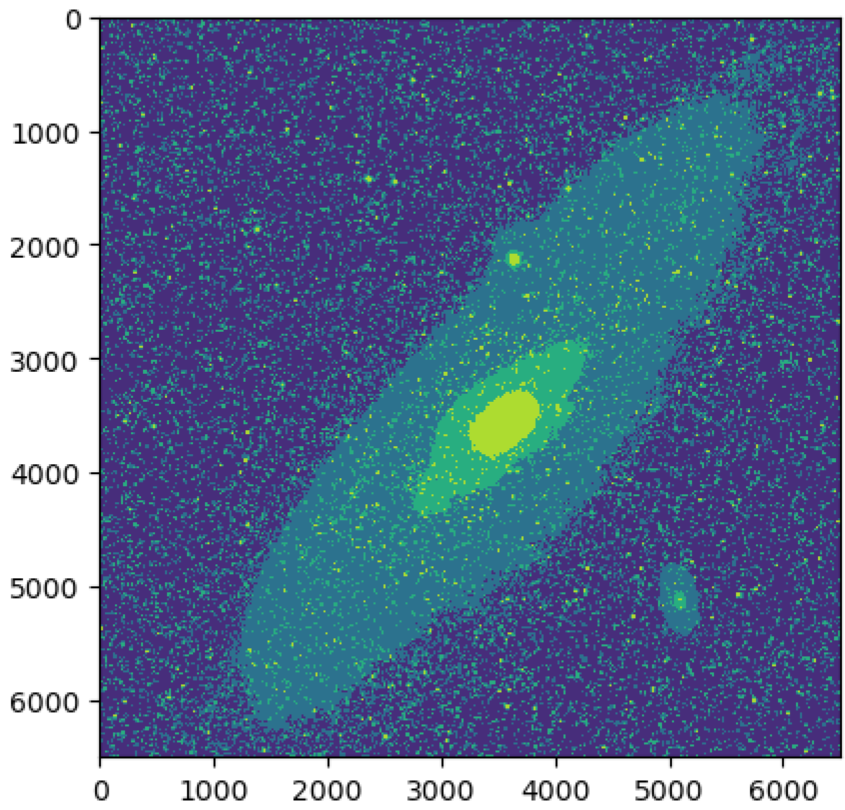


Рисунок 7.6. Построение структуры галактики M31

Здесь уже выделены четко границы ядра, балджа и диска в доступном для изучения формате. Отметим также, что Python самостоятельно смог выделить основные области галактики и нам не требовалось вводить дополнительные параметры.

Домашнее задание

Определите по структурированной картинке галактики Андромеды угол наклона галактики к плоскости луча зрения.

Примечание: подобные задачи встречаются на различных олимпиадах по астрономии. Как пример можно разобрать практическую задачу СПбАО 2023 года за 11 класс[6]*, 3 задачу СПбАО 2019 года за 8-9 класс*[7] *или 5 задачу МАО 2023 года за 10 класс*[8]*.*

[6] http://school.astro.spbu.ru/?q=system/files/11%20%D0%BA%D0%BB%D0%B0%D1%81%D1%81%20-%20%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D1%8F_52.pdf

[7] http://school.astro.spbu.ru/?q=system/files/8-9%20%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D1%8B%20-%20%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D1%8F_4.pdf

[8] https://mosastro.olimpiada.ru/upload/files/mos2023/mos_astro_2023_teor_10_ans.pdf

Анализ изображения M100

Эту же методику можно применить и для поиска областей активного звездообразования в спиральных галактиках. Построим fits-изображение M100:

```
M100 = fits.open('Downloads/m100_optical_B.fits')
M100.info()
M100_data = M100[0].data

plt.figure()
plt.imshow(M100_data, cmap='bone')
plt.colorbar()
```

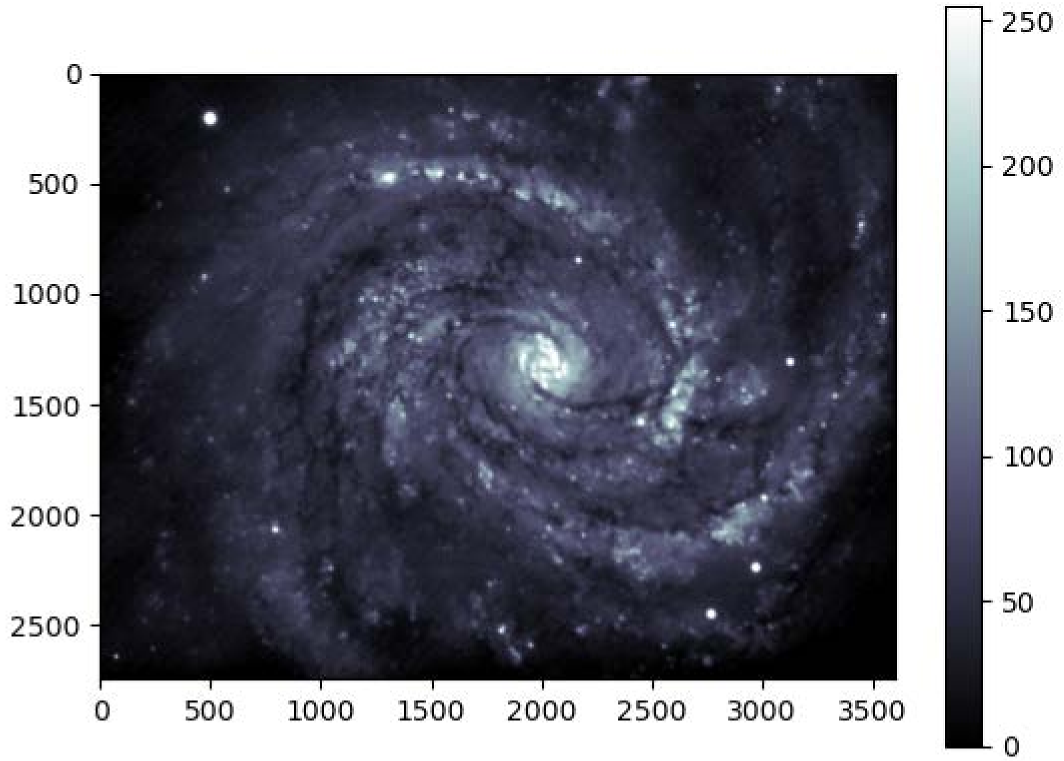


Рисунок 7.6. Построение изображения M100 в Matplotlib

Обратим внимание, что на рисунке помимо галактики отображены еще несколько отдельных звезд. Теперь вызовем функцию contour для выделения самых ярких областей в спиральной галактике:

```
plt.imshow(M100_data, cmap='bone')
plt.contour(M100_data, levels=1)
```

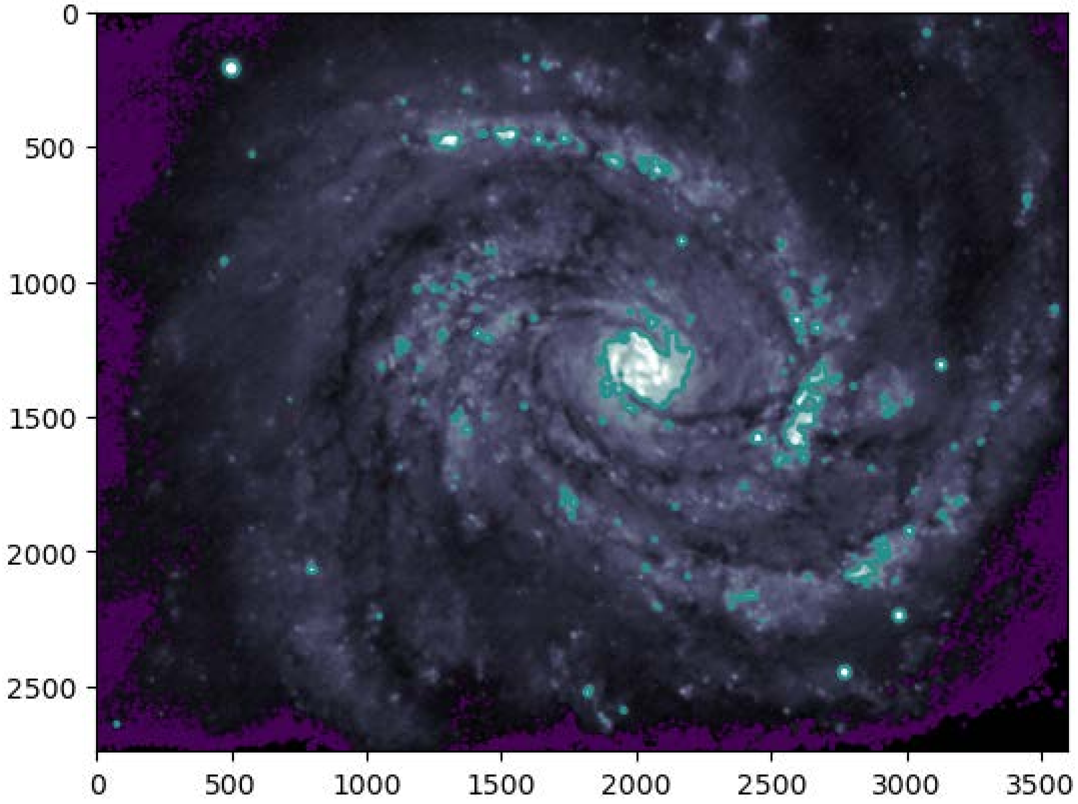


Рисунок 7.7. Выделение областей повышенной яркости в M100

Не сложно заметить, что основные очаги яркости соответствуют положению центра и рукавов галактики, что соответствует теории о том, что именно в них происходит наиболее активные звездообразовательные процессы.

Для более детальной картины можем также воспользоваться функцией contourf:

```
plt.imshow(M100_data, cmap='bone')
plt.contourf(M100_data, levels=7)
```

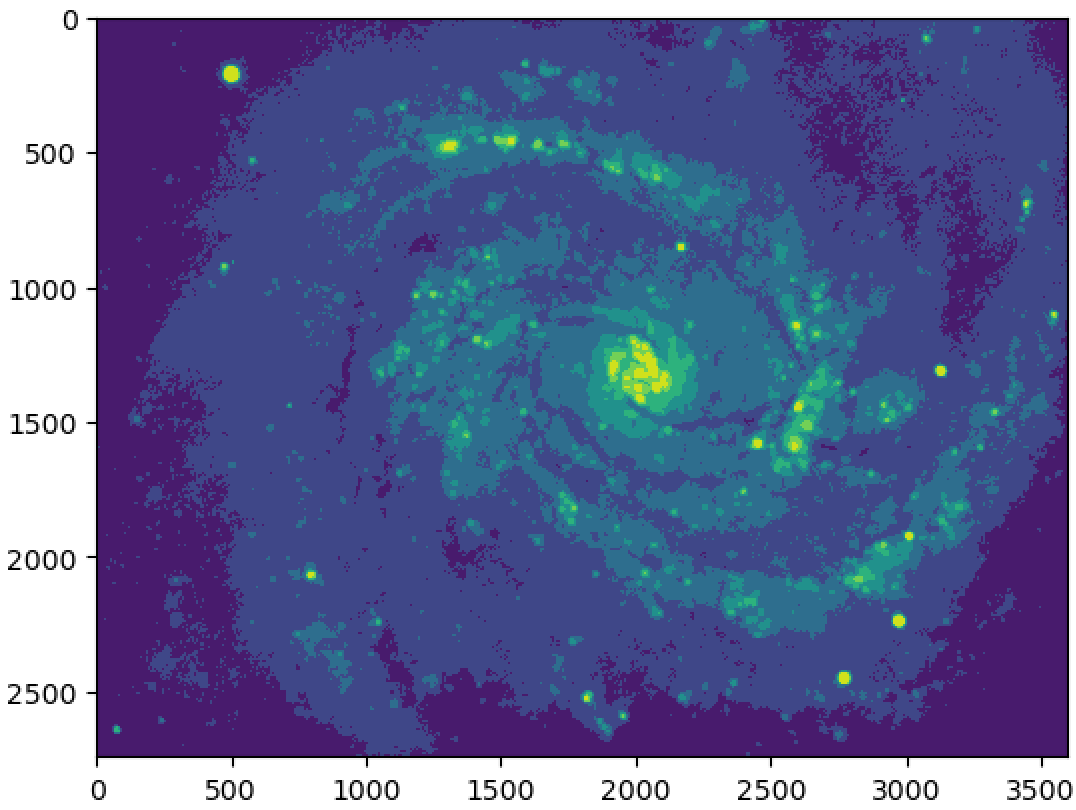


Рисунок 7.8. Построение структуры M100

На данной проекции чётко наблюдаются спиральные рукава и очаги звездообразования, а также общий диск галактики.

Анализ изображения Солнца

Интересной задачей является анализ фотометрических свойств Солнца по его изображению. Известно, что чем ближе к краю происходит наблюдение, тем меньше яркости на единицу телесного угла приходит оттуда. Это можно пронаблюдать, открыв изображение “sun.fits”.

```
sun = fits.open('Downloads/sun.fits')
sun.info()
sun_data = sun[0].data

plt.figure()
plt.imshow(sun_data, cmap='hot')
plt.colorbar()
```

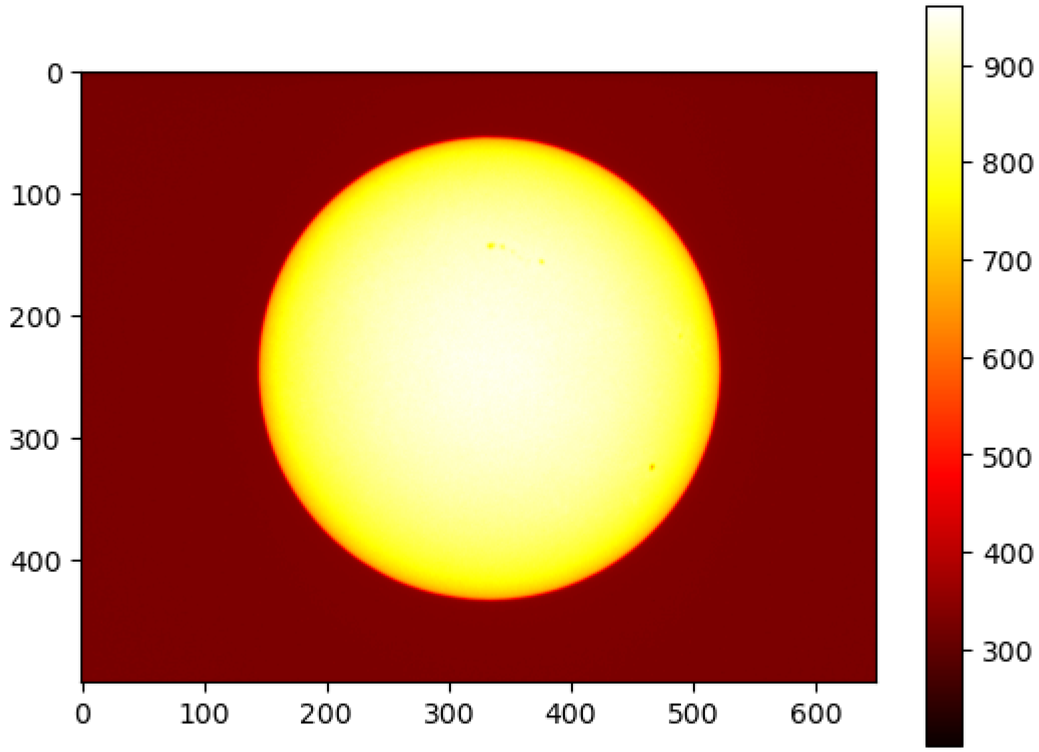


Рисунок 7.8. Построение изображения Солнца в Matplotlib

С помощью функции contour можно выделить наиболее яркие области:

```
plt.imshow(sun_data, cmap='hot')
plt.contour(sun_data, levels=1)
```

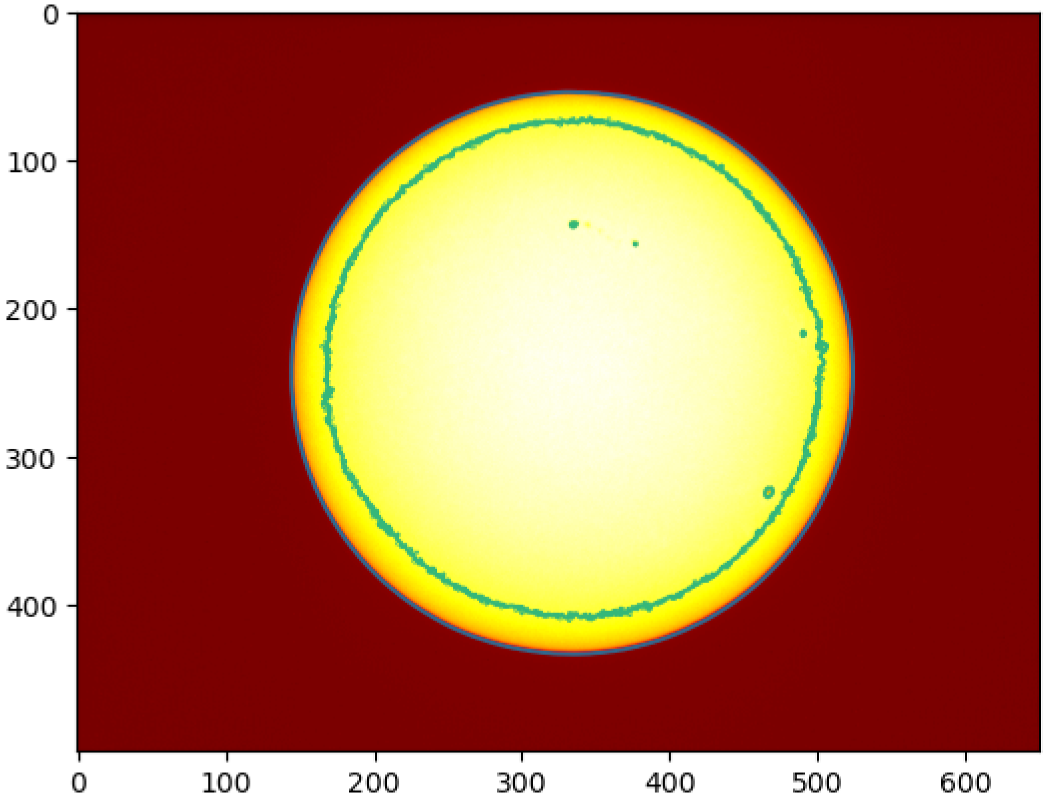


Рисунок 7.10. Выделение группы пятен на Солнце

```
plt.imshow(sun_data, cmap='hot')
plt.contour(sun_data, levels=2)
```

