Python for Astronomy

Python 基础知识

何勃亮
中国科学院国家天文台 中国虚拟天文台 (China-VO)

Outline

- Python 基础
- Python 科学计算
- Python 与天文学计算
- 案例分享
    - 赵永恒: 使用Python进行历法计算
    - 余恒: 用Python搞定星表
    - 王鑫：使用Python方便快捷高效的载入输出数据，浅谈yaml在天文中的应用

课程特点

- 不特别讲述深奥的知识点，比如面向对象编程，模块和包的开发等
- 实用性，着重分解最常用的一些功能和用法
- 案例分享

Python 基础知识

- Python 简介
- Python 语法
- Python 容器
- Python 代码风格(`PEP8`)
- 软件包管理
- Python 2 和 Python 3
- Python 编程范式

Python 简介

历史 (https://zh.wikipedia.org/wiki/Python#.E6.AD.B7.E5.8F.B2)

Python的创始人为吉多·范罗苏姆（Guido van Rossum）。1989年的圣诞节期间，吉多·范罗苏姆为了在阿姆斯特丹打发时间，决心开发一个新的脚本解释程序，作为ABC语言的一种继承。之所以选中Python作为程序的名字，是因为他是BBC电视剧——蒙提·派森的飞行马戏团（Monty Python's Flying Circus）的爱好者。ABC是由吉多参加设计的一种教学语言。就吉多本人看来，ABC这种语言非常优美和强大，是专门为非专业程序员设计的。但是ABC语言并没有成功，究其原因，吉多认为是非开放造成的。吉多决心在Python中避免这一错误，并获取了非常好的效果，完美结合了C和其他一些语言。

就这样，Python在吉多手中诞生了。实际上，第一个实现是在Mac机上。可以说，Python是从ABC发展起来，主要受到了Modula-3（另一种相当优美且强大的语言，为小型团体所设计的）的影响。并且结合了Unix shell和C的习惯。

目前吉多仍然是Python的主要开发者，决定整个Python语言的发展方向。Python社区经常称呼他是仁慈的独裁者。

Python 2.0于2000年10月16日发布，增加了实现完整的垃圾回收，并且支持Unicode。同时，整个开发过程更加透明，社区对开发进度的影响逐渐扩大。Python 3.0于2008年12月3日发布，此版不完全兼容之前的Python源代码。不过，很多新特性后来也被移植到旧的Python 2.6/2.7版本。

Python是完全面向对象的语言。函数、模块、数字、字符串都是对象。并且完全支持继承、重载、派生、多重继承，有益于增强源代码的复用性。Python支持重载运算符，因此Python也支持泛型设计。相对于Lisp这种传统的函数式编程语言，Python对函数式设计只提供了有限的支持。有两个标准库（functools, itertools）提供了与Haskell和Standard ML中类似的函数式程序设计工具。

虽然Python可能被粗略地分类为“脚本语言”（script language），但实际上一些大规模软件开发项目例如Zope、Mnet及BitTorrent，Google也广泛地使用它。Python的支持者较喜欢称它为一种高级动态编程语言，原因是“脚本语言”泛指仅作简单程序设计任务的语言，如shell script、VBScript等只能处理简单任务的编程语言，并不能与Python相提并论。

Python本身被设计为可扩充的。并非所有的特性和功能都集成到语言核心。Python提供了丰富的API和工具，以便程序员能够轻松地使用C、C++、Cython来编写扩充模块。Python编译器本身也可以被集成到其它需要脚本语言的程序内。因此，有很多人把Python作为一种“胶水语言”（glue language）使用。使用Python将其他语言编写的程序进行集成和封装。在Google内部的很多项目，例如Google App Engine使用C++编写性能要求极高的部分，然后用Python或Java/Go调用相应的模块。《Python技术手册》的作者马特利（Alex Martelli）说：“这很难讲，不过，2004年，Python已在Google内部使用，Google召募许多Python高手，但在这之前就已决定使用Python。他们的目的是尽量使用Python，在不得已时改用C++；在操控硬件的场合使用C++，在快速开发时候使用Python。”

Python 老爹怎么说（PyCon2016）

Guido van Rossum

- **Python的性能提升** Python 3的性能已经跟上来了，比2012年时要快的多。另外，还有像PyPy这样的Python实现。有一些新版本的Python解释器也在试图提升速度。 其实，Python的性能并没有人们说的那样差，而且因为Python大部分是用C语言实现的，很多事情做起来可以和C语言一样快。我还是认为，Python对于大部分事情来说已经足够快了。

  尽管没有往Python 3中新增特性以改善速度，但是我们已经让语言的很多方面变快了：比如，引用计数比以前快了些。主要还是优化现有的代码，但是作为用户来说，很难注意到区别。

  而且如果你急需提升某个Python程序的速度，可以尝试使用PyPy。它已经足够成熟，值得尝试。
- **SciPy和NumPy** 这两个团队正在推动使用Python替代Matlab。我们的替代方案是开源的，而且更好，他们能做到的。他们正在将Python带领到从未想象过的领域。他们开发出了像Jupyter Notebooks这样的工作，可以在浏览器中使用交互式Python。

Who use Python?

- Radio / Submm (NRAO, ESO, JAOJ, CSIRO): CASA
- IR: HIPE (Herschel Interactive Processing Environment)
- Optical: STSci (PyRAF, PyFITS)
- Optical: Gemini IRAF package - new development in Python
- Optical: ESO PyMidas
- X-ray: Chandra CIAO and Sherpa
- Gamma-ray: Fermi Science Analysis tools

安装

- Windows `Anaconda`
    - https://www.continuum.io/downloads (https://www.continuum.io/downloads)
    - `NumPy,SciPy,Matplotlib,Pandas,IPython,Spyder`
- MacOS `brew install python3`
- Linux

```
yum -y install gcc gcc-c++ make gcc-gfortran bzip2 bzip2-devel bison \
flex readline-devel sqlite-devel gdbm-devel xz-devel xz-libs

./configure --prefix=/usr/local --enable-ipv6 --enable-shared
make -j4
sudo make install
```

下载安装

`Anaconda` http://t.cn/R5CKu9m (https://hebl.china-vo.org/course/PIA2016/software/)

https://hebl.china-vo.org/course/PIA2016/software/ (https://hebl.china-vo.org/course/PIA2016/software/)

Anaconda

- NumPy
- SciPy
- IPython / Jupyter
- Matplotlib
- AstroPy
- Spyder

更多软件列表：https://docs.continuum.io/anaconda/pkg-docs (https://docs.continuum.io/anaconda/pkg-docs#)

Python 语法

数据类型

- 整形 `Integer`
- 浮点型 `Floating`
- 复数 `Complex`
- 字符串 `String`

In [1]:

```
a = 10
b = 12.232
c = 10. + 3j
d = 'hello Python'
a, b, c , d
```

Out[1]:

```
(10, 12.232, (10+3j), 'hello Python')
```

算数运算符

- 加 `+`
- 减 `-`
- 乘 `*`
- 除 `/`
- 取模 `%`
- 幂 `**`

In [2]:

```
a = 12 + 21
b = 12 - 21
c = 12 * 21
d = 12 / 21
e = 12 % 21
f = 12 ** 21
a, b, c, d, e, f
```

Out[2]:

```
(33, -9, 252, 0.5714285714285714, 12, 46005119909369701466112)
```

控制语句

- 控制语句: `if` , `else` , `elif`
- 判断
  - `>`
  - `<`
  - `==`
  - `>=`
  - `<=`
  - `!=`
- 多个判断语句可以使用: `and` , `or` , `not`

下面的情况会被任务是 **`False`**

- `False`
- `None`
- `0`
- `0.0`
- `''` 空字符串
- `[]`
- `{}`
- `()`
- `set()`

循环

常见两种模式:

```
for ... in ... :
    statement
```

```
while ... :
    statement
```

break

`break` 用来跳出循环

continue

`continue` 终止之后的运算，跳到循环开始下一个的循环

循环嵌套

函数

```
def FunctionName(param1, param2):
   ...
   return Outcome
```

默认参数

```
def FunctionName(param1, param2=12):
   ...
   return Outcome
```

复杂参数

```python
def fun(id, *args, **kwargs):
    print("id = ", id)
    print("args = ", args)
    print("kwargs = ", kwargs)
    print("---------------------------------")

if __name__ == '__main__':
    fun(1,2,3,4)
    fun(1, a=1,b=2,c=3)
    fun('a','b','c', a=1,b=2,c=3)

    a = (1,2,3,4)
    b = {'a':1,'b':2,'c':3}

    fun(*a, **b)
```

```
id =  1
args =  (2, 3, 4)
kwargs =  {}
---------------------------------
id =  1
args =  ()
kwargs =  {'a': 1, 'c': 3, 'b': 2}
---------------------------------
id =  a
args =  ('b', 'c')
kwargs =  {'a': 1, 'c': 3, 'b': 2}
---------------------------------
id =  1
args =  (2, 3, 4)
kwargs =  {'a': 1, 'c': 3, 'b': 2}
---------------------------------
```