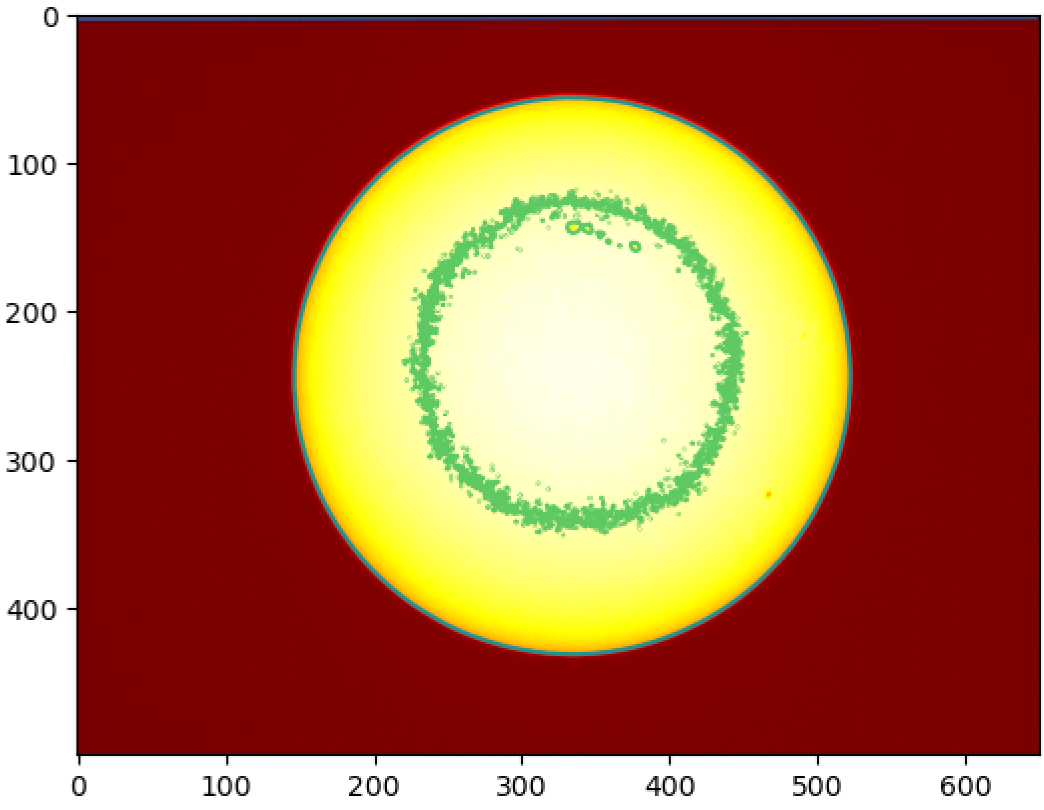


Рисунок 7.11. Выделение наиболее яркой части Солнца

Здесь выделяется наиболее яркая область Солнца, а также все группы солнечных пятен. По данной структуризации можно теперь легко определить степень активности Солнца, посчитав число Вольфа:

$$W = (f + 10g)k$$

Где f – количество наблюдаемых пятен, g – количество групп пятен, а k – поправочный коэффициент, который применяется Международным координационным центром.

В данном случае g = 3, а f = 8. Поправочный коэффициент можем принять равным 1, значит число Вольфра для данной фотографии Солнца составляет 38. Пример подобной задачи можно встретить например в задании для 5-8 класса СПбАО за 2011 год[9].

При увеличении параметра ‘levels’ можно более детально пронаблюдать эффект затемнения края диска Солнца:

```
plt.imshow(sun_data, cmap='hot')
plt.contourf(sun_data, levels=25)
```

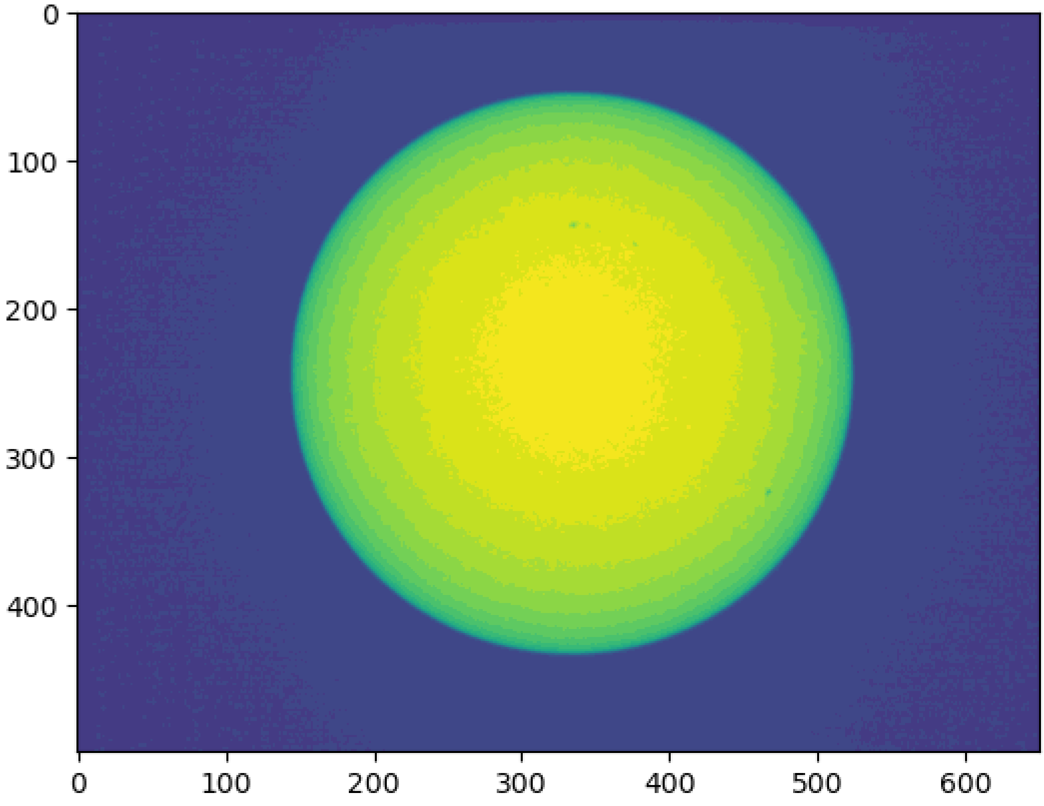


Рисунок 7.12. Построение структуры Солнца в зависимости от расстояния от центра

[9] http://school.astro.spbu.ru/?q=system/files/5678_2.pdf

Данный эффект связан с оптическим поглощением света при прохождении через Солнечную корону и с понижением эффективной температуры, которая по закону Стефана-Больцмана приводит к снижению яркости. Лучи, которые выходят в области близкой к центру, проходят меньший оптический путь, чем лучи, которые выходят с края диска. В качестве примера разбора данного явления можно изучить задачу с практического тура 10 класса СПбАО за 2015 год[10].

Глава 8. Анализ спектра звезды

Формула Планка

В современной астрономии большую часть информации об объектах мы получаем с помощью спектров. Для начала рассмотрим физическую модель абсолютно черного тела, то есть объекта, который поглощает и излучает на всех длинах волн. Данная модель излучения описывается с помощью формулы Планка:

$$B_\nu(\nu, T) = \frac{2h\nu^3}{c^2} \cdot \frac{1}{e^{\frac{h\nu}{kT}} - 1}$$

Где B_ν – это плотность потока энергии в единице диапазона частот, которая зависит от частоты света(ν) и от температуры(T). h – постоянная планка, c - скорость света, k - постоянная Больцмана. Также можно выразить данную величину в единицах длин волн:

10

http://school.astro.spbu.ru/?q=system/files/10%20%D0%BA%D0%BB%D0%B0%D1%81%D1%81%20-%20%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D1%8F_11.pdf

$$B_\lambda(\lambda, T) = \frac{2hc^2}{\lambda^5} \cdot \frac{1}{e^{\frac{hc}{\lambda kT}} - 1}$$

Построим данную зависимость для различных значений температур:

```
T1 = 6000
T2 = 7000
T3 = 5000
h = 6.63*10**(-34)
c = 3*10**8
k = 1.38*10**(-23)
x = np.arange(0, 2000.0, 1.0)
B1 = 2*h*c**2/((np.exp(h*c/(k*0.000000001*x*T1))-1)*(0.000000001*x)**5)
B2 = 2*h*c**2/((np.exp(h*c/(k*0.000000001*x*T2))-1)*(0.000000001*x)**5)
B3 = 2*h*c**2/((np.exp(h*c/(k*0.000000001*x*T3))-1)*(0.000000001*x)**5)
plt.plot(x, B1, 'g', label='6000K')
plt.plot(x, B2, 'r', label='7000K')
plt.plot(x, B3, 'b', label='5000K')
plt.xlabel('Длина волны, нм')
plt.ylabel('Плотность энергии, Вт/м^3')
plt.legend()
```

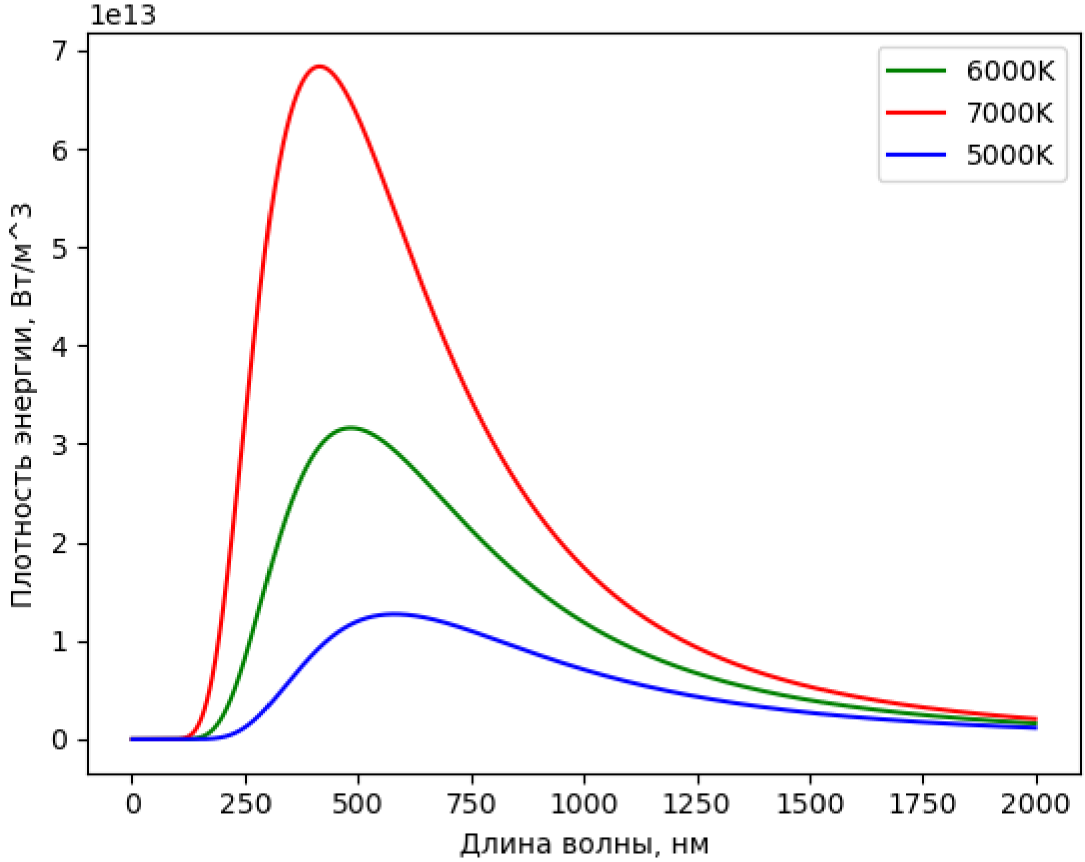


Рисунок 8.1. Зависимость плотности энергии от длины волны для разных температур

Не сложно заметить, что при увеличении температуры положение максимума смещается влево, а сам график становится выше. Попробуем изучить данные закономерности с помощью программных методов. Вначале создадим массив из n зависимостей для разных температур, а дальше определим значение длины волны максимума для каждой из зависимостей:

```
n = 1000
x = np.arange(0.0, 0.00001, 0.000000001)
T_max = 100000.0
T_min = 500.0
T = np.arange(T_min, T_max, (T_max-T_min)/n)


lambda_max = [0]*n
for i in range(0, n):
    B = 2*h*c**2/((np.exp(h*c/(k*x*T[i]))-1)*x**5)
    max_B = np.nanmax(B)
    j = np.where(B == max_B)
    lambda_max[i] = x[j]

plt.plot(T, lambda_max)
plt.xlabel('Температура, K')
plt.ylabel('Длина волны максимума, м')
```

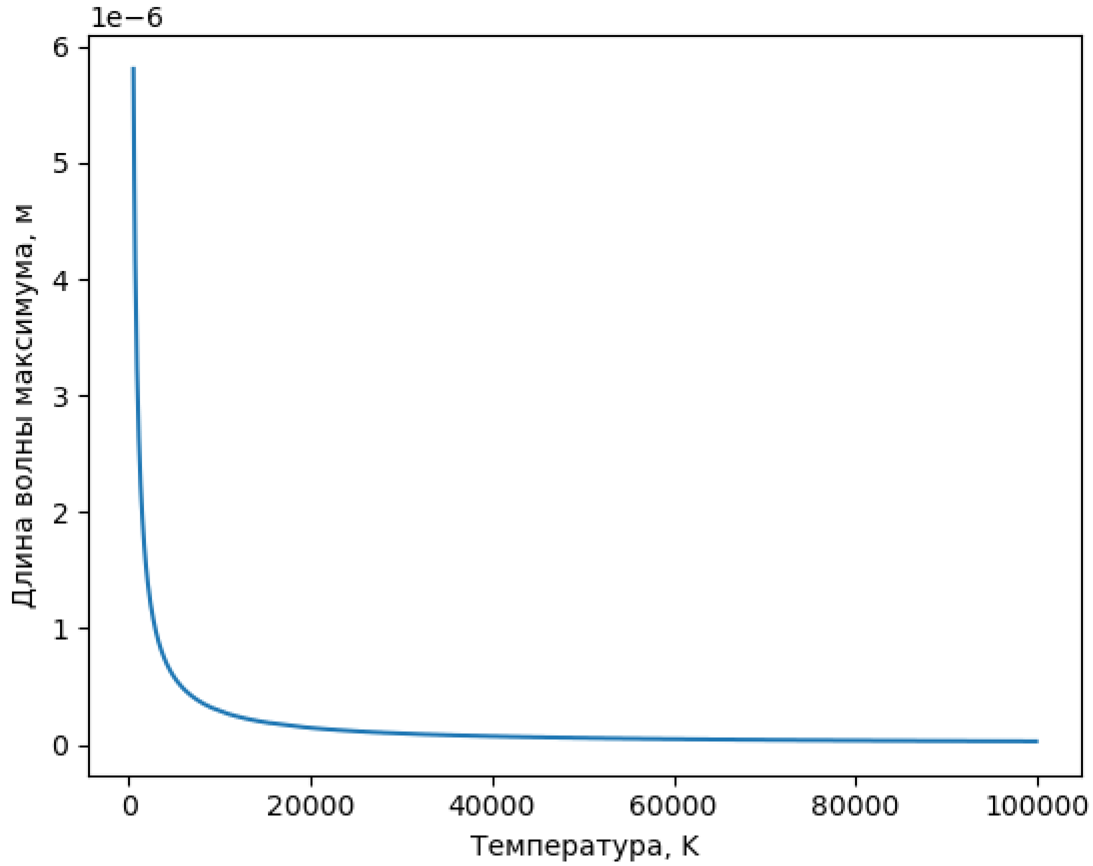


Рисунок 8.2. Построение зависимости длины волны максимума излучения от температуры

Зависимость длины волны максимума от температуры получилось обратная, чтобы убедиться, что это гипербола, сделаем логарифмирование осей:

```
plt.plot(np.log(T), np.log(lambda_max))
plt.xlabel('log(T)')
plt.ylabel('log(lambda)')
```

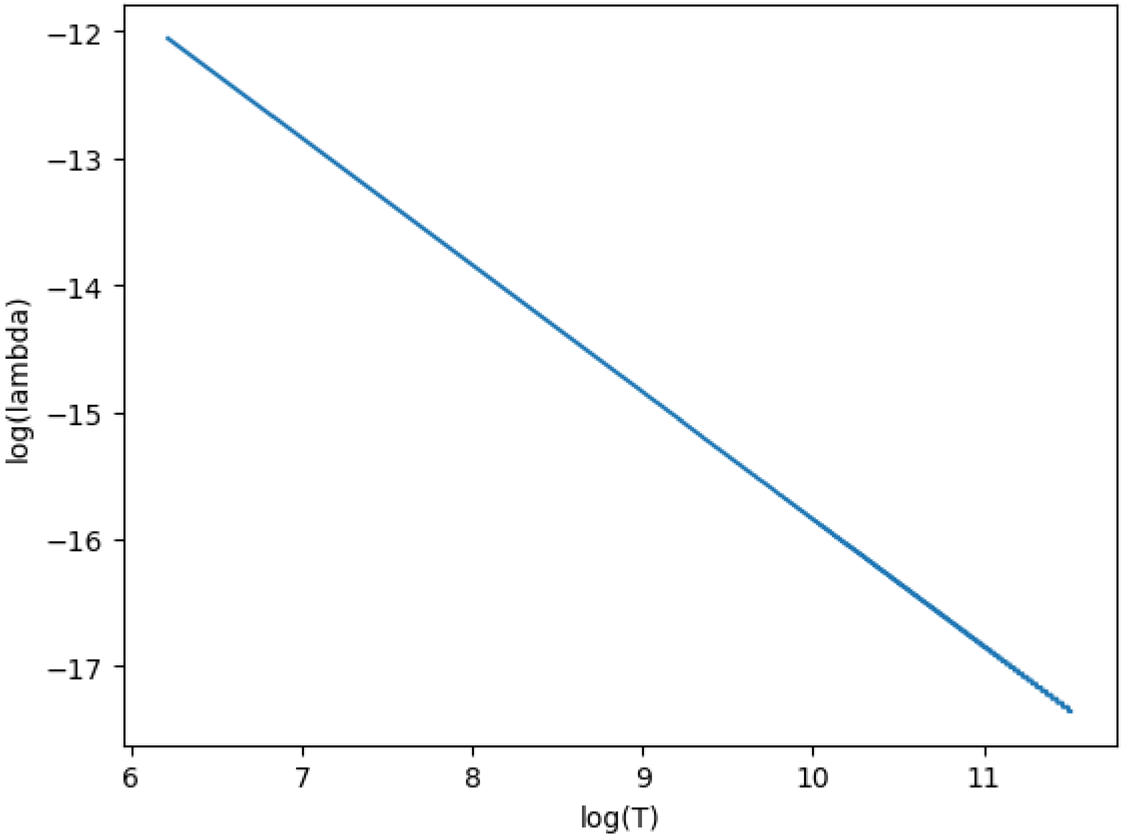


Рисунок 8.3. Построение зависимости длины волны максимума излучения от температуры в логарифмическом масштабе

Получаем, что коэффициент угла наклона графика составляет –1, значит исходная зависимость действительно является гиперболой. Также можно определить параметр данной зависимости:

```
print(MNK([T**(-1), lambda_max]))
```

Получим, что коэффициент пропорциональности составляет 0,0029 К·м. Данная зависимость называется законом Вина:

$$\lambda \cdot T = b = 0{,}0029\ \text{К} \cdot \text{м}$$

Поскольку B_λ – это плотность энергии на единицу длины волны, то при суммировании по всем длинам волн мы получим общую плотность энергии от абсолютно черного тела. В данном случае площадь под графиком можно определить с помощью метода Монте-Карло для различных температур:

```
n = 1000
x_min = 0.0
x_max = 0.00001
x = np.arange(0.0, 0.00001, (x_max-x_min)/n)
T_max = 1000.0
T_min = 500.0
T = np.arange(T_min, T_max, (T_max-T_min)/n)


L = [0]*n
for i in range(0, n):
    B = 2*h*c**2/((np.exp(h*c/(k*x*T[i]))-1)*x**5)
    t = np.random.uniform(0.0, np.nanmax(B), n)
    q = 0
    for j in range(0, n):
        if t[j]<B[j]:
            q += 1
    L[i] = (q/n)*(x_max*np.nanmax(B))

plt.plot(T, L)
plt.xlabel('Температура, K')
plt.ylabel('Светимость с единицы площади, Вт/м^2')
```

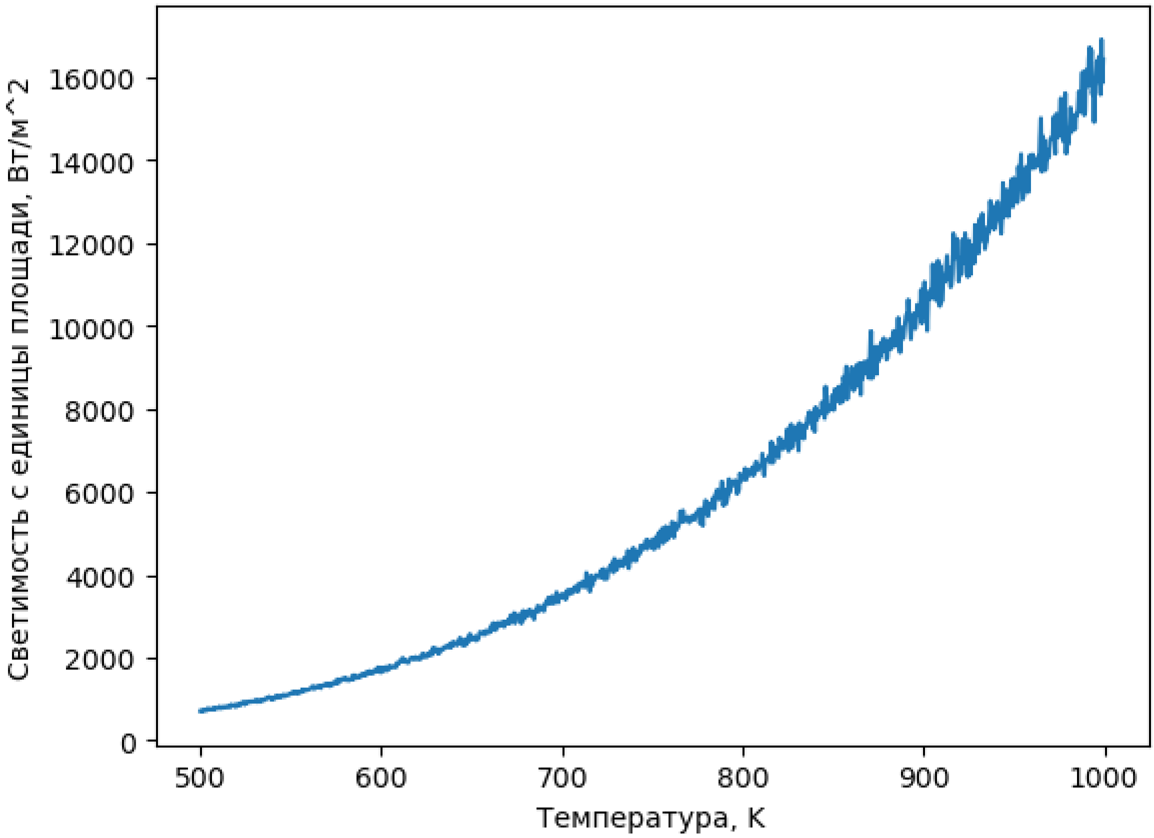


Рисунок 8.4. Построение зависимости светимости с единицы площади от температуры

И изучим данную зависимость в логарифмическом масштабе:

```
plt.plot(np.log(T), np.log(L))
```

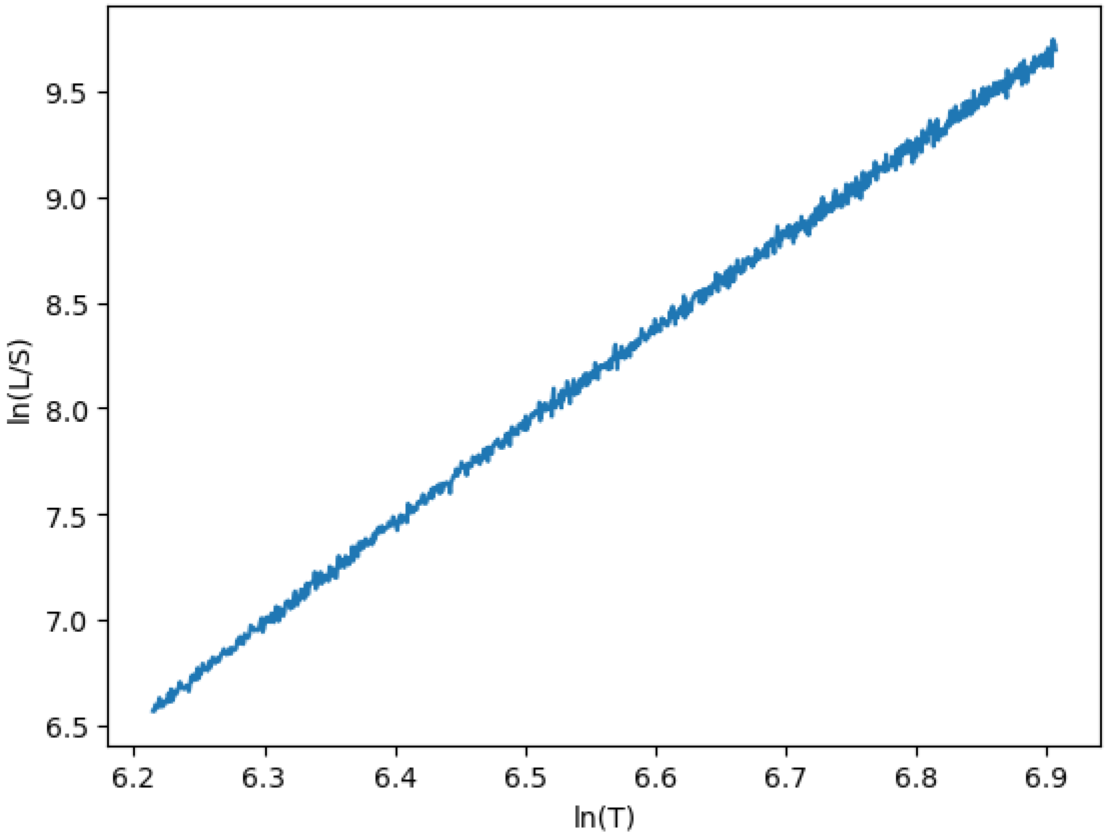


Рисунок 8.5. Построение зависимости светимости с единицы площади от температуры в логарифмическом масштабе

Получаем, что коэффициент угла наклона графика составляет 4, значит исходная зависимость будет подчиняться следующему выражению:

$$\frac{L}{S} = \sigma T^4$$

Таким образом, при помощи моделирования, мы получили закон Стефана-Больцмана – закон излучения абсолютно черного тела. Если умножить данную величину на площадь поверхности звезды, то получится светимость данной звезды.

Изучение реальных спектров звезд

Для настоящих спектров будут наблюдаться неточности, в сравнении с математической моделей. Для моделирования определенных спектров звезд в зависимости от набора задаваемых параметров можно воспользоваться библиотекой PyAstronomy:

```
from PyAstronomy import pyasl
import os
```

```
sl = pyasl.SpectralLib()

# List available libraries
sl.listDescription()

# Disk-integrated spectrum for Teff, logg, and metallicty
sdifn = sl.requestModel(5000, 4.5, 0.0, nex="diskint")

print("Model filename: ", os.path.basename(sdifn))

w, f = sl.read1dFitsSpec(sdifn)

plt.xlabel("Wavelength [$\AA$]")
plt.ylabel("Astrophysical flux [erg/cm**2/s/A]")
plt.plot(w, f)
plt.show()
```

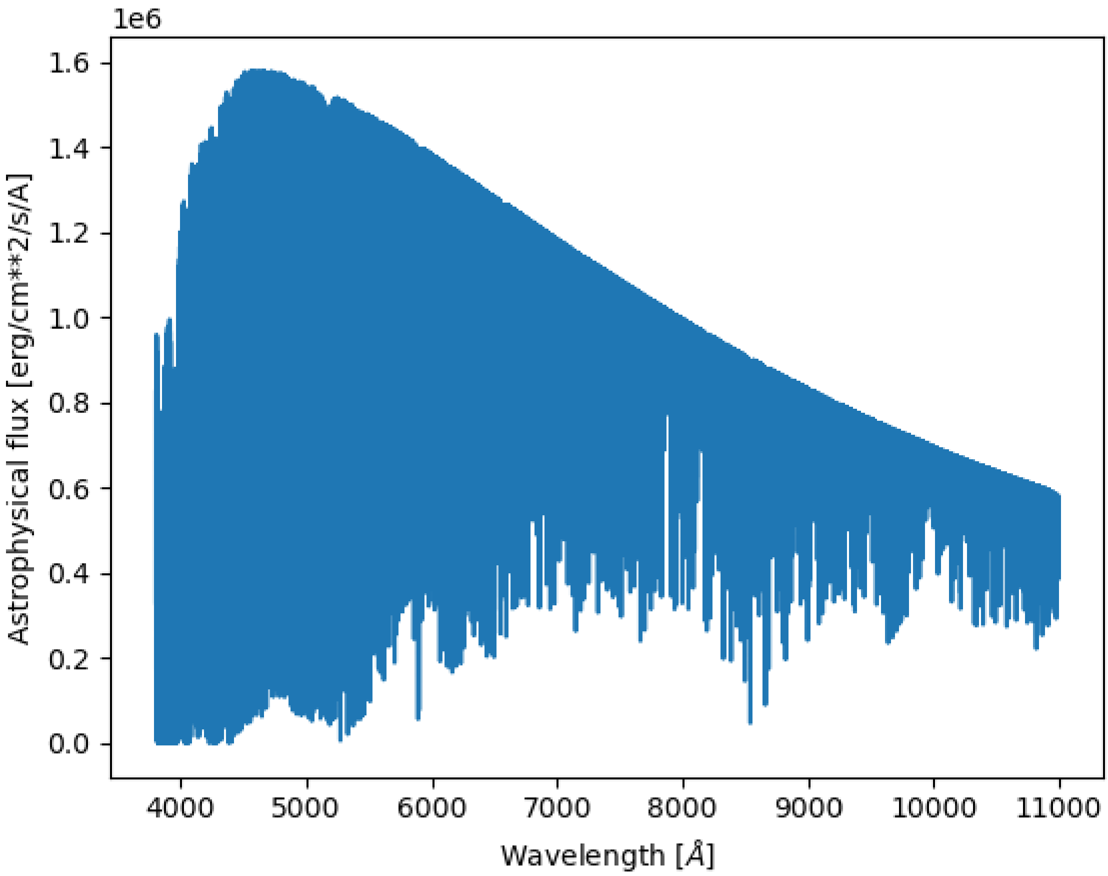


Рисунок 8.6. Смоделированный спектр с заданными параметрами

Для выделения точечной области:

```
wlmin = 6564-20
wlmax = 6564+20
indi = (w > wlmin) & (w < wlmax)
```

```
# Specific intensity resolved w.r.t. viewing angle
# for Teff, logg, and metallicty
musfn = sl.requestModel(3500, 4.5, 0.0, nex="muspecs")

w, mus, intens = sl.readMuFits(musfn)

print("Available cos(theta) = mu angles: ", mus)

# Plot specific intensity
for i, mu in enumerate(mus):
    plt.plot(w[indi], intens[indi,i], label="mu = %3.1f" % mu)
plt.xlabel("Wavelength [$\AA$]")
plt.ylabel("Specific intensity")
plt.legend()
plt.show()
```

Также можно сравнить этот спектр с зависимостью Планка:

```
# Real (not astrophysical) flux
rf = np.zeros_like(w)

for i in range(len(mus)-1):
    # rf += I*mu*dmu
    rf += (intens[::,i]*mus[i] + intens[::,i+1]*mus[i+1])/2 * (mus[i+1] - mus[i])
rf *= 2*np.pi
# Astrophysical flux
af = rf/np.pi

plt.plot(w[::10], f[::10], 'g:', label="Disk-integrated library flux")
plt.plot(w[::10], af[::10], 'b.:', label="Astrophysical flux (integral)")
plt.plot(w[::10], pyasl.planck(5750, w[::10]*1e-10)*1e-7/np.pi, 'r--', label="Planck")
plt.legend()
plt.show()
```

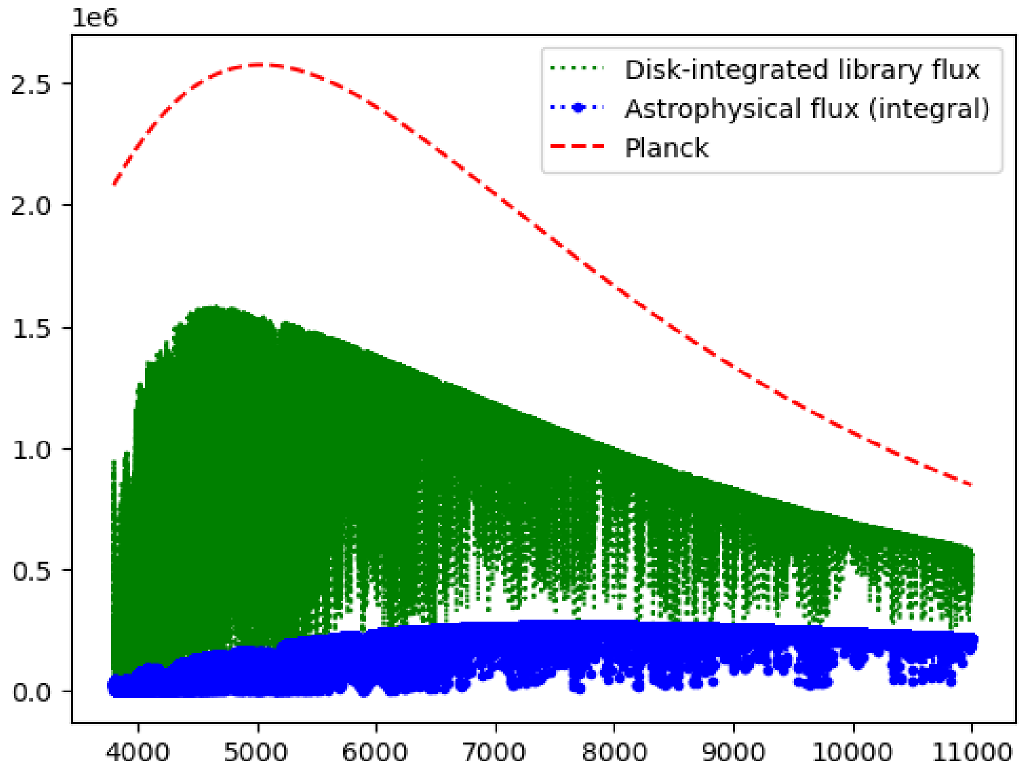


Рисунок 8.7. Построение модели реального спектра и его сравнение с приближением Планка

Для изучения реального спектра звезды можно воспользоваться каталогом SDSS[11], в котором хранится информация о миллионах звезд. Также можно скачать напрямую интересующий вас спектр[12] или воспользоваться дополнительными инструментами для построения отдельных спектров[13].