总结

`*args` 是一个元组，而 `**kwargs` 是一个字典，同时，根据Python的要求， `**kwargs` 必须放在 `*args` 后面。

这一特性非常适合编写含有大量参数的程序，比如使用字典，在程序中可以先判断该值是否存在，然后进行下一步的操作。判断字典值是否存在可以使用 `dict.has_key('key')` 进行判断。

惯用法

```
x = kwargs.pop('x', 0.5)
y = kwargs.pop('y', 0.98)

if ('horizontalalignment' not in kwargs) and ('ha' not in kwargs):
    kwargs['horizontalalignment'] = 'center'
```

Python 容器

Python 容器指的是以下四类数据结构。

- 列表 list
- 元组 turple
- 字典 dict
- 集合 set

列表和元组

列表和元组均可理解为数组,而且数组值的类型可以任意,或者不一致。二者的主要区别在于列表是**可变**的,元组是**不可变**的。

In [3]:

```
# 列表的创建

empty_list = []
empty_list = list()
```

In [4]:

```
# 列表的操作: list()

a = ('a1', 'a2', 3, 4, 'b12', [1,2,3])
b = list(a)
b
```

Out[4]:

```
['a1', 'a2', 3, 4, 'b12', [1, 2, 3]]
```

In [5]:

```
# 列表的操作： [offset]

b[0]
```

Out[5]:

```
'a1'
```

In [6]:

```
b[1:3]
```

Out[6]:

```
['a2', 3]
```

In [7]:

```
b[-1]
```

Out[7]:

```
[1, 2, 3]
```

In [8]:

```
# 列表的操作： 修改

b[3] = 'b3'
b
```

Out[8]:

```
['a1', 'a2', 3, 'b3', 'b12', [1, 2, 3]]
```

In [9]:

```
# 列表的操作： 切片 Slice
# [start:end:step]

b[1:6:2]
```

Out[9]:

```
['a2', 'b3', [1, 2, 3]]
```

In [10]:

```
b[::-2]
```

Out[10]:

```
[[1, 2, 3], 'b3', 'a2']
```

In [11]:

```
# 列表的操作： 尾部添加元素

b.append('append1')
b
```

Out[11]:

```
['a1', 'a2', 3, 'b3', 'b12', [1, 2, 3], 'append1']
```

In [12]:

```
# 列表的操作： 合并 extend, +=

b2 = [1,2,3]
b.extend(b2)
b
```

Out[12]:

```
['a1', 'a2', 3, 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3]
```

In [13]:

```
b += b2
b
```

Out[13]:

```
['a1', 'a2', 3, 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2, 3]
```

In [14]:

```
# 列表的操作：指定位置添加 insert

print(b)
b.insert(3, 'python3')
print(b)
```

```
['a1', 'a2', 3, 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2, 3]
['a1', 'a2', 3, 'python3', 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2,
 3]
```

In [15]:

```
# 列表的操作： 指定位置删除 del

del b[3]
print(b)
```

```
['a1', 'a2', 3, 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2, 3]
```

In [16]:

```
# 列表的操作:  删除具有特定值的元素

print(b)
b.remove(3)
print(b)
```

```
['a1', 'a2', 3, 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2, 3]
['a1', 'a2', 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2, 3]
```

In [17]:

```
# 列表的操作:  获取并删除 pop

print(b)
pb = b.pop(1)
print(pb)
print(b)
```

```
['a1', 'a2', 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2, 3]
a2
['a1', 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2, 3]
```

In [18]:

```
# 列表的操作:  查询元素的位置 index

print(b)
print( b.index('append1') )
```

```
['a1', 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2, 3]
4
```

In [19]:

```
# 列表的操作:  判断是否在列表中 in

print(b)
print( 'append1' in b )
```

```
['a1', 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2, 3]
True
```

In [20]:

```
# 列表的操作:  值出现的次数

print(b)
print( b.count(2) )
```

```
['a1', 'b3', 'b12', [1, 2, 3], 'append1', 1, 2, 3, 1, 2, 3]
2
```

In [21]:

```
# 列表的操作： 转化为字符串

c = ['1', '2', '3', '4', '5', '6']
print(c)

','.join(c)
```

```
['1', '2', '3', '4', '5', '6']
```

Out[21]:

```
'1,2,3,4,5,6'
```

In [22]:

```
# 列表的操作： 排序

c = ['includes', 'several', 'open', 'source', 'development', 'environments', 'such',
'as']

# 副本排序 sorted

sorted_c = sorted(c)

print(c)
print(sorted_c)
```

```
['includes', 'several', 'open', 'source', 'development', 'environments', 'su
ch', 'as']
['as', 'development', 'environments', 'includes', 'open', 'several', 'sourc
e', 'such']
```

In [23]:

```
# 原地排序 sort

c.sort()

print(c)
```

```
['as', 'development', 'environments', 'includes', 'open', 'several', 'sourc
e', 'such']
```

In [24]:

```
# 反序

c.sort(reverse=True)
print(c)
```

```
['such', 'source', 'several', 'open', 'includes', 'environments', 'developme
nt', 'as']
```

In [25]:

```
# 列表的操作： 列表长度

print(len(c))
```

```
8
```

In [26]:

```
# 列表的操作： 赋值， 复制

# 使用 = 进行赋值
# 使用 copy 进行复制

a = [1, 2, 3]
b = a
print(b)
a[1] = 1234
print("a", a)
print("b", b)
```

```
[1, 2, 3]
a [1, 1234, 3]
b [1, 1234, 3]
```

In [27]:

```
c = a.copy()
print(c)
```

```
[1, 1234, 3]
```

字典 dict

`K-V`

In [28]:

```
# 字典的操作： 创建 {}

empty_dict = {}
kv_dict = {
    "key": "this is a key",
    "value": 1234,
}

print(empty_dict)
print(kv_dict)
```

```
{}
{'key': 'this is a key', 'value': 1234}
```

In [29]:

```
# 字典的操作： 转化 dict()

s = [['a', 'aa'], ['b', 'bb']]
d = dict(s)
print(d)
```

```
{'a': 'aa', 'b': 'bb'}
```

In [30]:

```
# 字典的操作： 修改元素 [key]

kv_dict['key'] = 'kkkkkk'
print(kv_dict)
```

```
{'key': 'kkkkkk', 'value': 1234}
```

In [31]:

```
# 字典的操作：合并字典 update

other_dict = {
    "other" : 987654,
}

kv_dict.update(other_dict)
print(kv_dict)
```

```
{'key': 'kkkkkk', 'other': 987654, 'value': 1234}
```

In [32]:

```
# 字典的操作：删除指定键

del kv_dict['other']
print(kv_dict)
```

```
{'key': 'kkkkkk', 'value': 1234}
```

In [33]:

```
# 字典的操作：删除所有元素 clear

kv_dict.clear()
print(kv_dict)
```

```
{}
```

In [34]:

```
# 字典的操作：判断是否在字典中 in

kv_dict = {
    "key": "this is a key",
    "value": 1234,
}

print('value1' in kv_dict)
```

```
False
```

In [35]:

```
# 字典的操作：获取所有键 keys()

print(kv_dict.keys())

print(list(kv_dict.keys()))
```

```
dict_keys(['key', 'value'])
['key', 'value']
```

In [36]:

```
# 字典的操作：获取所有值 values()


print(list(kv_dict.values()))
```

```
['this is a key', 1234]
```

In [37]:

```
# 字典的操作：获取所有键值对 items


print(list(kv_dict.items()))
```

```
[('key', 'this is a key'), ('value', 1234)]
```

集合 set

`set` 只有键，键不允许重复，支持集合的运算

In [38]:

```
# 集合的操作：创建 {} 或者 set()

odd = {1, 3, 5, 7, 9}
even = {0, 2, 4, 6, 8}

samset = set('letters')
print(samset)
```

```
{'s', 'l', 't', 'r', 'e'}
```

In [39]:

```
# 字典的操作：测试值是否存在 in

print(2 in odd)
```

```
False
```

In [40]:

```
# 字典的操作: 交集 &

odd & even
```

Out[40]:

```
set()
```

In [41]:

```
# 字典的操作: 并集 |

odd | even
```

Out[41]:

```
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

In [42]:

```
# 字典的操作: 差集 - 或者 difference

odd - even
odd.difference(even)
```

Out[42]:

```
{1, 3, 5, 7, 9}
```

In [43]:

```
# 字典的操作: 异或 ^ (去掉仅在两个集合中出现一次)

a = {1, 2, 3}
b = {3, 4, 5}

a^b
```

Out[43]:

```
{1, 2, 4, 5}
```

In [44]:

```
# 字典的操作: 是否为子集

a = {3}
b = {3, 4, 5}

a <= b

# 真子集 <

a < b
```

Out[44]:

```
True
```

In [45]:

```
# 字典的操作：超集

a >= b

a > b
```

Out[45]:

False

推导式

Python的推导式是一个具有Python风格的创建数据结构的方式，可以加速迭代，以少量的代码生成复杂的数据结构

列表推导式

`[expression for item in iterable if condition]`

In [46]:

```
# 生成偶数

a = [n for n in range(1,10) if n % 2 == 0]
print(a)
```

```
[2, 4, 6, 8]
```

In [47]:

```
# 生成数组

a = [(x, y) for x in range(1, 4) for y in range (1,5)]
print(a)
```

```
[(1, 1), (1, 2), (1, 3), (1, 4), (2, 1), (2, 2), (2, 3), (2, 4), (3, 1), (3,
 2), (3, 3), (3, 4)]
```

字典推导式

`[key:val for expression in iterable if condition]`

In [48]:

```
#

a = {k:v for k in range(1, 4) for v in range (6,9)}
print(a)
```

```
{1: 8, 2: 8, 3: 8}
```

Python 代码风格(PEP8)

Python **E**nhancement **P**roposal (**PEP**) 8 – *Style Guide for Python Code*

Python 增强提案定义了一系列的规范,PEP8 是代码风格的规范,可以 参考: http://pep8.org (http://pep8.org)。

- 代码布局
- 字符串引用
- 表达式和语句中的空白
- 注释
- 版本
- 命名

代码布局

缩进

- 每级缩进使用4个空格
- 参数尽量对齐

```python
# Aligned with opening delimiter.
foo = long_function_name(var_one, var_two,
                         var_three, var_four)

# More indentation included to distinguish this from the rest.
def long_function_name(
        var_one, var_two, var_three,
        var_four):
    print(var_one)

# Hanging indents should add a level.
foo = long_function_name(
    var_one, var_two,
    var_three, var_four)
```

```python
# Arguments on first line forbidden when not using vertical alignment.
foo = long_function_name(var_one, var_two,
    var_three, var_four)

# Further indentation required as indentation is not distinguishable.
def long_function_name(
    var_one, var_two, var_three,
    var_four):
    print(var_one)
```

- 长 `if` 语句

```python
# No extra indentation.
if (this_is_one_thing and
    that_is_another_thing):
    do_something()

# Add a comment, which will provide some distinction in editors
# supporting syntax highlighting.
if (this_is_one_thing and
    that_is_another_thing):
    # Since both conditions are true, we can frobnicate.
    do_something()

# Add some extra indentation on the conditional continuation line.
if (this_is_one_thing
        and that_is_another_thing):
    do_something()
```

- 长数组、元组

```python
my_list = [
    1, 2, 3,
    4, 5, 6,
    ]
result = some_function_that_takes_arguments(
    'a', 'b', 'c',
    'd', 'e', 'f',
    )
```

```python
my_list = [
    1, 2, 3,
    4, 5, 6,
]
result = some_function_that_takes_arguments(
    'a', 'b', 'c',
    'd', 'e', 'f',
)
```

Tab还是空格?

- 推荐使用空格进行缩进
- Python 2可以混用，但不推荐，可以使用 –t 选项进行检查
- Python 3不允许混用

每行最大长度

- 每行长度限制在 79 个字符以内
- 一个推荐的换行方式是使用反斜杠
- 逻辑运算符附近的换行最好在运算符之后

空行

- 使用2个空行来分隔最高级的函数（function）和类（class）定义
- 使用1个空行来分隔类中的方法（method）定义

源文件编码

- Python核心发行版中的代码一支使用UTF-8（Python2使用ASCII）
- 几种常见的编码声明方式

```
# -*- coding: utf-8 -*-
```

imports

- imports应该分行写
- imports应该写在代码文件的开头，位于模块注释和文档字符串之后，模块全局变量和常量声明之前
- imports顺序，不同组之间使用空格隔开：
    - 标准库
    - 相关第三方
    - 本地库
- 推荐使用绝对（absolue）imports，处理复杂包布局时，可以使用相对imports
- 避免使用通配符imports，比如 `from <module> import *`

```
import mypkg.sibling

from subprocess import Popen, PIPE
from mypkg.sibling import example

from . import sibling
from .sibling import example
```

字符串引用

- 在Python的世界，单引号和双引号是一样的，但不推荐混用，最好选择一种规则坚持使用。
- 三引号字符串，使用双引号字符，即 `“””` ，这样可以和 `PEP 257` 的规则保持一致

表达式中的空白

一些痛点

- 方括号、圆括号和花括号之后

```
spam(ham[1], {eggs: 2})       # Yes

spam( ham[ 1 ], { eggs: 2 } )     # No
```

- 逗号、分号和冒号之前

```
if x == 4: print x, y; x, y = y, x            # Yes

if x == 4 : print x , y ; x , y = y , x          # No
```

- 切片操作

```
# Yes
ham[1:9], ham[1:9:3], ham[:9:3], ham[1::3], ham[1:9:]
ham[lower:upper], ham[lower:upper:], ham[lower::step]
ham[lower+offset : upper+offset]
ham[: upper_fn(x) : step_fn(x)], ham[:: step_fn(x)]
ham[lower + offset : upper + offset]
```

```
# No
ham[lower + offset:upper + offset]
ham[1: 9], ham[1 :9], ham[1:9 :3]
ham[lower : : upper]
ham[ : upper]
```

- 函数调用

```
spam(1)      # Yes

spam (1)     # No
```

- 切片左中括号前

```
dct['key'] = lst[index]      # Yes

dct ['key'] = lst [index]    # No
```

- 赋值不需要进行对齐

```
x = 1                   # Yes
y = 2
long_variable = 3

x             = 1       # No
y             = 2
long_variable = 3
```

其他建议

- 二元运算符前后都使用一个空格
  - 赋值运算符 `=`
  - 增减运算符 `+=` , `-=`
  - 比较运算符 `==` , `<` , `>` , `!=` , `<>` , `<=` , `>=` , `in` , `not in` , `is` , `is not`
  - 布尔运算符 `and` , `or` , `not`

- 如果使用了优先级不同的运算符，优先级低的周围增加空白

```
i = i + 1                # Yes
submitted += 1
x = x*2 - 1
hypot2 = x*x + y*y
c = (a+b) * (a-b)

i=i+1                    # No
submitted +=1
x = x * 2 - 1
hypot2 = x * x + y * y
c = (a + b) * (a - b)
```

- 函数参数中的 `=` 前后可以不留白

```
# Yes
def complex(real, imag=0.0):
    return magic(r=real, i=imag)

# No
def complex(real, imag = 0.0):
    return magic(r = real, i = imag)
```

- 带注解的函数 `->` 前后有空白， `:` 后面一个空白

```python
# Yes
def munge(input: AnyStr): ...
def munge() -> AnyStr: ...

# No
def munge(input:AnyStr): ...
def munge()->PosInt: ...
```

- 复合语句（多条语句放在一行中）一般不鼓励使

pep8工具

`pip install pep8`

Usage:

`pep8 inputfile`

In [52]:

```
!cat pep8_test.py
```

```
def Hello():
    a=[12,34,56]
    for i in a:
        print(a)
```

In [53]:

```
!pep8 pep8_test.py
```

```
pep8_test.py:2:6: E225 missing whitespace around operator
pep8_test.py:2:10: E231 missing whitespace after ','
pep8_test.py:2:13: E231 missing whitespace after ','
pep8_test.py:4:1: E101 indentation contains mixed spaces and tabs
pep8_test.py:4:1: W191 indentation contains tabs
```

Python 软件包管理

pip

```
Usage:
  pip <command> [options]

Commands:
  install                     Install packages.
  download                    Download packages.
  uninstall                   Uninstall packages.
  freeze                      Output installed packages in requirements format.
  list                        List installed packages.
  show                        Show information about installed packages.
  search                      Search PyPI for packages.
  wheel                       Build wheels from your requirements.
  hash                        Compute hashes of package archives.
  completion                  A helper command used for command completion
  help                        Show help for commands.