вот реальный спектр звезды
https://skyserver.sdss.org/dr17/VisualTools/navi
а отсюда можно загрузить оцифрованный сректр
http://specdash.idies.jhu.edu/?catalog=sdss&specid=3256188368865748992
Data--> Download-->Save data as json dump
Сможете применить Вашу программу для подгонки непрерывного спектра реальной звезды из каталога SDSS кривой Планка и оценкой эффективной температуры по Вину или С-Б?

[11] https://www.sdss.org/
[12] http://specdash.idies.jhu.edu/?catalog=sdss&specid=3256188368865748992
[13] https://skyserver.sdss.org/dr17/VisualTools/navi

Глава 9. Работа с каталогами

Каталог Мессье

Следующей немаловажной частью является отображение объектов на карте. Но для отображении полусферы на плоскость происходит множество искривлений (вспомните карту Земли на уроках географии). Для проецирования сферы на плоскость принято использовать проекцию Мольвейде. Эта карта представляет собой эллипс отношение полуосей которого составляет 2:1.

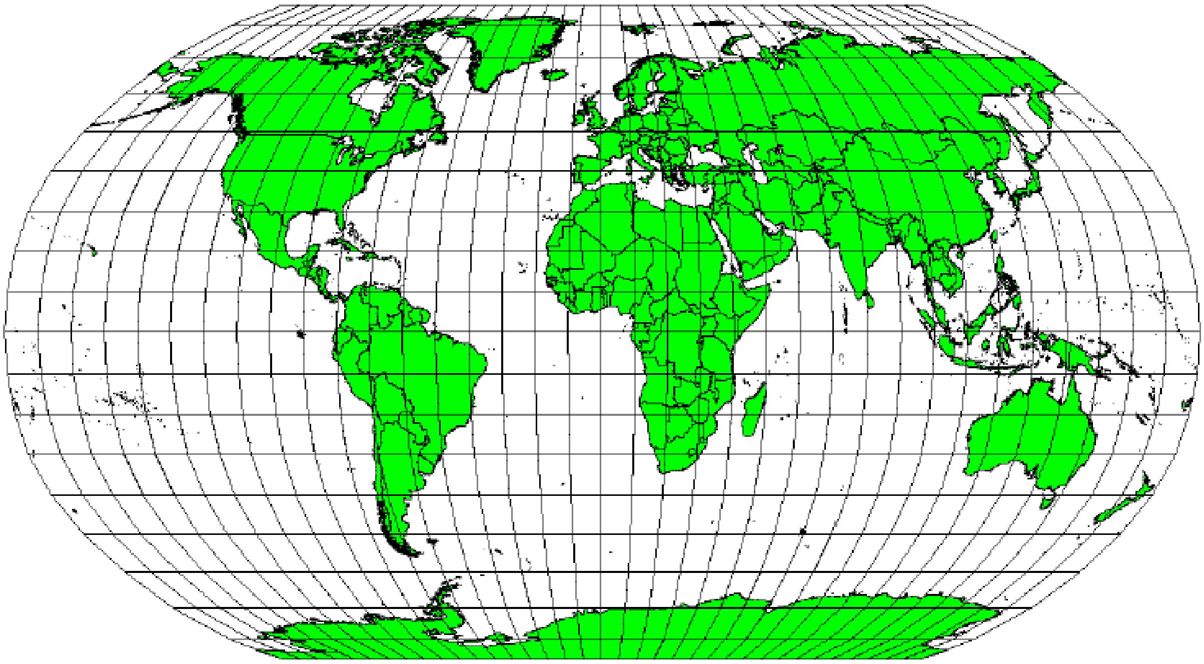


Рисунок 9.1. Проекция Мольвейде на примере карты Земли

В рамках подготовки к данному упражнению был подготовлен отдельный файл формата “.csv” с объектами каталога Мессье, который можно найти в приложении. Для упрощения задачи были произведены первичные итерации по преобразованию склонений и прямых восхождений объектов в десятичную меру градусного угла из классических форматов (**h**m у прямого восхождения и xx°xx’ у склонения). Также прямое восхождение было переведено в диапазон от -180° до 180°.

Таблица 5. Вид таблицы файла “Messier.csv”

Номер объекта	Название объекта	Тип объекта	Прямое восхождение	Склонение	Расстояние, пк
M 1	NGC 1952	nebula	83.625	22.0166	2000
...	…	…	…	…	…

Отобразим объекты Мессье из заготовленного файла в проекции Мольвейде.

```
messier     #исходный массив messier .csv

ra = [float(i) for i in messier [:, 3]]     #столбец прямого восхождения
dec = [float(i) for i in messier [:, 4]]     #столбец склонения
d = [float(i) for i in messier [:, 5]]     #столбец расстояния в пк

fig = plt.figure(figsize=(8,6))
ax = fig.add_subplot(111, projection="mollweide")
ax.grid(True)
ax.set_xticklabels(['14h','16h','18h','20h','22h','0h','2h','4h','6h','8h','10h'])
ax.scatter(np.radians(ra), np.radians(dec))
```

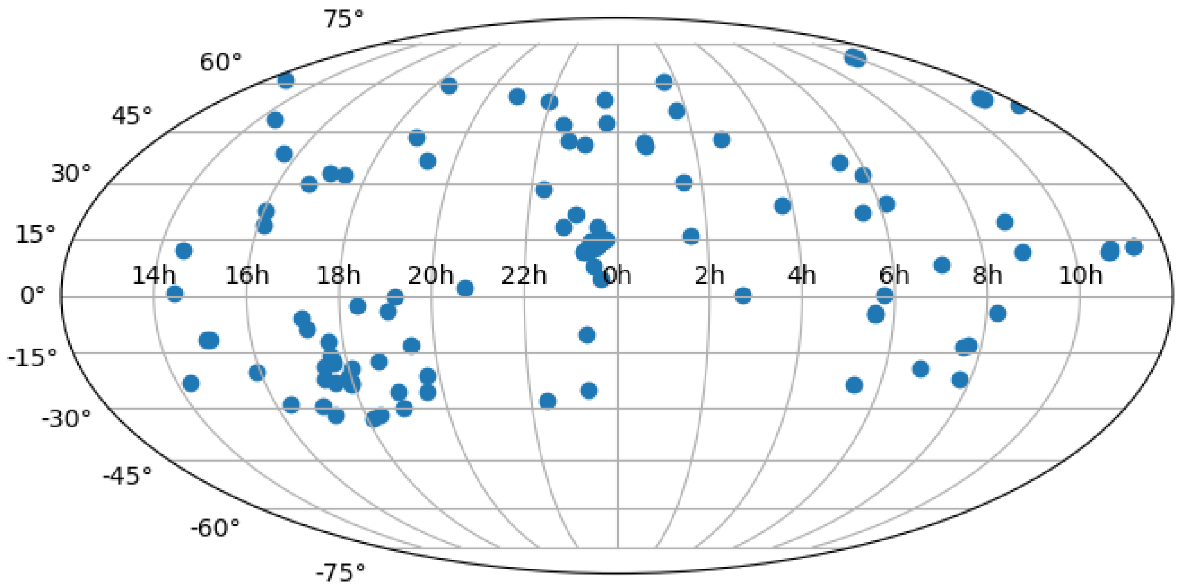


Рисунок 8.2. Отображение объектов Мессье на проекции Мольвейде

Теперь необходимо отобразить объекты с учетом их типа:

```
fig = plt.figure(figsize=(8,6))
ax = fig.add_subplot(111, projection="mollweide")
ax.grid(True)
```

```
ax.set_xticklabels(['14h','16h','18h','20h','22h','0h','2h','4h','6h','8h','10h'])

type = messier[:, 2]

for (i, _) in enumerate(type):
    if type[i] == 'Galaxy':
        ax.scatter(np.radians(ra[i]), np.radians(dec[i]), c='#FF0000') #красный
    elif type[i] == 'open cluster':
        ax.scatter(np.radians(ra[i]), np.radians(dec[i]), c='#FF00FF') #розовый
    elif type[i] == 'globular cluster':
        ax.scatter(np.radians(ra[i]), np.radians(dec[i]), c='#7CFC00') #зеленый
    elif type[i] == 'nebula':
        ax.scatter(np.radians(ra[i]), np.radians(dec[i]), c='#1E90FF') #синий
    else:
        ax.scatter(np.radians(ra[i]), np.radians(dec[i]), c='#000000') #черный
```

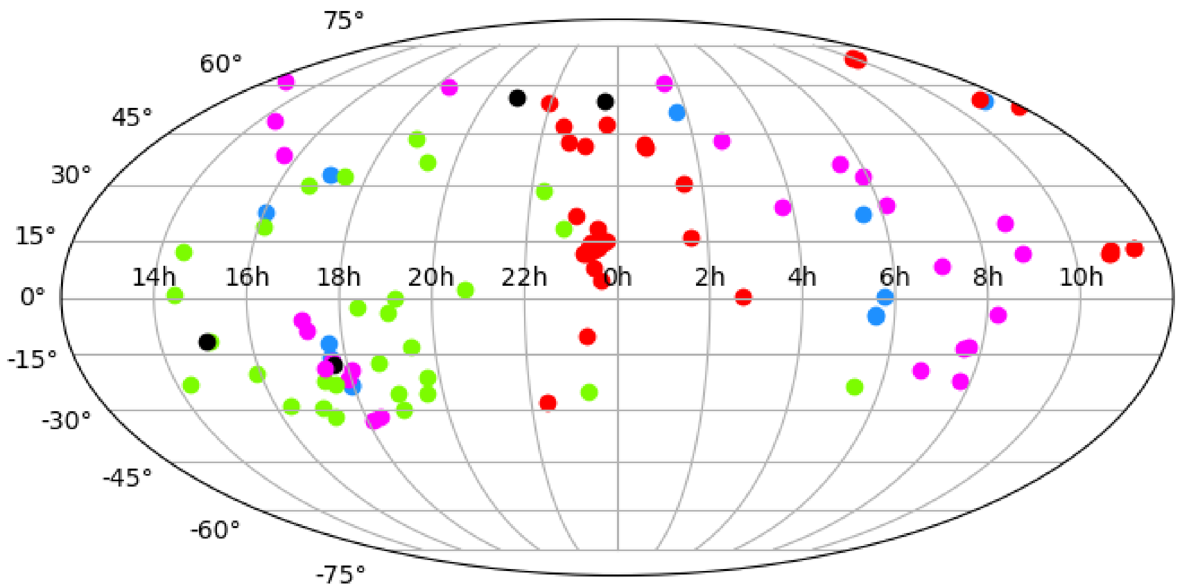


Рисунок 9.3. Типизация объектов Мессье на карте. Галактики - красный цвет, рассеянные звездные скопления - розовый цвет, шаровые звездные скопления - зеленый цвет, туманности - синий цвет, иное - черный цвет

Из этой картинки можно зафиксировать расположение галактического центра на данной проекции. Также видно, что рассеянные скопления располагаются в диске галактики. В отличии от привычной карты звездного неба в проекции Мольвейде галактический диск будет идти по смещенному эллипсу. Характерное его положение как раз соответствует положению розовых точек на полученном графике.

Также обратим отдельное внимание на положение шаровых звездных скоплений. Шаровые звездные скопления (зеленые точки) располагаются достаточно кучно, так как они располагаются в гало нашей галактики. С помощью них попробуем оценить положение центра галактики. Для этого необходимо сначала перевести сферические координаты в декартовы:

$$x_i = r_i cos\ \delta_i\ cos\ \alpha_i$$

$$y_i = r_i cos\, \delta_i\, sin\, \alpha_i$$
$$z_i = r_i sin\, \delta$$

А далее для оценки положения центра системы шаровых скоплений Галактики необходимо просто рассчитать среднеарифметическое значение по координатам:

$$\underline{x} = \frac{\sum_{i=1}^{N} \quad x_i}{N}$$
$$\underline{y} = \frac{\sum_{i=1}^{N} \quad y_i}{N}$$
$$\underline{z} = \frac{\sum_{i=1}^{N} \quad z_i}{N}$$

И далее совершить обратный переход в экваториальные координаты:

$$r_{\text{центра}} = \sqrt{\underline{x}^2 + \underline{y}^2 + \underline{z}^2}$$
$$\delta_{\text{центра}} = arcsin(\frac{\underline{z}}{r_{\text{центра}}})$$
$$\alpha_{\text{центра}} = arctg(\frac{\underline{y}}{\underline{x}})$$

```
import math
z = []
x = []
y = []

for (i, _) in enumerate(type):
    if type[i] == 'globular cluster':
        z.append(d[i]*np.sin(np.radians(dec[i])))
        x.append(d[i]*np.cos(np.radians(dec[i]))*np.cos(np.radians(ra[i])))
        y.append(d[i]*np.cos(np.radians(dec[i]))*np.sin(np.radians(ra[i])))

z_mean = np.mean(z)
x_mean = np.mean(x)
y_mean = np.mean(y)

print('d_center=', np.sqrt(z_mean**2+x_mean**2+y_mean**2))
print('ra_center=', math.degrees(np.arctan(y_mean/x_mean)))
print('dec_center=', math.degrees(np.arctan(z_mean/(np.sqrt(x_mean**2+y_mean**2)))))
```

Получим следующие результаты:

$$r_{\text{центра}} = 6{,}4 \text{ кпк}$$
$$\delta_{\text{центра}} = \ -9,6°$$
$$\alpha_{\text{центра}} = 18h03m$$

В реальности центр ядра Галактики находится в 8,5 кпк от Солнца в созвездии Стрельца ($\alpha = 17h40m\,, \delta = -29°$). Из чего ученикам предлагается сделать вывод, что расхождение рассчитанных значений центра подсистемы шаровых скоплений в

Галактике с реальным положением центра Галактики очевидным образом связаны с неполнотой каталога Мессье, но дают неплохой оценочный результат.

Обсерватория Гершель

Космический телескоп “Гершель” во время своей работы располагался вблизи второй точки Лагранжа L2, что позволяло значительно экранировать его от солнечного излучения и обладал самым крупным зеркалом на тот момент (до запуска космического телескопа “Джеймс Уэбб”). Диаметр его зеркала составлял 3,5 метра и состоял из 12 элементов, в которых поддерживали температуру порядка 1,4 К.

Основные инструменты КТ “Гершель” были следующие:

- Основная фотокамера со спектрометром низкого разрешения (Photodetecting Array Camera and Spectrometer, PACS) работала в инфракрасном диапазоне от 60 до 210 мкм.

- Приемник фотометрических изображений (Spectral and Photometric Imaging Receiver, SPIRE) работал на диапазоне 194-672 мкм.

- А гетеродинный датчик (Heterodyne Instrument for the Far Infrared, HIFI) работал в диапазонах 157-212 мкм и 240-625 мкм.

Благодаря данным спектрометра Гершель мог фиксировать галактики с красным смещением больше 2-3 и накопил обширную базу данных по объектам глубокого космоса.

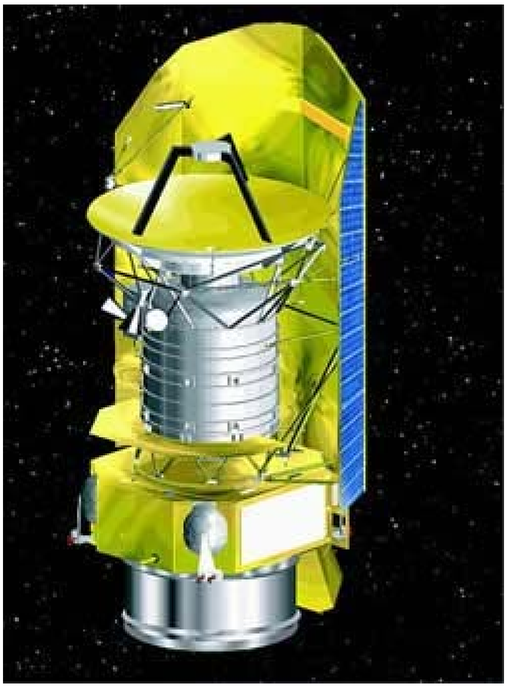


Рисунок 9.4. Фотография телескопа “Гершель”

Теперь давайте попробуем поработать с настоящим каталогом астрономических объектов с космической обсерватории “Гершель”. Проект VOX (The Virtual Observatory at susseX) предоставляет возможность получать данные о точечных объектах излучения в заданном участке небесной сферы. Стоит отметить, что проект “Гершель” смог собрать информацию не по всей небесной сфере, а только в ограниченном участке. В дальнейшем обучающиеся могут самостоятельно изучить в каких диапазонах склонения и прямого восхождения получены данные с помощью этой обсерватории.

В ходе работы потребуется отобразить все данные, расположенные в радиусе 0,01° от заданных координат на небе. Данный радиус был выбран из соображений ограниченности времени на выполнение задания. Больший радиус приведет к квадратичному росту данных и увеличению времени обработки.

Задание:

Отобразите все объекты находящиеся на расстоянии не более 0,01° от точки с координатами: α=161,63° и δ=59,05°. Также определите число данных объектов в указанном диапазоне.

```
from astropy.io import ascii, fits
from astropy import units as u
from astropy.coordinates import SkyCoord
import numpy as np
from astropy.table import Table
import pylab as plt
import seaborn as sns
%matplotlib inline
from astropy import wcs
import pyvo as vo
import matplotlib.pyplot as plt
RA=160.0
 Dec=60.0
radius=0.01
service = vo.dal.TAPService("https://herschel-
vos.phys.sussex.ac.uk/__system__/tap/run/tap")
resultset = service.search("SELECT TOP 10000 * FROM herschelhelp.main WHERE
1=CONTAINS(POINT('ICRS', ra, dec),CIRCLE('ICRS',"+str(RA)+",
"+str(Dec)+","+str(radius)+" ))")
masterlist=resultset.to_table()
plt.plot(masterlist['ra'],masterlist['dec'], 'o',label='search results')
plt.plot(RA,Dec,'ro',label='search position')
plt.xlabel('Right Ascension')
plt.ylabel('Declination')
plt.legend()
masterlist

plt.scatter(masterlist['ra'], masterlist['dec'], c=masterlist['redshift'], cmap='Oranges', s=1,
alpha = 0.8)
```

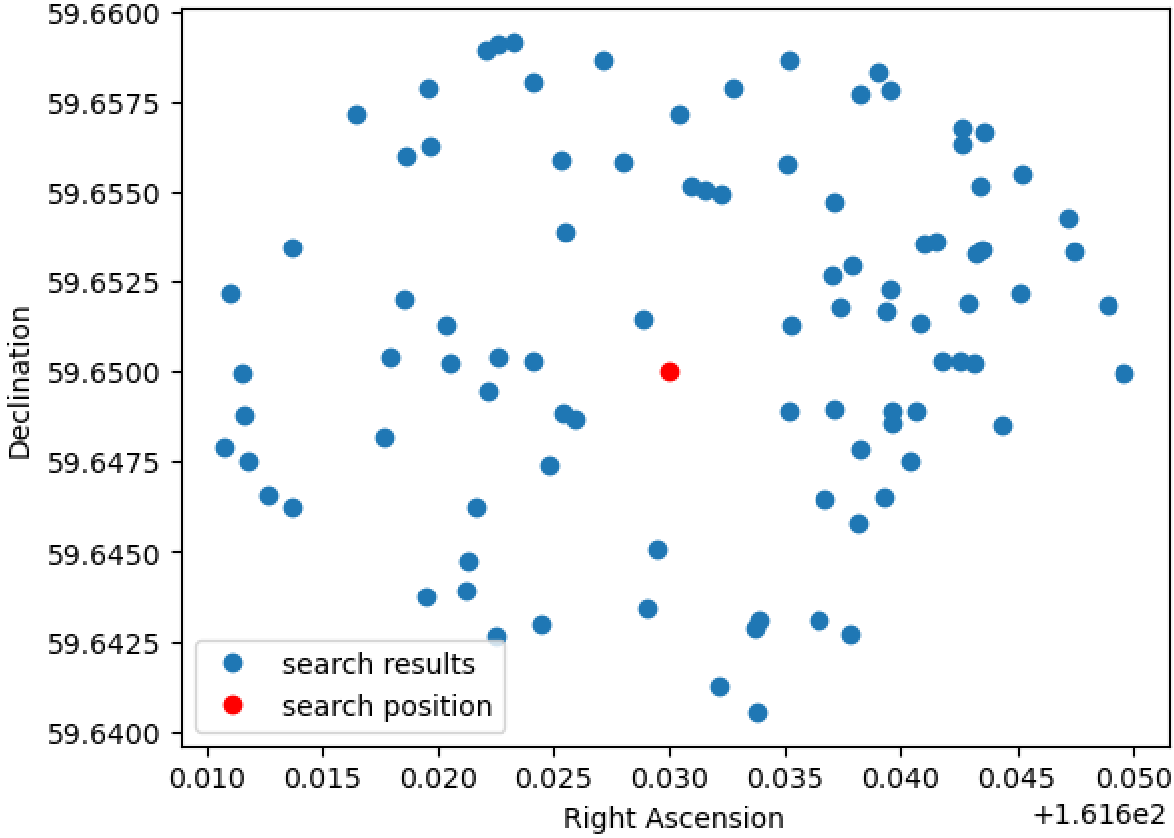


Рисунок 9.5. Отображение объектов на заданном участке

Число объектов: 93

Корреляция между красным смещением и числом галактик

Попробуем изучить корреляцию между красным смещением объектов и их количеством:

```
y=np.array(y)
h = 10000
z = [float(i) for i in y*h]

d = 400
p = 0
b = []
g = []
while p <= h:
    t = 0
    for i in z:
        if i <= p+d:
            t += 1
    b.append(t)
    g.append(p)
    p += d
```

```
print(g)
print(b)

plt.scatter(g, b)
plt.xlabel("z - красное смещение")
plt.ylabel("N - Число галактик")
```

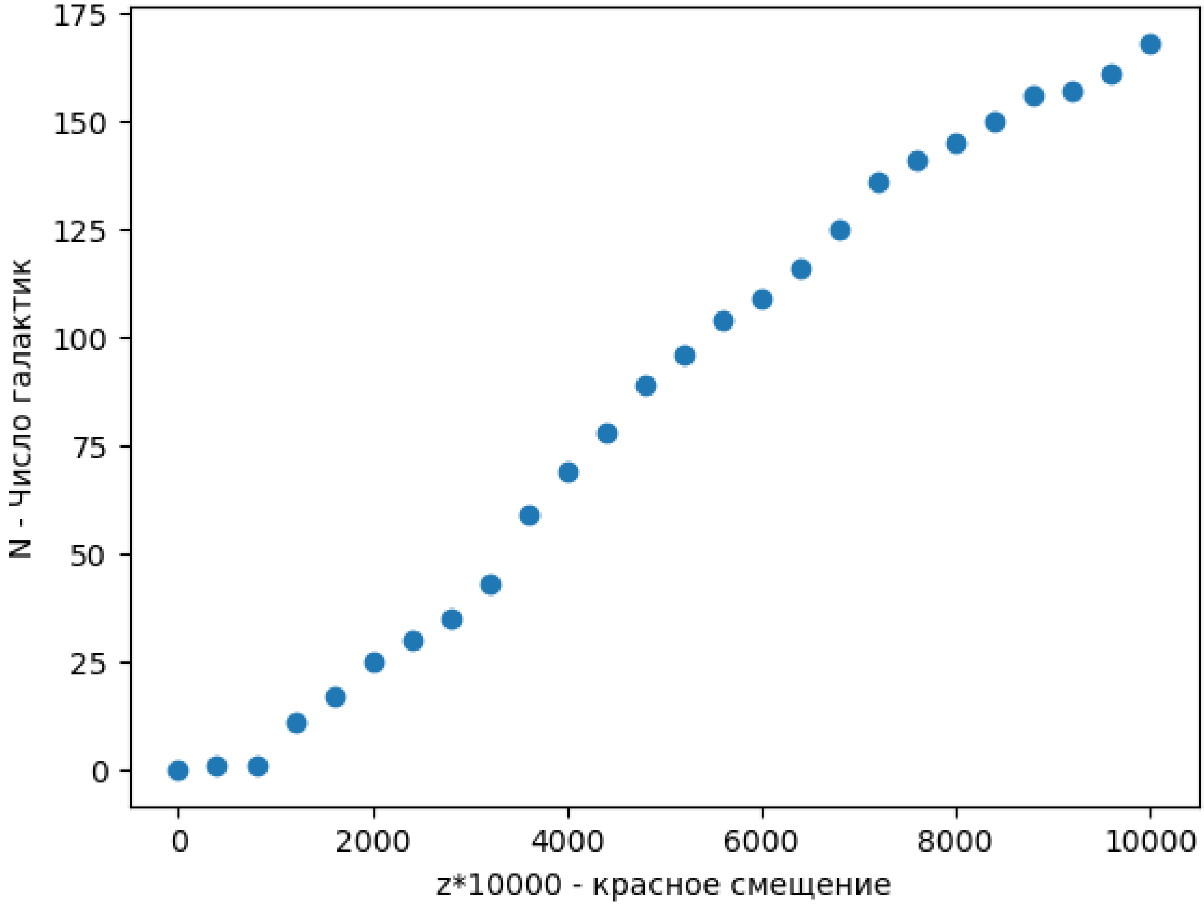


Рисунок 9.6. График корреляции между красным смещением и числом объектов[14]

Мы получили интересную зависимость между красным смещением и числом галактик.

$$N(R) = n(R)V = n(R) \cdot (\frac{4}{3}\pi R^3)$$
$$N(R) \sim z$$
$$z \sim R$$
$$n(R) \sim R^{-2}$$

Т.е. получается, что пространство неоднородно, а концентрация галактик убывает с красным смещением. В данной задаче мы не учитывали типы галактик, ограниченность работы спектрографов Гершеля и нелинейную зависимость между

[14] Данная нормировка была произведена в силу упрощения циклов в Python, поскольку он лучше работает с целыми числами, чем с десятичными дробями с плавающей запятой.

расстоянием и красным смещением, поэтому будут существовать дополнительные поправки, но согласно последним исследованиям пространство Вселенной действительно неоднородно. В целом на базе данной задачи можно построить множество проектных и научных задач по исследованию структуры Вселенной.

Обсерватории HIPPARCOS

Данные о светимости и спектрах звёзд уже в начале XX века были сопоставлены двумя астрономами — Эйнаром Герцшпрунгом (Дания) и Генри Расселлом (США) — и представлены в виде диаграммы, которая получила название **«диаграмма Герцшпрунга-Расселла»**.

Диаграмма Герцшпрунга-Рассела (Г-Р диаграмма) — это графическое представление, показывающее связь между светимостью звезд и их поверхностной температурой. Г-Р диаграмма стала ключевым инструментом в астрофизике, позволяющим астрономам классифицировать звезды и понимать их эволюционные пути.

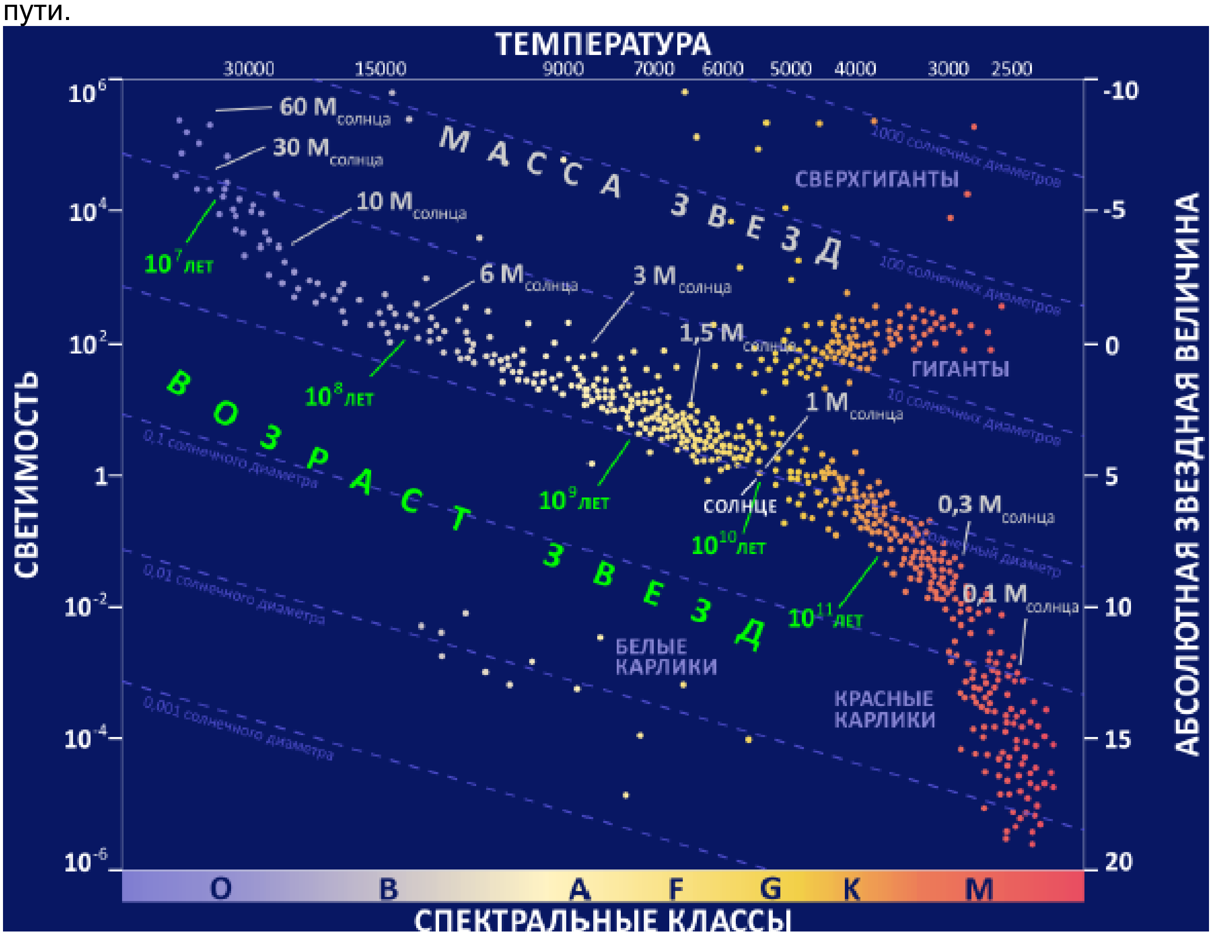


Рисунок 9.7. Диаграмма Герцшпрунга-Расселла

В результате обнаруживается определённая закономерность в расположении звёзд на диаграмме — они не заполняют всё её поле, а образуют несколько групп, названных последовательностями. Наиболее многочисленной (примерно 90% всех звёзд) оказалась **главная последовательность**, к числу звёзд которой принадлежит наше Солнце. Эту стадию характеризуют стабильные термоядерные реакции в ядрах звезд, где водород превращается в гелий. В этот период звезды поддерживают гидростатическое равновесие: давление ионизованного газа и радиации за счет нагрева термоядерными реакциями в ядре звезды уравновешивает силы гравитации, стремящейся сжать звезду. Большинство звезд, включая наше Солнце, находятся на главной последовательности в течение значительной части своей жизни. Звёзды этой последовательности отличаются друг от друга по светимости и температуре, и взаимосвязь этих характеристик соблюдается: самую высокую светимость имеют наиболее горячие звёзды, а по мере уменьшения температуры светимость падает. Красные звёзды малой светимости получили название красных карликов.

Вместе с тем на диаграмме существуют и другие последовательности, где подобная закономерность не соблюдается. Особенно заметно это среди более холодных (красных) звёзд: помимо звёзд, принадлежащих главной последовательности и потому имеющих малую светимость, на диаграмме представлены звёзды высокой светимости, которая практически не меняется при изменении их температуры. Такие звёзды принадлежат двум последовательностям (**гиганты** и **сверхгиганты**), получившим эти названия вследствие своей светимости, которая значительно превосходит светимость Солнца. Особое место на диаграмме занимают горячие звёзды малой светимости — **белые карлики**. Лишь к концу XX века, когда объём знаний о физических процессах, происходящих в звёздах, существенно увеличился и стали понятными пути их эволюции, удалось найти теоретическое обоснование тем эмпирическим закономерностям, которые отражает диаграмма «спектр-светимость».

Время пребывания звёзд на главной последовательности зависит от их первоначальной массы. Чем больше излучение и масса звезды, тем скорее она израсходует свой водород. Большинство звёзд находится на главной последовательности. Это объясняется тем, что продолжительность пребывания на стадии главной последовательности превышает время эволюции на других стадиях.

По прошествии определённого времени — от миллиона до десятков миллиардов лет (в зависимости от начальной массы) — звезда истощает водородные ресурсы ядра. В больших и горячих звёздах это происходит гораздо быстрее, чем в маленьких и более холодных. Истощение запаса водорода приводит к остановке термоядерных реакций. Без давления, возникавшего в ходе этих реакций и уравновешивавшего внутреннюю гравитацию в теле звезды, звезда снова начинает сжиматься, как уже было ранее в процессе её формирования. Температура и давление снова растут, но, в отличие от стадии протозвезды, до гораздо более высокого уровня. **Коллапс** (катастрофически быстрое сжатие массивных тел под действием гравитационных сил) продолжается до тех пор, пока при температуре приблизительно в 100 млн. К не начнутся термоядерные реакции с участием гелия.

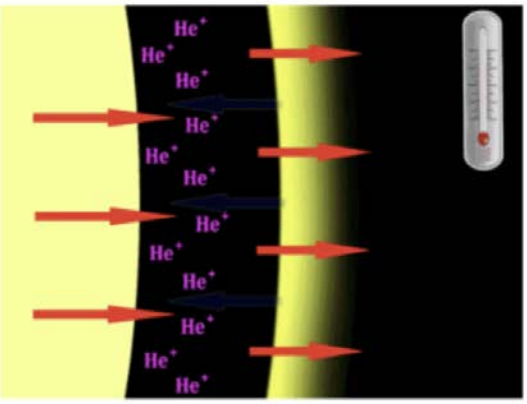
Возобновившееся на новом уровне термоядерное «горение» вещества становится причиной значительного расширения звезды. Звезда увеличивается в размерах приблизительно в 100 раз, а её плотность уменьшается. Так звезда становится красным гигантом, а фаза горения гелия продолжается около нескольких миллионов лет. Практически все красные гиганты являются переменными звёздами**.** Эти звезды имеют очень маленькое ядро (его радиус около 0,001 доли радиуса звезды). Термоядерные реакции происходят в окружающем его тонком слое; далее на протяжении около 0,1 радиуса звезды происходит передача энергии излучением. Практически весь остальной объём (0,9 радиуса) составляет протяжённая конвективная зона.