General Options:
  -h, --help                  Show help.
  --isolated                  Run pip in an isolated mode, ignoring environment variab
les and user configuration.
  -v, --verbose               Give more output. Option is additive, and can be used up
 to 3 times.
  -V, --version               Show version and exit.
  -q, --quiet                 Give less output.
  --log <path>                Path to a verbose appending log.
  --proxy <proxy>             Specify a proxy in the form [user:passwd@]proxy.server:p
ort.
  --retries <retries>         Maximum number of retries each connection should attempt
 (default 5 times).
  --timeout <sec>             Set the socket timeout (default 15 seconds).
  --exists-action <action>    Default action when a path already exists: (s)witch, (i)
gnore, (w)ipe, (b)ackup.
  --trusted-host <hostname>   Mark this host as trusted, even though it does not have
 valid or any HTTPS.
  --cert <path>               Path to alternate CA bundle.
  --client-cert <path>        Path to SSL client certificate, a single file containing
 the private key and the
                              certificate in PEM format.
  --cache-dir <dir>           Store the cache data in <dir>.
  --no-cache-dir              Disable the cache.
  --disable-pip-version-check
                              Don't periodically check PyPI to determine whether a new
 version of pip is available for
                              download. Implied with --no-index.
```

安装

```
pip install pkg
```

升级

```
pip install -U pkg
```

搜索

```
pip search pkg
```

列出已安装的包

```
pip list
```

Python 2 和 Python 3

In [49]:

```
%%HTML
<iframe width=100% height=600 src="https://pythonclock.org/" ></iframe>
```

Python 2.7 will retire in...

3	9	14	9	4	34
Years	Months	Days	Hours	Minutes	Seconds

Enable Guido Mode Huh?

What's all this, then?

Python 2.7 will not be maintained past 2020. No official date has been given, so this clock counts down until April 12th, 2020, which will be roughly the time of the 2020 PyCon. I am hereby suggesting we *make* PyCon 2020 the official end-of-life date, and we throw a massive party to celebrate all that Python 2 has done for us. (If this sounds interesting to you, email pythonclockorg@gmail.com).

Python 2, thank you for your years of faithful service.

Python 3, your time is now.

How do I get started?

If the code you care about is still on Python 2, that's totally understandable. Most of PyPI's popular packages now work on Python 2 and 3, and more are being added every day. To ease the transition, the official porting guide has advice

- Python 3 是升级版本,与 Python 2 部分不兼容
- Python 2 仍有大量的用户,还有不少库与 Python 3 不兼容
- Python 3 的性能比 Python 2 有很大提升

In [50]:

```
# Python2 和 Python3

from __future__ import (absolute_import, division,
                        print_function, unicode_literals)
```

In [51]:

```
# -*- coding: utf-8 -*-

from __future__ import (absolute_import, division,
                        print_function, unicode_literals)
```

学习资源

- Python https://www.codecademy.com/zh/learn/python (https://www.codecademy.com/zh/learn/python)
- 用Python玩转数据 http://www.icourse163.org/course/nju-1001571005 (http://www.icourse163.org/course/nju-1001571005)
- 疯狂的Python：快速入门精讲 http://study.163.com/course/introduction/302001.htm (http://study.163.com/course/introduction/302001.htm)

Python for Astronomy

Python 基础知识 :: 编程范式

何勃亮

中国科学院国家天文台 中国虚拟天文台 (China-VO)

编程范式

- 文件操作 （ `IO` ）
- 字符串

文件操作

读文件

- read()
- readline()
- readlines()

open

`open(file, mode='r', buffering=-1, encoding=None, errors=None, newline=None, closefd=True, opener=None)`

```
'r'    open for reading (default)
'w'    open for writing, truncating the file first
'x'    open for exclusive creation, failing if the file already exists
'a'    open for writing, appending to the end of the file if it exists
'b'    binary mode
't'    text mode (default)
'+'    open a disk file for updating (reading and writing)
'U'    universal newlines mode (deprecated)
```

In [10]:

```
# read()

f = open('data/sample.txt', 'rt')
dat = f.read()
f.close()
print(dat)
```

```
obsid|designation|obsdate|lmjd|planid|spid|fiberid|ra|dec|snru|snrg|snrr|snr
i|snrz|objtype|class|subclass|magtype|mag1|mag2|mag3|mag4|mag5|mag6|mag7|tso
urce|fibertype|tfrom|t_info|rv|z|z_err
101001|J220848.54-020324.3|2011-10-24|55859|F5902|1|1|332.2022740000|-2.0567
670000|2.23|10.69|17.99|23.07|13.93|Star|STAR|K1|ugriz|18.78|17.12|16.42|16.
15|15.97|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-23.06902964||0.00000297
101002|J220953.17-020506.0|2011-10-24|55859|F5902|1|2|332.4715760000|-2.0850
150000|2.00|5.52|14.19|20.30|14.05|Star|STAR|M0|ugriz|20.91|18.10|16.66|16.0
5|15.67|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||27.10000040||0.00017775
101008|J220928.49-015720.7|2011-10-24|55859|F5902|1|8|332.3687450000|-1.9557
710000|1.84|9.94|25.25|32.32|18.29|Star|STAR|G5|ugriz|18.25|16.64|15.97|15.7
7|15.64|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||25.03866609||0.00000287
101009|J220849.59-015207.1|2011-10-24|55859|F5902|1|9|332.2066650000|-1.8686
530000|1.86|9.13|18.80|25.28|14.18|Star|STAR|G0|ugriz|18.64|17.19|16.63|16.3
7|16.25|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-22.16965227||0.00000537
101016|J220923.69-020809.9|2011-10-24|55859|F5902|1|16|332.3487250000|-2.136
0960000|2.17|28.22|52.30|72.89|46.52|Star|STAR|K5|ugriz|18.64|16.21|15.23|1
4.85|14.62|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-6.63140917||0.00000130
101017|J220946.66-015526.5|2011-10-24|55859|F5902|1|17|332.4444170000|-1.924
0460000|2.60|16.56|29.63|38.19|22.15|Star|STAR|G0|ugriz|17.97|16.53|16.00|1
5.78|15.65|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-2.46129608||0.00000233
101020|J220853.37-015915.4|2011-10-24|55859|F5902|1|20|332.2223790000|-1.987
6260000|2.65|17.26|26.29|36.30|20.29|Star|STAR|F5|ugriz|17.01|15.98|15.51|1
5.35|15.27|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||10.84948906||0.00000751
101021|J220924.33-014833.5|2011-10-24|55859|F5902|1|21|332.3513810000|-1.809
3330000|6.05|34.57|53.87|62.42|37.85|Star|STAR|F5|ugriz|16.75|15.61|15.16|1
4.98|14.92|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-17.91859521||0.00000354
101023|J221001.52-020100.8|2011-10-24|55859|F5902|1|23|332.5063740000|-2.016
9000000|2.35|12.14|22.38|27.72|16.25|Star|STAR|F9|ugriz|18.46|16.97|16.39|1
6.18|16.12|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||52.65854525||0.00000227
```

In [11]:

```
# read(chunksize)

f = open('data/sample.txt')
dat = f.read(100)
f.close()
print(dat)
```

```
obsid|designation|obsdate|lmjd|planid|spid|fiberid|ra|dec|snru|snrg|snrr|snr
i|snrz|objtype|class|sub
```

In [12]:

```
# readline

f = open('data/sample.txt', 'rt')
while True:
    line = f.readline()
    if not line:
        break
    print(line)
f.close()
```

```
obsid|designation|obsdate|lmjd|planid|spid|fiberid|ra|dec|snru|snrg|snrr|snr
i|snrz|objtype|class|subclass|magtype|mag1|mag2|mag3|mag4|mag5|mag6|mag7|tso
urce|fibertype|tfrom|t_info|rv|z|z_err

101001|J220848.54-020324.3|2011-10-24|55859|F5902|1|1|332.2022740000|-2.0567
670000|2.23|10.69|17.99|23.07|13.93|Star|STAR|K1|ugriz|18.78|17.12|16.42|16.
15|15.97|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-23.06902964||0.00000297

101002|J220953.17-020506.0|2011-10-24|55859|F5902|1|2|332.4715760000|-2.0850
150000|2.00|5.52|14.19|20.30|14.05|Star|STAR|M0|ugriz|20.91|18.10|16.66|16.0
5|15.67|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||27.10000040||0.00017775

101008|J220928.49-015720.7|2011-10-24|55859|F5902|1|8|332.3687450000|-1.9557
710000|1.84|9.94|25.25|32.32|18.29|Star|STAR|G5|ugriz|18.25|16.64|15.97|15.7
7|15.64|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||25.03866609||0.00000287

101009|J220849.59-015207.1|2011-10-24|55859|F5902|1|9|332.2066650000|-1.8686
530000|1.86|9.13|18.80|25.28|14.18|Star|STAR|G0|ugriz|18.64|17.19|16.63|16.3
7|16.25|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-22.16965227||0.00000537

101016|J220923.69-020809.9|2011-10-24|55859|F5902|1|16|332.3487250000|-2.136
0960000|2.17|28.22|52.30|72.89|46.52|Star|STAR|K5|ugriz|18.64|16.21|15.23|1
4.85|14.62|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-6.63140917||0.00000130

101017|J220946.66-015526.5|2011-10-24|55859|F5902|1|17|332.4444170000|-1.924
0460000|2.60|16.56|29.63|38.19|22.15|Star|STAR|G0|ugriz|17.97|16.53|16.00|1
5.78|15.65|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-2.46129608||0.00000233

101020|J220853.37-015915.4|2011-10-24|55859|F5902|1|20|332.2223790000|-1.987
6260000|2.65|17.26|26.29|36.30|20.29|Star|STAR|F5|ugriz|17.01|15.98|15.51|1
5.35|15.27|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||10.84948906||0.00000751