Причиной изменения блеска, радиуса и температуры является пульсация наружных слоев звезды. Они периодически то расширяются, то сжимаются. При сжатии звезда нагревается и становится ярче, при расширении её светимость уменьшается. Т.е., при повышении температуры в звезде протекают такие процессы, которые вызывают снижение прозрачности во внешнем слое. Это приводит к тому, что излучение энергии колебаний через излучение становится затруднительным, и энергия накапливается в звезде. При сжатии такой звезды излучение, распространяясь из центра к поверхности, сталкивается с этим плотным и горячим слоем. В результате энергия нагревает его еще больше, и слой, как любой газ, расширяется. При расширении он охлаждается и становится более прозрачным. Энергия освобождается, и теперь преобладающая сила тяжести начинает сжимать звезду снова, создавая циклический процесс.

Ключевую роль в данном механизме играет так называемая зона двойной критической ионизации гелия. Это зона, где в ходе пульсаций гелий то ионизируется до состояния "гологo"ядра, то вновь рекомбинирует до однократно ионизированного состояния, что делает его значительно более прозрачным. При сжатии температура увеличивается, гелий более интенсивно ионизируется, поглощая энергию и задерживая её в слое. При последующем расширении гелий рекомбинирует, освобождая энергию, которая уходит наружу.

Этот процесс пульсации атмосферы звезды носит название "клапанный механизм" (аналогично клапанам в тепловых двигателях, которые регулируют теплоотдачу при сжатии). Другое известное название этого механизма каппа-механизм, так как непрозрачность звездного вещества в астрофизике обычно обозначается греческой буквой к (каппа)и называется оптической толщой. Такие колебания в звездах могут сохраняться в течение тысячелетий без значимого затухания.

Максимум радиуса



Минимум радиуса

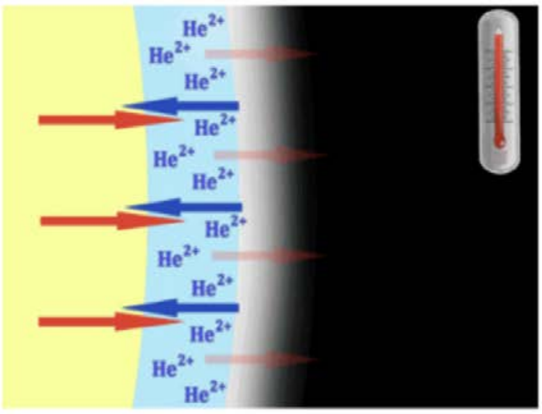


Рисунок 9.8. Распределение вещества в фотосфере звезды в зависимости от фазы. Источник: https://habr.com/ru/articles/481578/

При расширении звезды, площадь, с которой излучается энергия возрастает, это влияет на светимость, однако, температура звезды понижается, что, в свою очередь уменьшает интенсивность излучаемой энергии. Совокупность этих факторов, обуславливает общее изменение светимости звезды, в зависимости от фазы колебания. График зависимости видимого блеска звезды от времени называется кривой блеска. Так как за период колебания переодической звезды, расстояние до нее меняется не значительно, видимый блеск, прямопропорционален светимости звезды.

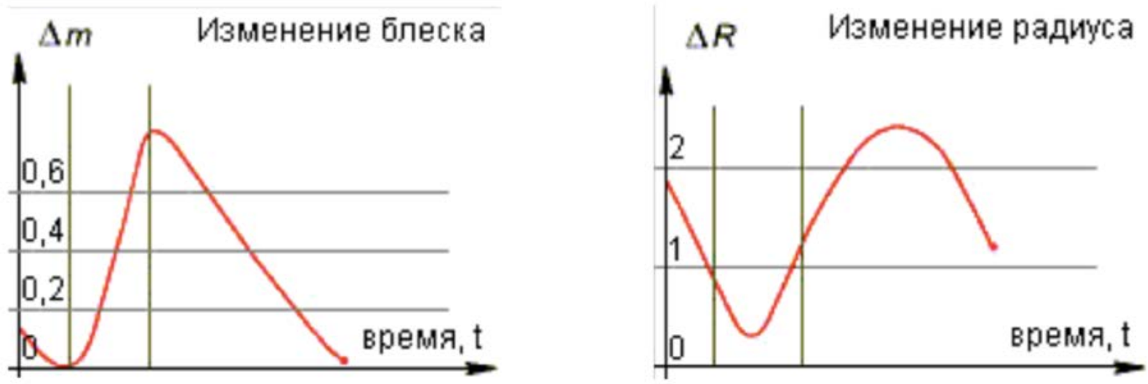


Рисунок 9.9. Кривые блеска и радиуса звезды в зависимости от фазы колебания. Источник: https://checktests.by/p26/

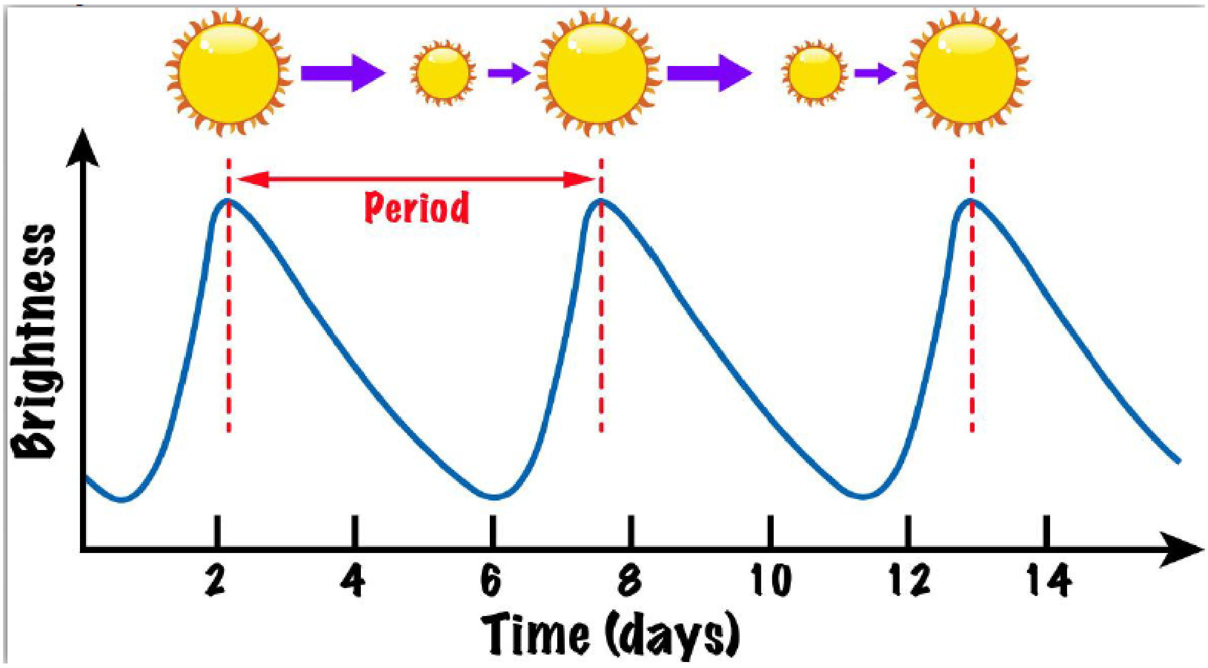


Рисунок 9.10. Фазовая кривая блеска

К числу переменных звезд со строгой периодичностью принадлежат прежде всего цефеиды. Они получили это название потому, что первой среди звезд этого типа была открыта δ Цефея. δ Цефея меняет свою светимость с периодом 5,37 суток, а амплитуда изменения светимости примерно 1^m. У Цефеид эта амплитуда не превышает $1,5^m$, зато периоды изменения светимости весьма различны: от десятков минут до нескольких десятков суток, причем этот период у них долгие годы сохранятеся постоянным.

Вначале 20-го века гарвардский астроном Генриетта Левитт занималась поиском и классификацией переменных звёзд в Магеллановых облаках. Просматривая данные, Левитт заметила, что цефеиды в Малом Магеллановом Облаке, которые имеют большую абсолютную максимальную яркость также имеют более длительный период пульсации. Она подметила, что светимость этих звёзд коррелировала с длительностью их периодов пульсаций. Левитт установила, что существует логарифмическая зависимость между периодами пульсаций и светимостью. Таким образом, если известен период пульсации цефеиды, можно точно расчитать её абсолютную звездную величину, а значит, и расстояние до неё. В 1912 году она опубликовала свои результаты, которые включали данные по 25 цефеидам в Малом Магеллановом Облаке [14]. Итоговая зависимость имеет вид:

$$M_v = (-2.43 \pm 0.12) \cdot (log10P - 1) - (4.05 \pm 0.02) \quad (10)$$

Где P - период переменной звезды в днях, M_v средняя абсолютная звездная величина.

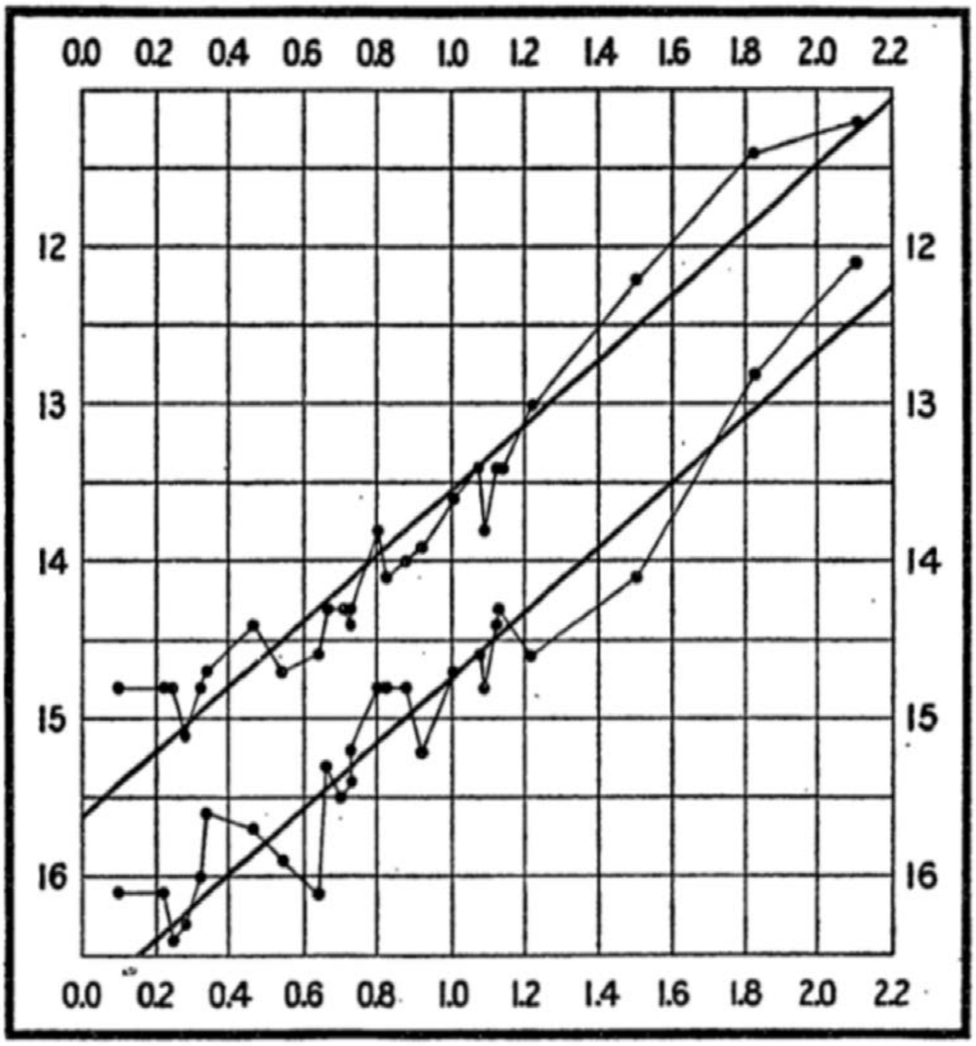


FIG. 2.

Рисунок 9.11. Полученная Левитт зависимость "период—светимость"для классических цефеид: $M_v = (-2.43 \pm 0.12) \cdot (log10P - 1) - (4.05 \pm 0.02)$. По оси абсцисс логарифм периода в днях, по оси ординат магнитуда в звездных величинах. Источник: https://en.wikipedia.org/wiki/Period-luminosity_relation

Цефеиды – звёзды-сверхгиганты, они обладают высокой светимостью. Так, например, светимость цефеиды с периодом 50 суток в 10 тыс. раз больше, чем у Солнца. Они заметны даже в других галактиках, поэтому их можно использовать для определения таких больших расстояний, когда годичный параллакс невозможно измерить. Цефеиды часто называют «маяками Вселенной» или «стандартными свечами».

Чем ярче цефеида, тем продолжительнее период изменения её светимости.

Зависимость «период-светимость», существующая у цефеид, используется для определения расстояний в астрономии. Получив из наблюдений период изменения светимости цефеиды, можно узнать её светимость, вычислить абсолютную звёздную величину M, а сравнив её с видимой звёздной величиной m, вычислить расстояние D до звезды: $\lg D = 0.2\,(m-M)+1$. Следовательно, обнаружив в какой-либо галактике цефеиду и измерив период колебания ее светимости, можно по этой заивисмости определить ее абсолютную звезд величину, а значит, и расстояние до галактик.

Эдвин Хаббл использовал закон Левитт для определения расстояний до далёких галактик, что в итоге привело к открытию существование других звездных систем. Это в корне изменило представления человечества о космосе. Понимание фактического размера Вселенной значительно улучшилось, поскольку учёные могли теперь измерять расстояния до других галактиках. Эдвин Хаббл построил диаграмму "скорости галактик расстояние до них что привело к открытию расширяющейся эволюционирующей Вселенной.

Такми образом, работа Генриетты Левитт имела колоссальное значение, поскольку впервые предоставила точный и надёжный метод для измерения расстояний в космосе до далёких объектов, к которым не применим метод параллакса. Это открытие позволило ученым установить масштабы Вселенной и лучше понять её структуру. Работа Генриетты Левитт оказала революционное воздействие на астрономию, делая её одним из ключей к пониманию самых фундаментальных вопросов о строении и эволюции Вселенной.

Сегодня можно повторить работу Генриетты Левитт, используя современные астрономические каталоги данных по переменным звездам, например, полученных космической обсерваторий **HIPPARCOS.**

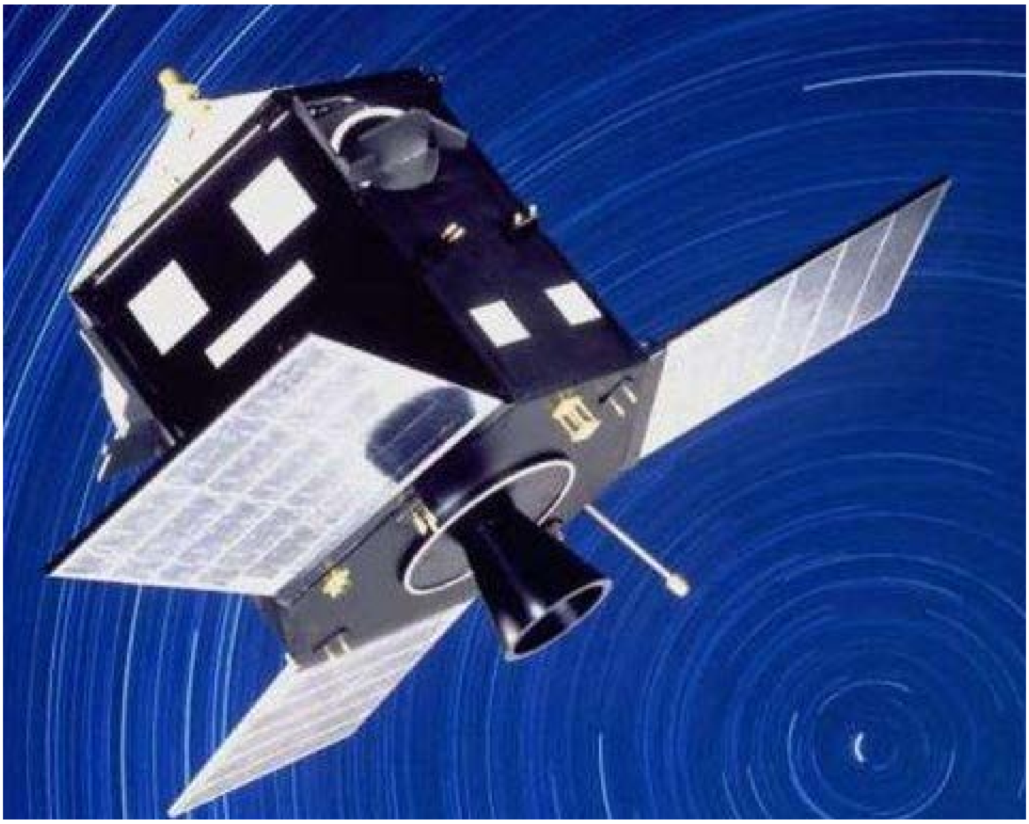


Рисунок 9.12. Космическая обсерватория HIPPARCOS

В 1989 году Европейское Космическое Агентство (ESA) осуществило запуск космического аппарата HIPPARCOS (HIgh Precision PARallax COllecting Satellite - "спутник для сбора высокоточных параллаксов“, что созвучно с именем древнегреческого астронома Гиппарха – составителя первого в Европе звёздного каталога) с целью получения координат, собственных движений и параллаксов звезд с точностью измерений до 10^{-3} mas (milli arc second - миллисекунд дуги). Космический аппарат проработал на орбите 37 месяцев (до 1993 г), в течение которых он выполнял астрометрические и фотометрические измерения звезд по заданной программе. Hipparcos смог измерить углы параллакса для звезд на расстоянии около 1600 световых лет от нас, что составляет чуть более одного процента диаметра Галактики Млечный Путь. Это была первая спутниковая миссия, полностью посвящённая астрометрии, науке о точных измерениях положений, расстояний и движений небесных объектов.