101021|J220924.33-014833.5|2011-10-24|55859|F5902|1|21|332.3513810000|-1.809
3330000|6.05|34.57|53.87|62.42|37.85|Star|STAR|F5|ugriz|16.75|15.61|15.16|1
4.98|14.92|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-17.91859521||0.00000354

101023|J221001.52-020100.8|2011-10-24|55859|F5902|1|23|332.5063740000|-2.016
9000000|2.35|12.14|22.38|27.72|16.25|Star|STAR|F9|ugriz|18.46|16.97|16.39|1
6.18|16.12|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||52.65854525||0.00000227
```

In [13]:

```
# readline
# 去掉行尾 '\n'

f = open('data/sample.txt', 'rt')
while True:
    line = f.readline()
    if not line:
        break
    print(line.rstrip())
f.close()
```

```
obsid|designation|obsdate|lmjd|planid|spid|fiberid|ra|dec|snru|snrg|snrr|snr
i|snrz|objtype|class|subclass|magtype|mag1|mag2|mag3|mag4|mag5|mag6|mag7|tso
urce|fibertype|tfrom|t_info|rv|z|z_err
101001|J220848.54-020324.3|2011-10-24|55859|F5902|1|1|332.2022740000|-2.0567
670000|2.23|10.69|17.99|23.07|13.93|Star|STAR|K1|ugriz|18.78|17.12|16.42|16.
15|15.97|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-23.06902964||0.00000297
101002|J220953.17-020506.0|2011-10-24|55859|F5902|1|2|332.4715760000|-2.0850
150000|2.00|5.52|14.19|20.30|14.05|Star|STAR|M0|ugriz|20.91|18.10|16.66|16.0
5|15.67|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||27.10000040||0.00017775
101008|J220928.49-015720.7|2011-10-24|55859|F5902|1|8|332.3687450000|-1.9557
710000|1.84|9.94|25.25|32.32|18.29|Star|STAR|G5|ugriz|18.25|16.64|15.97|15.7
7|15.64|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||25.03866609||0.00000287
101009|J220849.59-015207.1|2011-10-24|55859|F5902|1|9|332.2066650000|-1.8686
530000|1.86|9.13|18.80|25.28|14.18|Star|STAR|G0|ugriz|18.64|17.19|16.63|16.3
7|16.25|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-22.16965227||0.00000537
101016|J220923.69-020809.9|2011-10-24|55859|F5902|1|16|332.3487250000|-2.136
0960000|2.17|28.22|52.30|72.89|46.52|Star|STAR|K5|ugriz|18.64|16.21|15.23|1
4.85|14.62|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-6.63140917||0.00000130
101017|J220946.66-015526.5|2011-10-24|55859|F5902|1|17|332.4444170000|-1.924
0460000|2.60|16.56|29.63|38.19|22.15|Star|STAR|G0|ugriz|17.97|16.53|16.00|1
5.78|15.65|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-2.46129608||0.00000233
101020|J220853.37-015915.4|2011-10-24|55859|F5902|1|20|332.2223790000|-1.987
6260000|2.65|17.26|26.29|36.30|20.29|Star|STAR|F5|ugriz|17.01|15.98|15.51|1
5.35|15.27|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||10.84948906||0.00000751
101021|J220924.33-014833.5|2011-10-24|55859|F5902|1|21|332.3513810000|-1.809
3330000|6.05|34.57|53.87|62.42|37.85|Star|STAR|F5|ugriz|16.75|15.61|15.16|1
4.98|14.92|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-17.91859521||0.00000354
101023|J221001.52-020100.8|2011-10-24|55859|F5902|1|23|332.5063740000|-2.016
9000000|2.35|12.14|22.38|27.72|16.25|Star|STAR|F9|ugriz|18.46|16.97|16.39|1
6.18|16.12|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||52.65854525||0.00000227
```

In [14]:

```
# 迭代器读取

f = open('data/sample.txt', 'rt')
for line in f:
    print(line.rstrip())
f.close()
```

```
obsid|designation|obsdate|lmjd|planid|spid|fiberid|ra|dec|snru|snrg|snrr|snr
i|snrz|objtype|class|subclass|magtype|mag1|mag2|mag3|mag4|mag5|mag6|mag7|tso
urce|fibertype|tfrom|t_info|rv|z|z_err
101001|J220848.54-020324.3|2011-10-24|55859|F5902|1|1|332.2022740000|-2.0567
670000|2.23|10.69|17.99|23.07|13.93|Star|STAR|K1|ugriz|18.78|17.12|16.42|16.
15|15.97|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-23.06902964||0.00000297
101002|J220953.17-020506.0|2011-10-24|55859|F5902|1|2|332.4715760000|-2.0850
150000|2.00|5.52|14.19|20.30|14.05|Star|STAR|M0|ugriz|20.91|18.10|16.66|16.0
5|15.67|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||27.10000040||0.00017775
101008|J220928.49-015720.7|2011-10-24|55859|F5902|1|8|332.3687450000|-1.9557
710000|1.84|9.94|25.25|32.32|18.29|Star|STAR|G5|ugriz|18.25|16.64|15.97|15.7
7|15.64|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||25.03866609||0.00000287
101009|J220849.59-015207.1|2011-10-24|55859|F5902|1|9|332.2066650000|-1.8686
530000|1.86|9.13|18.80|25.28|14.18|Star|STAR|G0|ugriz|18.64|17.19|16.63|16.3
7|16.25|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-22.16965227||0.00000537
101016|J220923.69-020809.9|2011-10-24|55859|F5902|1|16|332.3487250000|-2.136
0960000|2.17|28.22|52.30|72.89|46.52|Star|STAR|K5|ugriz|18.64|16.21|15.23|1
4.85|14.62|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-6.63140917||0.00000130
101017|J220946.66-015526.5|2011-10-24|55859|F5902|1|17|332.4444170000|-1.924
0460000|2.60|16.56|29.63|38.19|22.15|Star|STAR|G0|ugriz|17.97|16.53|16.00|1
5.78|15.65|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-2.46129608||0.00000233
101020|J220853.37-015915.4|2011-10-24|55859|F5902|1|20|332.2223790000|-1.987
6260000|2.65|17.26|26.29|36.30|20.29|Star|STAR|F5|ugriz|17.01|15.98|15.51|1
5.35|15.27|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||10.84948906||0.00000751
101021|J220924.33-014833.5|2011-10-24|55859|F5902|1|21|332.3513810000|-1.809
3330000|6.05|34.57|53.87|62.42|37.85|Star|STAR|F5|ugriz|16.75|15.61|15.16|1
4.98|14.92|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-17.91859521||0.00000354
101023|J221001.52-020100.8|2011-10-24|55859|F5902|1|23|332.5063740000|-2.016
9000000|2.35|12.14|22.38|27.72|16.25|Star|STAR|F9|ugriz|18.46|16.97|16.39|1
6.18|16.12|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||52.65854525||0.00000227
```

In [15]:

```
# readlines 读取所有行

f = open('data/sample.txt', 'rt')
lines = f.readlines()

for line in lines:
    print(line.rstrip())
f.close()
```

```
obsid|designation|obsdate|lmjd|planid|spid|fiberid|ra|dec|snru|snrg|snrr|snr
i|snrz|objtype|class|subclass|magtype|mag1|mag2|mag3|mag4|mag5|mag6|mag7|tso
urce|fibertype|tfrom|t_info|rv|z|z_err
101001|J220848.54-020324.3|2011-10-24|55859|F5902|1|1|332.2022740000|-2.0567
670000|2.23|10.69|17.99|23.07|13.93|Star|STAR|K1|ugriz|18.78|17.12|16.42|16.
15|15.97|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-23.06902964||0.00000297
101002|J220953.17-020506.0|2011-10-24|55859|F5902|1|2|332.4715760000|-2.0850
150000|2.00|5.52|14.19|20.30|14.05|Star|STAR|M0|ugriz|20.91|18.10|16.66|16.0
5|15.67|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||27.10000040||0.00017775
101008|J220928.49-015720.7|2011-10-24|55859|F5902|1|8|332.3687450000|-1.9557
710000|1.84|9.94|25.25|32.32|18.29|Star|STAR|G5|ugriz|18.25|16.64|15.97|15.7
7|15.64|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||25.03866609||0.00000287
101009|J220849.59-015207.1|2011-10-24|55859|F5902|1|9|332.2066650000|-1.8686
530000|1.86|9.13|18.80|25.28|14.18|Star|STAR|G0|ugriz|18.64|17.19|16.63|16.3
7|16.25|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-22.16965227||0.00000537
101016|J220923.69-020809.9|2011-10-24|55859|F5902|1|16|332.3487250000|-2.136
0960000|2.17|28.22|52.30|72.89|46.52|Star|STAR|K5|ugriz|18.64|16.21|15.23|1
4.85|14.62|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-6.63140917||0.00000130
101017|J220946.66-015526.5|2011-10-24|55859|F5902|1|17|332.4444170000|-1.924
0460000|2.60|16.56|29.63|38.19|22.15|Star|STAR|G0|ugriz|17.97|16.53|16.00|1
5.78|15.65|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-2.46129608||0.00000233
101020|J220853.37-015915.4|2011-10-24|55859|F5902|1|20|332.2223790000|-1.987
6260000|2.65|17.26|26.29|36.30|20.29|Star|STAR|F5|ugriz|17.01|15.98|15.51|1
5.35|15.27|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||10.84948906||0.00000751
101021|J220924.33-014833.5|2011-10-24|55859|F5902|1|21|332.3513810000|-1.809
3330000|6.05|34.57|53.87|62.42|37.85|Star|STAR|F5|ugriz|16.75|15.61|15.16|1
4.98|14.92|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-17.91859521||0.00000354
101023|J221001.52-020100.8|2011-10-24|55859|F5902|1|23|332.5063740000|-2.016
9000000|2.35|12.14|22.38|27.72|16.25|Star|STAR|F9|ugriz|18.46|16.97|16.39|1
6.18|16.12|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||52.65854525||0.00000227
```

写文件

`write()`

In [16]:

```
fo = open('data/test.txt', 'wt')
fo.write('something')
fo.close()
```

In [17]:

```
!cat data/test.txt
```

```
something
```

with 语句

`with` 语句可以自动进行文件的关闭，简化编程。

```
with ... as ...:
    statement
```

In [18]:

```
with open('data/sample.txt', 'rt') as fi:
    for line in fi:
        print(line.rstrip())
```

```
obsid|designation|obsdate|lmjd|planid|spid|fiberid|ra|dec|snru|snrg|snrr|snr
i|snrz|objtype|class|subclass|magtype|mag1|mag2|mag3|mag4|mag5|mag6|mag7|tso
urce|fibertype|tfrom|t_info|rv|z|z_err
101001|J220848.54-020324.3|2011-10-24|55859|F5902|1|1|332.2022740000|-2.0567
670000|2.23|10.69|17.99|23.07|13.93|Star|STAR|K1|ugriz|18.78|17.12|16.42|16.
15|15.97|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-23.06902964||0.00000297
101002|J220953.17-020506.0|2011-10-24|55859|F5902|1|2|332.4715760000|-2.0850
150000|2.00|5.52|14.19|20.30|14.05|Star|STAR|M0|ugriz|20.91|18.10|16.66|16.0
5|15.67|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||27.10000040||0.00017775
101008|J220928.49-015720.7|2011-10-24|55859|F5902|1|8|332.3687450000|-1.9557
710000|1.84|9.94|25.25|32.32|18.29|Star|STAR|G5|ugriz|18.25|16.64|15.97|15.7
7|15.64|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||25.03866609||0.00000287
101009|J220849.59-015207.1|2011-10-24|55859|F5902|1|9|332.2066650000|-1.8686
530000|1.86|9.13|18.80|25.28|14.18|Star|STAR|G0|ugriz|18.64|17.19|16.63|16.3
7|16.25|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-22.16965227||0.00000537
101016|J220923.69-020809.9|2011-10-24|55859|F5902|1|16|332.3487250000|-2.136
0960000|2.17|28.22|52.30|72.89|46.52|Star|STAR|K5|ugriz|18.64|16.21|15.23|1
4.85|14.62|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-6.63140917||0.00000130
101017|J220946.66-015526.5|2011-10-24|55859|F5902|1|17|332.4444170000|-1.924
0460000|2.60|16.56|29.63|38.19|22.15|Star|STAR|G0|ugriz|17.97|16.53|16.00|1
5.78|15.65|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-2.46129608||0.00000233
101020|J220853.37-015915.4|2011-10-24|55859|F5902|1|20|332.2223790000|-1.987
6260000|2.65|17.26|26.29|36.30|20.29|Star|STAR|F5|ugriz|17.01|15.98|15.51|1
5.35|15.27|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||10.84948906||0.00000751
101021|J220924.33-014833.5|2011-10-24|55859|F5902|1|21|332.3513810000|-1.809
3330000|6.05|34.57|53.87|62.42|37.85|Star|STAR|F5|ugriz|16.75|15.61|15.16|1
4.98|14.92|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-17.91859521||0.00000354
101023|J221001.52-020100.8|2011-10-24|55859|F5902|1|23|332.5063740000|-2.016
9000000|2.35|12.14|22.38|27.72|16.25|Star|STAR|F9|ugriz|18.46|16.97|16.39|1
6.18|16.12|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||52.65854525||0.00000227
```

结构化文件的读写

- csv
- fits

后面详细介绍

字符串

format

- `%`
- `str.format()`

尽量使用 `.format` 而不是 `%`

传统的 %

In [19]:

```
"The Value is %012.8f" % 23.253427
```

Out[19]:

```
'The Value is 023.25342700'
```

In [20]:

```
"RA is %12.8f, Dec is %12.8f" % (332.202274, -2.056767)
```

Out[20]:

```
'RA is 332.20227400, Dec is  -2.05676700'
```

In [22]:

```
coord = {'RA': 332.202274, 'Dec': -2.056767}
"RA is %(RA)12.8f, Dec is %(Dec)12.8f" % coord
```

Out[22]:

```
'RA is 332.20227400, Dec is  -2.05676700'
```

.format

`[[fill]align][sign][#][0][width][,][.precision][type]`

- `fill` 填充符, 除 `{` 和 `}` 以外的其他字符
- `align` 对齐方式
- `sign` 符号
- `width` 宽度
- `precision` 精度
- `type` 转换类型

align 对齐方式

- `<` 左
- `>` 右，默认
- `=` 仅对数值类型有效
- `^` 居中

sign 符号

- `+` 正数前加 `+`，负数前加 `-`
- `-` 正数前不加 `+`，负数前加 `-`，默认
- `空格` 正数前加空格，负数前加 `-`

type 转换类型

- `b` 二进制
- `c` 字符
- `d` 十进制，默认
- `o` 9进制
- `x` 16进制小写
- `X` 16进制大写
- `e` 默认精度是6
- `E` 大写
- `f` 浮点，默认精度6
- `F` nan -> NAN， inf-> INF
- `g`
- `G`

In [23]:

```
"RA is {0:12.8f}, Dec is {1:12.8f}".format(332.202274, -2.056767)
```

Out[23]:

```
'RA is 332.20227400, Dec is  -2.05676700'
```

In [27]:

```
"RA is {RA:12.8f}, Dec is {Dec:12.8f}".format(RA=332.202274, Dec=-2.056767)
```

Out[27]:

```
'RA is 332.20227400, Dec is  -2.05676700'
```

In [28]:

```
coord = {'RA': 332.202274, 'Dec': -2.056767}
"RA is {0[RA]:12.8f}, Dec is {0[Dec]:12.8f}".format(coord)
```

Out[28]:

```
'RA is 332.20227400, Dec is  -2.05676700'
```

In [30]:

```
"RA is {}, Dec is {}".format(332.202274, -2.056767)
```

Out[30]:

```
'RA is 332.202274, Dec is -2.056767'
```

In [41]:

```
# Sample

print("decimal   hex   char   {0:^40}".format("name"))
print("-------  ----- ------ {0}".format("-"*40))

for v in range(10018, 10028):
    print("{0:7}  {0:^5X}  {0:^3c}".format(v))

print("-------  ----- ------ {0}".format("-"*40))
```

```
decimal   hex   char                   name
-------  ----- ------ ----------------------------------------
  10018  2722    ✢
  10019  2723    ✣
  10020  2724    ✤
  10021  2725    ✥
  10022  2726    ✦
  10023  2727    ✧
  10024  2728
  10025  2729    ✩
  10026  272A    ✪
  10027  272B    ✫
-------  ----- ------ ----------------------------------------
```

In [ ]:

In [21]:

```
import numpy as np
```

Python for Astronomy

IPython & Jupyter

何勃亮

中国科学院国家天文台 中国虚拟天文台 (China-VO)

In [22]:

```
# 列出模块的函数，常量等

dir(np)
```

Out[22]:

```
['ALLOW_THREADS',
 'BUFSIZE',
 'CLIP',
 'ComplexWarning',
 'DataSource',
 'ERR_CALL',
 'ERR_DEFAULT',
 'ERR_IGNORE',
 'ERR_LOG',
 'ERR_PRINT',
 'ERR_RAISE',
 'ERR_WARN',
 'FLOATING_POINT_SUPPORT',
 'FPE_DIVIDEBYZERO',
 'FPE_INVALID',
 'FPE_OVERFLOW',
 'FPE_UNDERFLOW',
 'False_',
 'Inf',
 'Infinity',
 'MAXDIMS',
 'MAY_SHARE_BOUNDS',
 'MAY_SHARE_EXACT',
 'MachAr',
 'ModuleDeprecationWarning',
 'NAN',
 'NINF',
 'NZERO',
 'NaN',
 'PINF',
 'PZERO',
 'PackageLoader',
 'RAISE',
 'RankWarning',
 'SHIFT_DIVIDEBYZERO',
 'SHIFT_INVALID',
 'SHIFT_OVERFLOW',
 'SHIFT_UNDERFLOW',
 'ScalarType',
 'Tester',
 'TooHardError',
 'True_',
 'UFUNC_BUFSIZE_DEFAULT',
 'UFUNC_PYVALS_NAME',
 'VisibleDeprecationWarning',
 'WRAP',
 '_NoValue',
 '__NUMPY_SETUP__',
 '__all__',
 '__builtins__',
 '__cached__',
 '__config__',
 '__doc__',
 '__file__',
 '__git_revision__',
 '__loader__',
 '__name__',
 '__package__',
 '__path__',
 '__spec__',
 '__version__',
```

```
'_import_tools',
'_mat',
'abs',
'absolute',
'absolute_import',
'add',
'add_docstring',
'add_newdoc',
'add_newdoc_ufunc',
'add_newdocs',
'alen',
'all',
'allclose',
'alltrue',
'alterdot',
'amax',
'amin',
'angle',
'any',
'append',
'apply_along_axis',
'apply_over_axes',
'arange',
'arccos',
'arccosh',
'arcsin',
'arcsinh',
'arctan',
'arctan2',
'arctanh',
'argmax',
'argmin',
'argpartition',
'argsort',
'argwhere',
'around',
'array',
'array2string',
'array_equal',
'array_equiv',
'array_repr',
'array_split',
'array_str',
'asanyarray',
'asarray',
'asarray_chkfinite',
'ascontiguousarray',
'asfarray',
'asfortranarray',
'asmatrix',
'asscalar',
'atleast_1d',
'atleast_2d',
'atleast_3d',
'average',
'bartlett',
'base_repr',
'bench',
'binary_repr',
'bincount',
'bitwise_and',
```

```
'bitwise_not',
'bitwise_or',
'bitwise_xor',
'blackman',
'bmat',
'bool',
'bool8',
'bool_',
'broadcast',
'broadcast_arrays',
'broadcast_to',
'busday_count',
'busday_offset',
'busdaycalendar',
'byte',
'byte_bounds',
'bytes0',
'bytes_',
'c_',
'can_cast',
'cast',
'cbrt',
'cdouble',
'ceil',
'cfloat',
'char',
'character',
'chararray',
'choose',
'clip',
'clongdouble',
'clongfloat',
'column_stack',
'common_type',
'compare_chararrays',
'compat',
'complex',
'complex128',
'complex256',
'complex64',
'complex_',
'complexfloating',
'compress',
'concatenate',
'conj',
'conjugate',
'convolve',
'copy',
'copysign',
'copyto',
'core',
'corrcoef',
'correlate',
'cos',
'cosh',
'count_nonzero',
'cov',
'cross',
'csingle',
'ctypeslib',
'cumprod',
```

```
'cumproduct',
'cumsum',
'datetime64',
'datetime_as_string',
'datetime_data',
'deg2rad',
'degrees',
'delete',
'deprecate',
'deprecate_with_doc',
'diag',
'diag_indices',
'diag_indices_from',
'diagflat',
'diagonal',
'diff',
'digitize',
'disp',
'divide',
'division',
'dot',
'double',
'dsplit',
'dstack',
'dtype',
'e',
'ediff1d',
'einsum',
'emath',
'empty',
'empty_like',
'equal',
'errstate',
'euler_gamma',
'exp',
'exp2',
'expand_dims',
'expm1',
'extract',
'eye',
'fabs',
'fastCopyAndTranspose',
'fft',
'fill_diagonal',
'find_common_type',
'finfo',
'fix',
'flatiter',
'flatnonzero',
'flexible',
'fliplr',
'flipud',
'float',
'float128',
'float16',
'float32',
'float64',
'float_',
'floating',
'floor',
'floor_divide',
```

```
'fmax',
'fmin',
'fmod',
'format_parser',
'frexp',
'frombuffer',
'fromfile',
'fromfunction',
'fromiter',
'frompyfunc',
'fromregex',
'fromstring',
'full',
'full_like',
'fv',
'generic',
'genfromtxt',
'get_array_wrap',
'get_include',
'get_printoptions',
'getbufsize',
'geterr',
'geterrcall',
'geterrobj',
'gradient',
'greater',
'greater_equal',
'half',
'hamming',
'hanning',
'histogram',
'histogram2d',
'histogramdd',
'hsplit',
'hstack',
'hypot',
'i0',
'identity',
'iinfo',
'imag',
'in1d',
'index_exp',
'indices',
'inexact',
'inf',
'info',
'infty',
'inner',
'insert',
'int',
'int0',
'int16',
'int32',
'int64',
'int8',
'int_',
'int_asbuffer',
'intc',
'integer',
'interp',
'intersect1d',
```

```
'intp',
'invert',
'ipmt',
'irr',
'is_busday',
'isclose',
'iscomplex',
'iscomplexobj',
'isfinite',
'isfortran',
'isinf',
'isnan',
'isneginf',
'isposinf',
'isreal',
'isrealobj',
'isscalar',
'issctype',
'issubclass_',
'issubdtype',
'issubsctype',
'iterable',
'ix_',
'kaiser',
'kron',
'ldexp',
'left_shift',
'less',
'less_equal',
'lexsort',
'lib',
'linalg',
'linspace',
'little_endian',
'load',
'loads',
'loadtxt',
'log',
'log10',
'log1p',
'log2',
'logaddexp',
'logaddexp2',
'logical_and',
'logical_not',
'logical_or',
'logical_xor',
'logspace',
'long',
'longcomplex',
'longdouble',
'longfloat',
'longlong',
'lookfor',
'ma',
'mafromtxt',
'mask_indices',
'mat',
'math',
'matmul',
'matrix',
```

```
'matrixlib',
'max',
'maximum',
'maximum_sctype',
'may_share_memory',
'mean',
'median',
'memmap',
'meshgrid',
'mgrid',
'min',
'min_scalar_type',
'minimum',
'mintypecode',
'mirr',
'mod',
'modf',
'moveaxis',
'msort',
'multiply',
'nan',
'nan_to_num',
'nanargmax',
'nanargmin',
'nanmax',
'nanmean',
'nanmedian',
'nanmin',
'nanpercentile',
'nanprod',
'nanstd',
'nansum',
'nanvar',
'nbytes',
'ndarray',
'ndenumerate',
'ndfromtxt',
'ndim',
'ndindex',
'nditer',
'negative',
'nested_iters',
'newaxis',
'nextafter',
'nonzero',
'not_equal',
'nper',
'npv',
'numarray',
'number',
'obj2sctype',
'object',
'object0',
'object_',
'ogrid',
'oldnumeric',
'ones',
'ones_like',
'outer',
'packbits',
'pad',
```

```
'partition',
'percentile',
'pi',
'piecewise',
'pkgload',
'place',
'pmt',
'poly',
'poly1d',
'polyadd',
'polyder',
'polydiv',
'polyfit',
'polyint',
'polymul',
'polynomial',
'polysub',
'polyval',
'power',
'ppmt',
'print_function',
'prod',
'product',
'promote_types',
'ptp',
'put',
'putmask',
'pv',
'r_',
'rad2deg',
'radians',
'random',
'rank',
'rate',
'ravel',
'ravel_multi_index',
'real',
'real_if_close',
'rec',
'recarray',
'recfromcsv',
'recfromtxt',
'reciprocal',
'record',
'remainder',
'repeat',
'require',
'reshape',
'resize',
'restoredot',
'result_type',
'right_shift',
'rint',
'roll',
'rollaxis',
'roots',
'rot90',
'round',
'round_',
'row_stack',
's_',
```

```
'safe_eval',
'save',
'savetxt',
'savez',
'savez_compressed',
'sctype2char',
'sctypeDict',
'sctypeNA',
'sctypes',
'searchsorted',
'select',
'set_numeric_ops',
'set_printoptions',
'set_string_function',
'setbufsize',
'setdiff1d',
'seterr',
'seterrcall',
'seterrobj',
'setxor1d',
'shape',
'shares_memory',
'short',
'show_config',
'sign',
'signbit',
'signedinteger',
'sin',
'sinc',
'single',
'singlecomplex',
'sinh',
'size',
'sometrue',
'sort',
'sort_complex',
'source',
'spacing',
'split',
'sqrt',
'square',
'squeeze',
'stack',
'std',
'str',
'str0',
'str_',
'string_',
'subtract',
'sum',
'swapaxes',
'sys',
'take',
'tan',
'tanh',
'tensordot',
'test',
'testing',
'tile',
'timedelta64',
'trace',
```

```
 'transpose',
 'trapz',
 'tri',
 'tril',
 'tril_indices',
 'tril_indices_from',
 'trim_zeros',
 'triu',
 'triu_indices',
 'triu_indices_from',
 'true_divide',
 'trunc',
 'typeDict',
 'typeNA',
 'typecodes',
 'typename',
 'ubyte',
 'ufunc',
 'uint',
 'uint0',
 'uint16',
 'uint32',
 'uint64',
 'uint8',
 'uintc',
 'uintp',
 'ulonglong',
 'unicode',
 'unicode_',
 'union1d',
 'unique',
 'unpackbits',
 'unravel_index',
 'unsignedinteger',
 'unwrap',
 'ushort',
 'vander',
 'var',
 'vdot',
 'vectorize',
 'version',
 'void',
 'void0',
 'vsplit',
 'vstack',
 'warnings',
 'where',
 'who',
 'zeros',
 'zeros_like']
```

In [23]:

```
# 函数帮助文档

help(np.loadtxt)

# https://github.com/numpy/numpy/blob/master/numpy/lib/npyio.py#l709
```

```
Help on function loadtxt in module numpy.lib.npyio:

loadtxt(fname, dtype=<class 'float'>, comments='#', delimiter=None, converte
rs=None, skiprows=0, usecols=None, unpack=False, ndmin=0)
    Load data from a text file.