Космическая обьсерватория HIPPARCOS была оснащена телескопом, который с высокой точностью измерял положения звёзд. В совокупности с передовыми технологиями в области детекторов и обработки данных, это позволило HIPPARCOS достичь точности измерения угловых расстояний порядка миллисекунд дуги. ТЕлесков не только измерял положения звёзд, но и следил за их изменениями со временем, что позволило получить точные значения параллаксов и собственных движений для более чем 118000 звёзд.

Ещё одной важной особенностью миссии была система звёздного сканирования, известная как большой круговой скан. Спутник вращался медленно, проходя по большому кругу на небосводе и сканируя множество звёзд за один проход. Полученные данные обрабатывались на борту и передавались на Землю для дальнейшей обработки. Этот метод позволил обеспечить высокую точность измерений для большого количества звёзд по всему небу.

Результаты миссии HIPPARCOS имели огромное значение для астрономии. Они предоставили картину точных расстояний и движений звёзд в нашей Галактике, что является фундаментальной информацией для многих областей астрономии. Эти данные позволили пересмотреть и уточнить модели строения и эволюции нашей ГАлактики Млечного Пути, а также лучше понять динамику входящих в неё звёздных систем. Миссия HIPPARCOS выступила важным шагом в эру прецизионной астрономии, закладывая основы для последующих астрометрических миссий, таких как космическая обсерватория GAIA, которая продолжает развивать успехи HIPPARCOS с ещё более высокой точностью и объёмом данных.

Отображение данных

Вначале попробуем скачать и открыть звездный каталог HIPPARCOS[15]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
import matplotlib.cm as cm
from astropy.table import Table

df = pd.read_csv("Downloads/hipparcos-voidmain.csv")
df
```

[15] Файл для скачивания доступен по ссылке: https://heasarc.gsfc.nasa.gov/W3Browse/all/hipparcos.html

Файл содержит 78 столбцов и 113710 строк. Для определения подзаголовков можно воспользоваться функцией df.info(), а более подробную информацию о том, какие параметры есть в данном каталоге можно определить по ссылке: https://www.astronet.ru/db/msg/1210304/node8.html. Теперь попробуем отобразить эти данные на графике:

```
plt.figure(figsize = (5,5))
plt.scatter(df['RAdeg'], df['DEdeg'], s= 0.01)
plt.xlabel('RA')
plt.ylabel('DE')
plt.show()
```

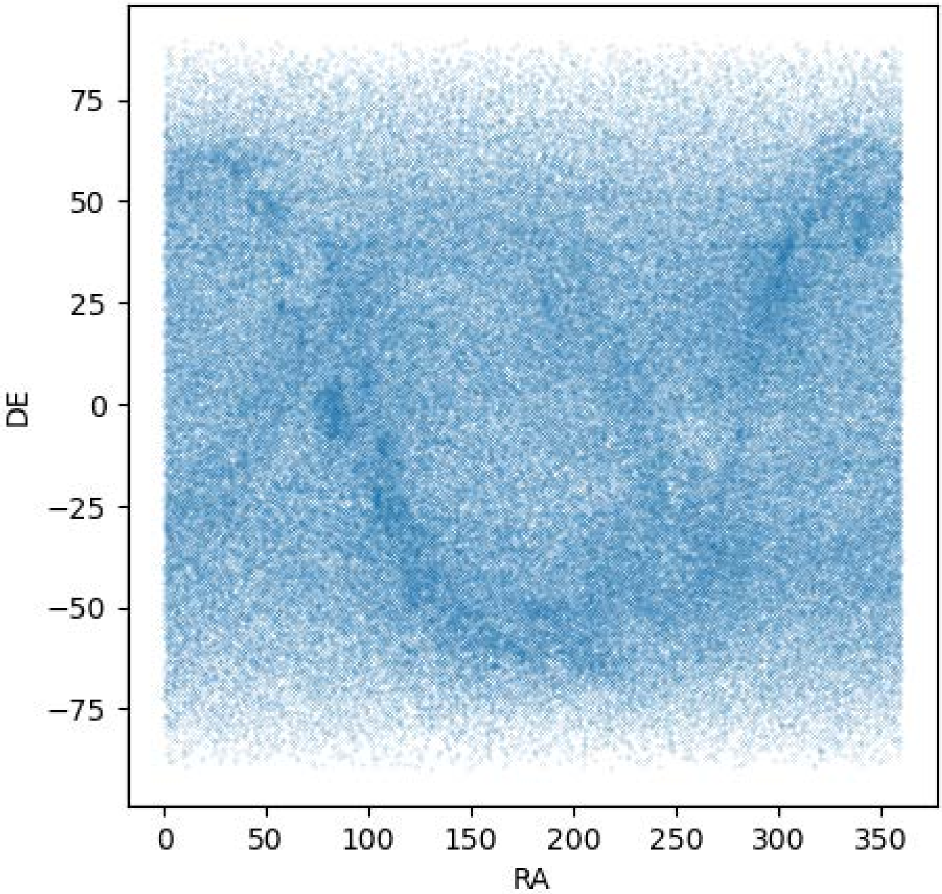


Рисунок 9.13. Отображение данных каталога HIPPARCOS в плоскости

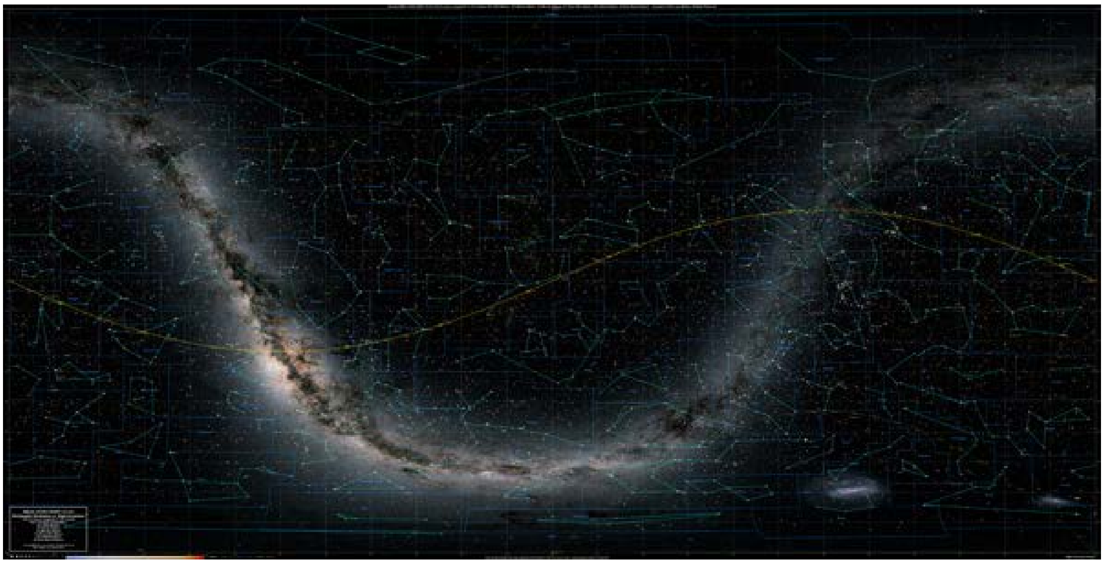


Рисунок 9.14. Проекция Млечного пути на плоскость (Calgary Centre)

Для лучшей картины можем воспользоваться проекцией Мольвейде:

```
fig = plt.figure(figsize=(8,6))
ax = fig.add_subplot(111, projection="mollweide")
ax.grid(True)
ax.set_xticklabels(['14h','16h','18h','20h','22h','0h','2h','4h','6h','8h','10h'])
ax.scatter(np.radians(df['RAdeg']-180), np.radians(df['DEdeg']), s= 0.01)
```

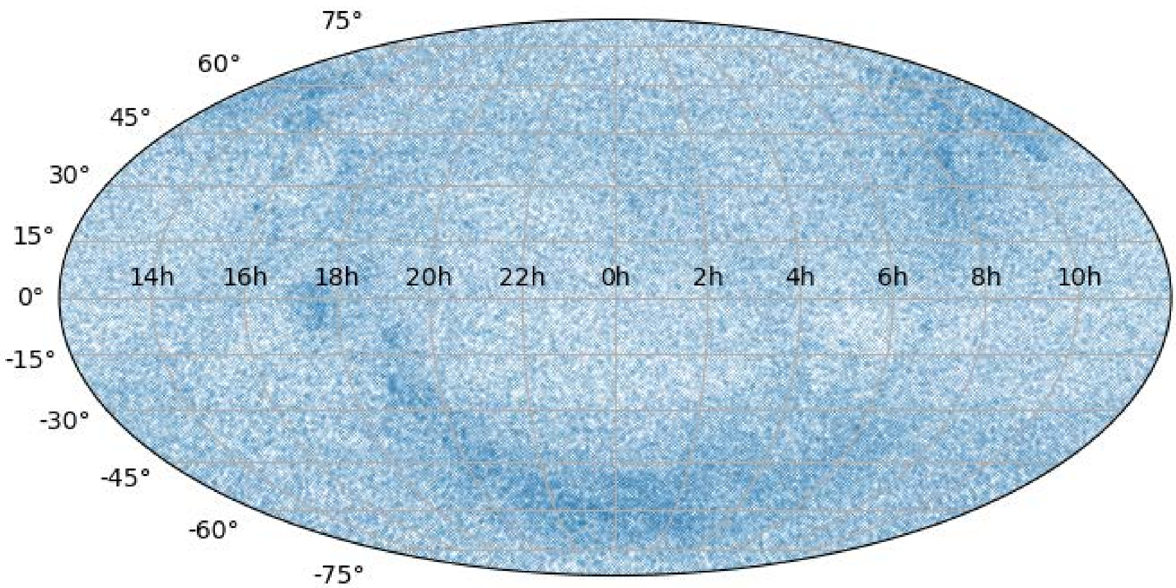


Рисунок 9.15. Отображение данных каталога HIPPARCOS в проекции Мольвейде

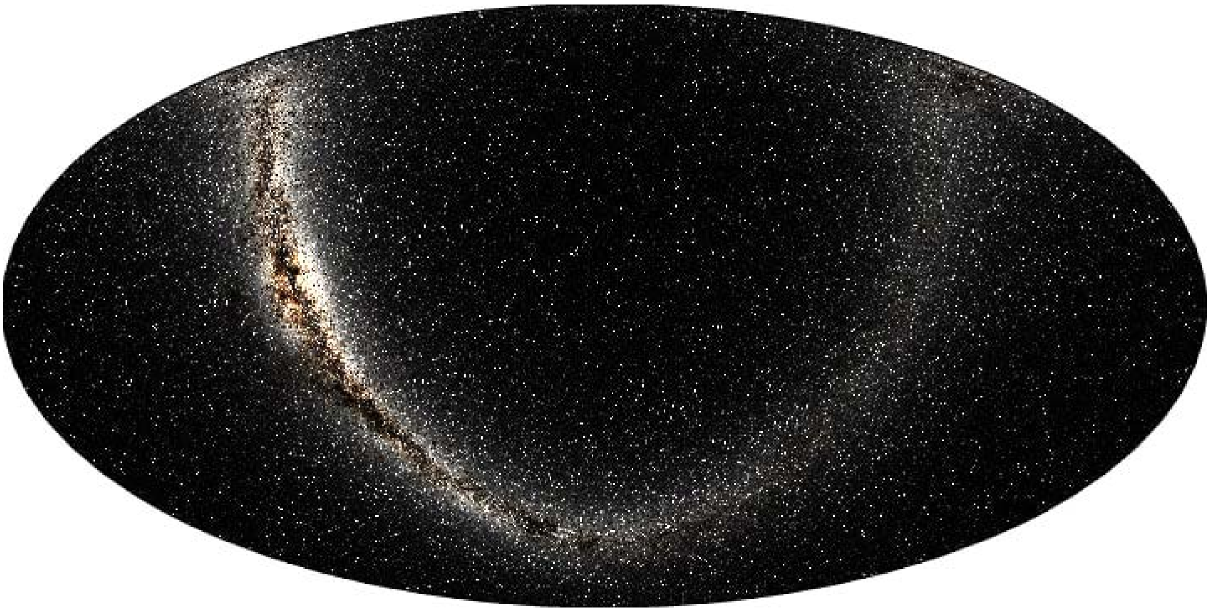


Рисунок 9.16. Млечный путь в проекции Мольвейде. https://www.researchgate.net/figure/The-integrated-starlight-model-in-full-sky-view-real-color-in-logarithmic-scale_fig3_372685324

Но это распределение звезд в экваториальных координатах не очень информативно, так ничего не говорит нам о распределении звезд в Галактике. Поэтому отобразим это распределение в галактических координатах. Для этого выведем формулу перевода координат в галактические и напишем функции для перевода:

$$sin\, b = sin\, \delta \cdot sin\, \delta_0 \; + \; cos\, \delta \cdot cos\, \delta_0 \cdot cos(\alpha - \alpha_0)$$

$$tg(l - l_0) = \frac{sin\, \delta \cdot cos\, \delta_0 \; - \; cos\, \delta \; \cdot sin\, \delta_0 \cdot cos(\alpha - \alpha_0)}{cos\, \delta \; \cdot \; sin(\alpha \; - \; \alpha_0)}$$

Алгоритм по переводу экваториальных координат в галактические. Для галактической долготы необходимо поставить условие, поскольку Python воспроизводит arctg только в пределах $[-\pi/2;\, \pi/2]$:

```
def galactic_coordinates_b(row, alpha0=np.radians(192.85), np.radians(delta0=27.13), 
l0=np.radians(122.3)):
   ra = np.radians(row['RAdeg'])
   de = np.radians(row['DEdeg'])
   sin_b = np.sin(de) * np.sin(delta0) + np.cos(de) * np.cos(delta0) * np.cos(ra - alpha0)
   b = np.arcsin(sin_b)
   b = b * 180 / np.pi
   return b

def galactic_coordinates_l(row, alpha0=np.radians(192.85), np.radians(delta0=27.13), 
l0=np.radians(122.3)):
   ra = np.radians(row['RAdeg'])
   de = np.radians(row['DEdeg'])
   sin_b = np.sin(de) * np.sin(delta0) + np.cos(de) * np.cos(delta0) * np.cos(ra - alpha0)
   cos_b = np.cos(np.arcsin(sin_b))
```

```
    sin_delta_l = np.cos(de) * np.sin(ra - alpha0)
    cos_delta_l = np.cos(delta0) * np.sin(de) - np.sin(delta0) * np.cos(de) * np.cos(ra -
alpha0)
    delta_l = np.arctan(sin_delta_l/cos_delta_l)
    if (sin_delta_l/cos_b) > 0.0:
        if (cos_delta_l/cos_b) > 0.0:
            l = l0 - delta_l
        else:
            l = l0 - delta_l+np.pi
    else:
        if (cos_delta_l/cos_b) > 0.0:
            l = l0 - delta_l
        else:
            l = l0 - delta_l+np.pi
    l = l * 180 / np.pi
    if l > 360:
        l = l - 360
    return l
```

Теперь отобразим данный каталог в галактических координатах:

```
df['GCdeg_b'] = df.apply(galactic_coordinates_b, axis=1)
df['GCdeg_l'] = df.apply(galactic_coordinates_l, axis=1)

fig = plt.figure(figsize=(8,6))
ax = fig.add_subplot(111, projection="mollweide")
ax.grid(True)
ax.set_xticklabels(['14h','16h','18h','20h','22h','0h','2h','4h','6h','8h','10h'])
ax.scatter(np.radians(df['GCdeg_l']-180), np.radians(df['GCdeg_b']), c='g', s=0.005)
```

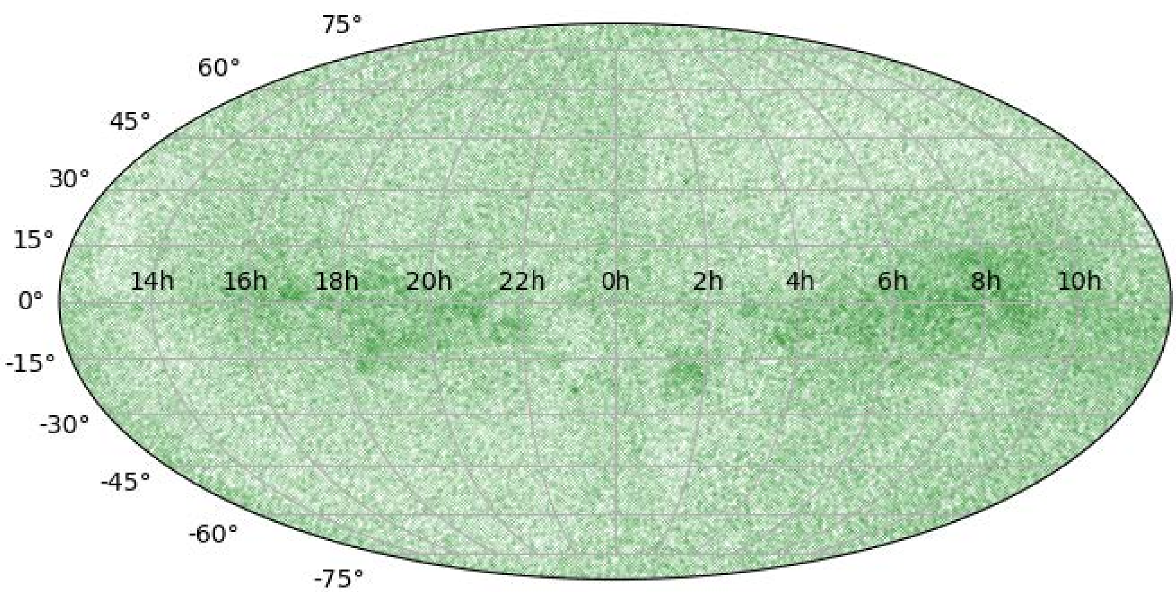


Рисунок 9.17. Отображение данных каталога HIPPARCOS в галактических координатах в проекции Мольвейде

Важный вывод**:** из этого рисунка можно сделать вывод, что звезды преимущественно концентрируются к плоскости галактического экватора. Т.е. анализируя наблюдательные данные, по характеру распределения звезд в галактических координатах можно сделать очень важный вывод о типе нашей Галактики: звезды в основном распределены в плоскости диска следовательно наша галактика - дисковая. Наблюдать за звездами близкими к центру галактики сложно из-за большого количества межзвездного газа, поэтому на графике наблюдается некая пустота вблизи галактического центра, соответствующего началу координат.

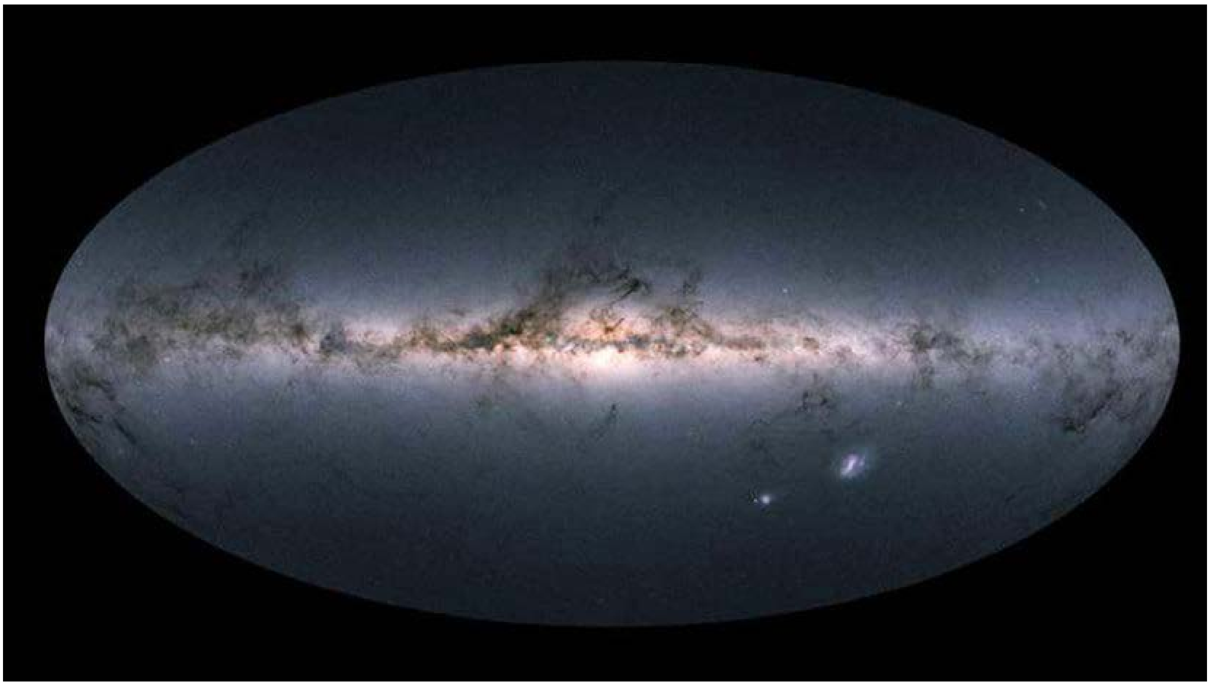


Рисунок 9.18. Изображение Млечного пути в галактических координатах в проекции Мольвейде с телескопа Гайя.

Повторение классической работы «период-светимость» Г.Левитт по цефеидам.

Теперь попробуем повторить идею работы Г. Левитт по построению гистограммы для звезд разных спектральных классов для, чтобы отобрать подкласс переменных звезд – цефеид. По данным звездам построим зависимость “период-светимость”.

В начале сделаем перевод из миллисекунд параллакса в парсеки:

$$L(\text{пк}) = \frac{1}{\pi('')}$$

А также сделаем перевод видимых звездных величин в абсолютные:

$$M = m - 5lg\frac{d}{10}$$

```
def distanse(Plx):
    return 1000/(Plx)

def luminosity(row):
    dist = distanse(row['Plx'])
```

```
    return row['Vmag']-5*np.log10(dist/10)

# Отбор звезд заданного типа
def SpType(row):
    return str(row['SpType'])[:1]

# Отбор звезд более широкого типа
def SpType_extended(row):
    return str(row['SpType'])[:2]

df['luminosity'] = df.apply(luminosity, axis=1)
df['Type'] = df.apply(SpType, axis=1)
df['Type_extended'] = df.apply(SpType_extended, axis=1)
```

Теперь построим гистограмму распределения количества звезд от их периода:

```
classes = ['O', 'B', 'A', 'F', 'G', 'K', 'M']

fig = plt.figure(figsize=(15,10))
gs = fig.add_gridspec(2, 4, hspace=0, wspace=0)

# Построим гистограммы периодов от спектрального класса
(ax1, ax2, ax3, ax4), (ax5, ax6, ax7, ax8) = gs.subplots(sharex='col', sharey='row')
axs= [ax1, ax2, ax3, ax4, ax5, ax6, ax7, ax8]

# Возьмем только переодические звезды
variable = df[df['HvarType']=='P']
fig.supxlabel('Period, days')
fig.supylabel('num_stars')

for cls, ax in zip(classes, axs):
    temp = variable[variable['Type'] == cls]
    ax.set_title(cls)
    ax.hist(temp[temp['Period'] < 200]['Period'], bins=50)

plt.show()
```

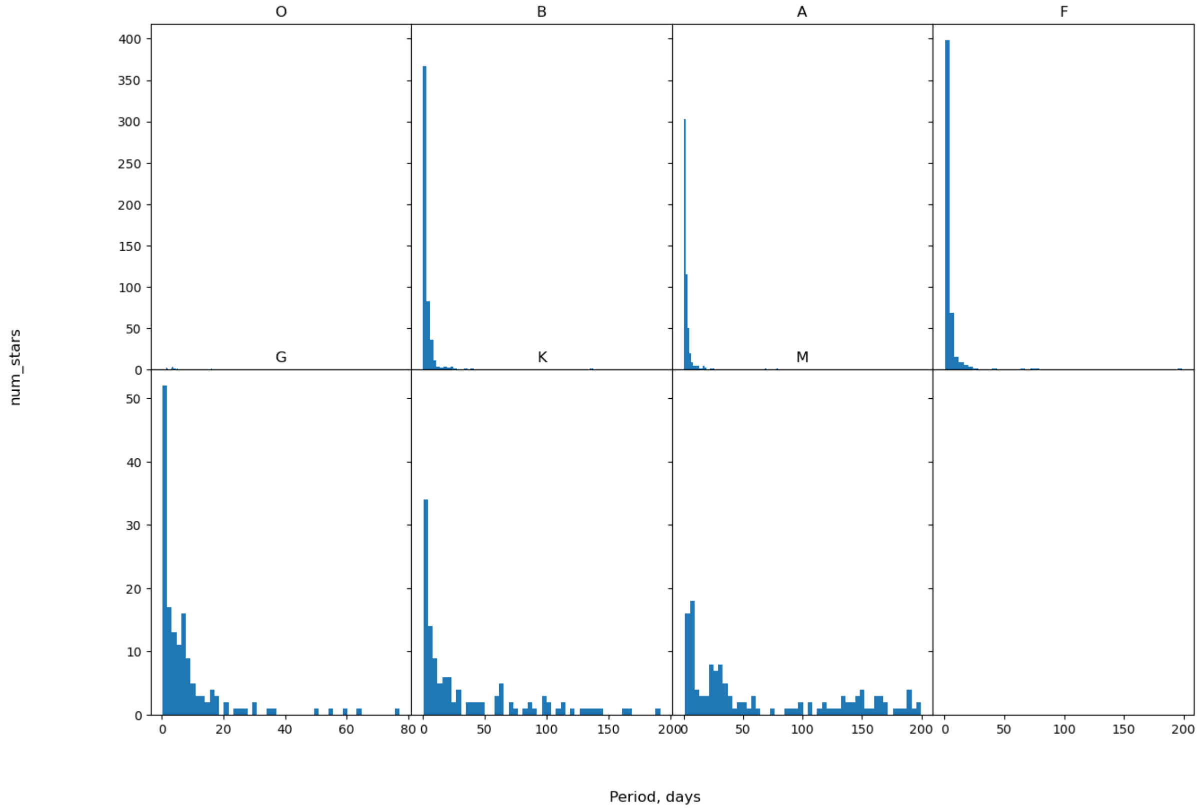


Рисунок 9.19. Гистограмма распределения звёзд в каталоге HIPPARCOS в зависимости от периода

Из данной гистограммы можно заметить, что наиболее плотное распределение звезд в периоде от 1 до 100 суток будет у классов: G, K, M. Значит данные звезды можно воспринимать, как классические цефеиды. Теперь попробуем построить график зависимости абсолютной звёздной величины цефеид от периода:

```
plt.figure(figsize=(10,5))

plt.xlabel('log(period)')
plt.ylabel('M_k')

cef = ['G0', 'G1', 'G2', 'G3', 'G4', 'G5','G6', 'G7','G8', 'G9', 'K0','K1', 'K2', 'K3','K4', 'K5',
'K6','K7', 'K8', 'K9']
cls = ['G', 'K']

ceffiids = df[(df['HvarType']=='P') & (df['Type_extended'].isin(cef))]

#Возьмем звезды с периодом больше 0 и меньше 100 дней
ceffiids = ceffiids[np.log10(ceffiids['Period']) > 0]
ceffiids = ceffiids[np.log10(ceffiids['Period']) < 2]

colors = ['red', 'blue']

for c, cl in zip(colors, cls):
    cef_class = ceffiids[ceffiids['Type'] == cl]
```

```
    plt.scatter(np.log10(cef_class['Period']), cef_class['luminosity'], color=c, label =cl, s= 5)

plt.legend()
ax = plt.gca()
ax.invert_yaxis()
plt.grid()

plt.show()
```

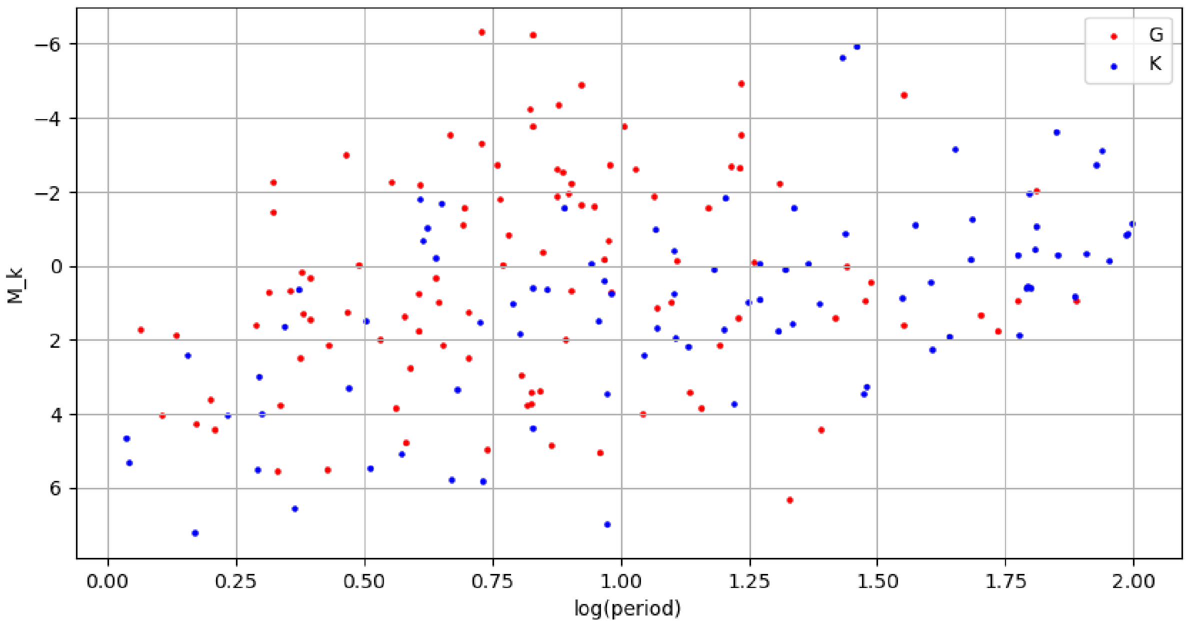


Рисунок 9.20. Распределение цефеид из каталога HIPPARCOS по светимости и периоду

Отметим, что существует некоторая зависимость между данными величинами. Попробуем определить параметры данной зависимости с помощью МНК:

```
T = np.log10(cef_class['Period'])
T_np = T.values
M = cef_class['luminosity']
M_np = M.values
a, b = MNK([T_np, M_np])
print(a, b)
x = np.arange(0.0, 2.5, 0.1)
y = a*x+b
plt.plot(x, y, c='black', label = '-2.75*log(p)+4.17')
```

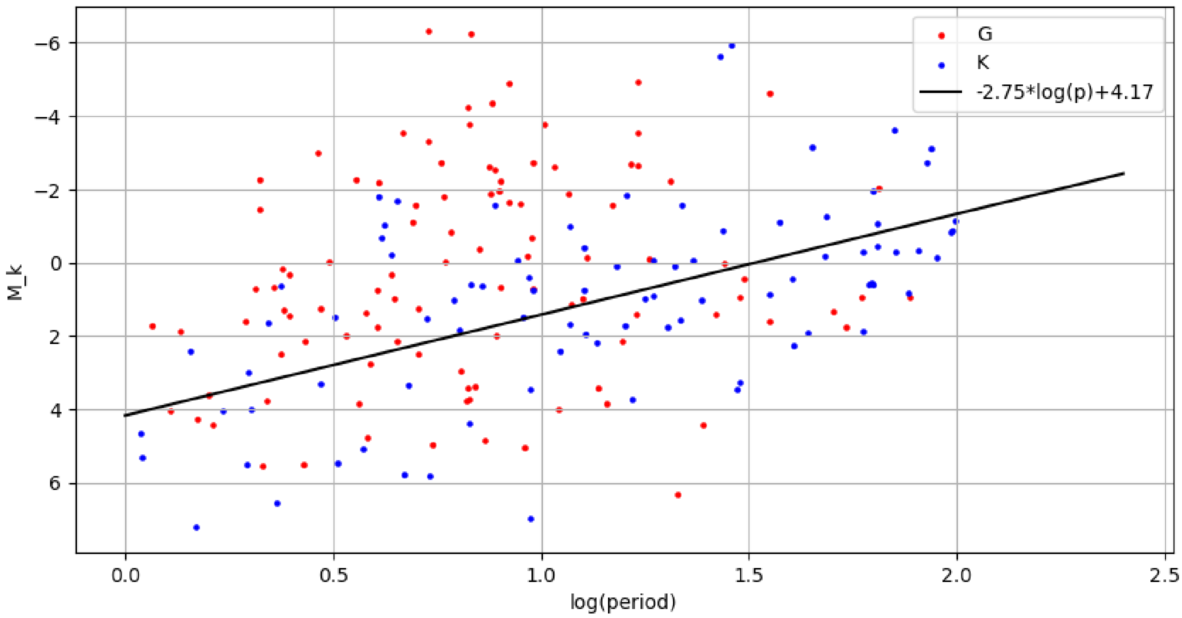


Рисунок 9.21. Построение линейной зависимости между светимостью и периодом

Данную зависимость можно описать линейным законом:

$$M = -2{,}75 \cdot log(P) + 4{,}17$$

$$M = -2{,}75 \cdot (log(P) - 1) + 1{,}42$$

Построенная из наблюдаемых данных зависимость имеет достаточно сильные отклонения от подгоночной кривой, что говорит о не очень тщательно отобранных среди переменных звезд именно цефеид, но тем не менее, значение полученных праметров очень близки к тем, что в своей работе получала Генриетта Левитт.

Эта работа является важнейшим этапом в построении шкалы расстояний во Вселенной - она позволила Э.Хабблу определить расстояния до других галактик, а также построить диаграмму Хаббла, что привело к открытию расширения Вселенной и оценке её размеров. А теперь, используя каталоги цефеид уже современных космических обсерваторий, школьники и студенты могут самостоятельно повторить эту работу и воспроизвести зависимость «период-светимость».

Список литературы

Список литературы

[1] Проект GAIA VARI: https://www.gaiavari.space/
[2] МиссияGAIA: https://www.esa.int/Science_Exploration/Space_Science/Gaia_overview/
[3] Миссия HIPPARCOS: https://www.esa.int/Science_Exploration/Space_Science/Hipparcos_overview
[4] Каталог HIPPARCOS: https://heasarc.gsfc.nasa.gov/W3Browse/all/hipparcos.html
[5] "Федеральный Государственный Образовательный Стандарт": https://fgos.ru/fgos/fgos-soo/.
[6] Воронцов-Вельяминов Б. А., Страцт Е. К. "Астрономия. Базовый курс. 11 класс. ISBN: 978-5-358-20451-5
[7] Буминович Е. А., Буличев Б. А."Теория вероятности и статистика.(10-11 класс) (Базовый и продвинутый уровень)"ISBN: 978-5-09-112145-2
[8] Засов А.Б., Постнов К.А. "Общая астрофизика".Фрязино: "Век"2 2011
[9] Левитан Е. П., "Астрономия. Базовый уровень. 11 класс.Просвещение 2022
[10] Чугарин Б. М., "Астрономия. Базовый уровень. Класс 10-11."ФГОС, "Просвещение2022
[11] Peer-review: https://ru.wikipedia.org/wiki/Рецензирование
[12] Principles of Effective School-to-School Peer Review, NAHT, 2019: https://www.schoolspartnershipprogramme.com/latest-news/report-the-principles-ofeffective-school-to-schoo
[13] General Catalogue of Variable Stars: http://www.sai.msu.su/gcvs/gcvs/
[14] Leavitt, Henrietta S.; Pickering, Edward C. Periods of 25 Variable Stars in the Small Magellanic Cloud. Harvard College Observatory Circular: journal.— 1912.— Vol. 173.— P. 1—3. https://ui.adsabs.harvard.edu/abs/1912HarCi.173....1L/ .