    Each row in the text file must have the same number of values.

    Parameters
    ----------
    fname : file or str
        File, filename, or generator to read.  If the filename extension is
        ``.gz`` or ``.bz2``, the file is first decompressed. Note that
        generators should return byte strings for Python 3k.
    dtype : data-type, optional
        Data-type of the resulting array; default: float.  If this is a
        structured data-type, the resulting array will be 1-dimensional, and
        each row will be interpreted as an element of the array.  In this
        case, the number of columns used must match the number of fields in
        the data-type.
    comments : str or sequence, optional
        The characters or list of characters used to indicate the start of a
        comment;
        default: '#'.
    delimiter : str, optional
        The string used to separate values.  By default, this is any
        whitespace.
    converters : dict, optional
        A dictionary mapping column number to a function that will convert
        that column to a float.  E.g., if column 0 is a date string:
        ``converters = {0: datestr2num}``.  Converters can also be used to
        provide a default value for missing data (but see also `genfromtxt
`):
        ``converters = {3: lambda s: float(s.strip() or 0)}``.  Default: Non
e.
    skiprows : int, optional
        Skip the first `skiprows` lines; default: 0.
    usecols : sequence, optional
        Which columns to read, with 0 being the first.  For example,
        ``usecols = (1,4,5)`` will extract the 2nd, 5th and 6th columns.
        The default, None, results in all columns being read.
    unpack : bool, optional
        If True, the returned array is transposed, so that arguments may be
        unpacked using ``x, y, z = loadtxt(...)``.  When used with a structu
red
        data-type, arrays are returned for each field.  Default is False.
    ndmin : int, optional
        The returned array will have at least `ndmin` dimensions.
        Otherwise mono-dimensional axes will be squeezed.
        Legal values: 0 (default), 1 or 2.

        .. versionadded:: 1.6.0

    Returns
    -------
    out : ndarray
        Data read from the text file.

    See Also
    --------
    load, fromstring, fromregex
```

```
    genfromtxt : Load data with missing values handled as specified.
    scipy.io.loadmat : reads MATLAB data files

    Notes
    -----
    This function aims to be a fast reader for simply formatted files.  The
    `genfromtxt` function provides more sophisticated handling of, e.g.,
    lines with missing values.

    .. versionadded:: 1.10.0

    The strings produced by the Python float.hex method can be used as
    input for floats.

    Examples
    --------
    >>> from io import StringIO   # StringIO behaves like a file object
    >>> c = StringIO("0 1\n2 3")
    >>> np.loadtxt(c)
    array([[ 0.,  1.],
           [ 2.,  3.]])

    >>> d = StringIO("M 21 72\nF 35 58")
    >>> np.loadtxt(d, dtype={'names': ('gender', 'age', 'weight'),
    ...                      'formats': ('S1', 'i4', 'f4')})
    array([('M', 21, 72.0), ('F', 35, 58.0)],
          dtype=[('gender', '|S1'), ('age', '<i4'), ('weight', '<f4')])

    >>> c = StringIO("1,0,2\n3,0,4")
    >>> x, y = np.loadtxt(c, delimiter=',', usecols=(0, 2), unpack=True)
    >>> x
    array([ 1.,  3.])
    >>> y
    array([ 2.,  4.])
```

In [24]:

```
data = {i: np.random.randn() for i in range(10)}
```

In [25]:

```
data
```

Out[25]:

```
{0: -0.9067812636484553,
 1: 0.1311190585286933,
 2: -0.8178045357588558,
 3: -0.5201475782939451,
 4: -1.4180843210736191,
 5: -0.72846928957921,
 6: 1.1734438280674648,
 7: -0.5670176955863584,
 8: -0.25666316028484326,
 9: 0.14257961949847245}
```

In [26]:

```
# 内省

data?
```

In [27]:

```
def func_hello():
    """
    This is a sample function

    Returns
    -------
    "hello" string
    """
    return "hello"
```

In [28]:

```
func_hello?
```

In [29]:

```
np.load
```

Out[29]:

```
<function numpy.lib.npyio.load>
```

<TAB>

%run

In [30]:

```
%run func_test.py
```

```
func_test
```

In [31]:

```
!ls
```

```
0.prerequisites.ipynb            3.1-astropy.ipynb
0.prerequisites.slides.html      3.2-astropy-io.ipynb
1.0-basic.ipynb                  Untitled.ipynb
1.0-basic.slides.html            Untitled1.ipynb
1.1-programming.ipynb            custom.css
1.1-programming.slides.html      data
2.0-ipython-jupyter.ipynb        downloads
2.0-ipython-jupyter.slides.html filename.png
2.0-numpy.ipynb                  filename200.png
2.1-scipy.ipynb                  func_test.py
2.2-matplotlib.ipynb             images
2.3-pandas.ipynb                 index.ipynb
3.0-sdss-images.ipynb            pep8_test.py
```

In [32]:

```
!cat func_test.py
```

```
def func_test():
    print("func_test")

ft = func_test()
```

In [33]:

```
ft
```

查看变量 %who %whos

In [34]:

```
%who
```

```
data     ft      func_hello      func_test       np
```

In [35]:

```
# %xdel 删除变量

%xdel data
```

In [36]:

```
who
```

```
ft       func_hello      func_test       np
```

In [37]:

```
%whos
```

```
Variable     Type        Data/Info
----------------------------------
ft           NoneType    None
func_hello   function    <function func_hello at 0x10ee7f158>
func_test    function    <function func_test at 0x10e2d0e18>
np           module      <module 'numpy' from '/us<...>kages/numpy/__init__.
py'>
```

%paste

In [ ]:

```

```

%time

In [38]:

```
%time func_hello()
```

```
CPU times: user 3 µs, sys: 0 ns, total: 3 µs
Wall time: 5.01 µs
```

Out[38]:

```
'hello'
```

%timeit

In [39]:

```
%timeit func_hello()
```

```
The slowest run took 18.54 times longer than the fastest. This could mean th
at an intermediate result is being cached.
10000000 loops, best of 3: 92.7 ns per loop
```

%reset

In [40]:

```
who
```

```
ft	 func_hello	 func_test	 np
```

In [41]:

```
%reset
```

```
Once deleted, variables cannot be recovered. Proceed (y/[n])? y
```

In [42]:

```
who
```

```
Interactive namespace is empty.
```

%hist

In [43]:

```
%hist
```

```
import numpy as np
# 列出模块的函数，常量等

dir(np)
data = {i: np.random.randn() for i in range(10)}
data
# 内省

data?
def func_hello():
    """
    This is a sample function

    Returns
    -------
    "hello" string
    """
    return "hello"
func_hello?
np.*load*?
%run func_test.py
%run func_test.py
!cat func_test.py
!ls
%who
%whos
%time func_hello()
who
%reset
who
%hist
%pwd
import numpy as np
# 列出模块的函数，常量等

dir(np)
# 函数帮助文档

help(np.loadtxt)

# https://github.com/numpy/numpy/blob/master/numpy/lib/npyio.py#l709
data = {i: np.random.randn() for i in range(10)}
data
# 内省

data?
def func_hello():
    """
    This is a sample function

    Returns
    -------
    "hello" string
    """
    return "hello"
func_hello?
np.load
%run func_test.py
!ls
!cat func_test.py
ft
```

```
%who
# %xdel 删除变量

%xdel data
who
%whos
%time func_hello()
%timeit func_hello()
who
%reset
who
%hist
```

In [44]:

```
%pwd
```

Out[44]:

```
'/Users/hebl/Desktop/Python-In-Astronomy/notebook'
```

In [ ]:

```

```

In [1]:

```
%matplotlib inline
import matplotlib.pyplot as plt
```

Python for Astronomy

NumPy

何勃亮

中国科学院国家天文台 中国虚拟天文台 (China-VO)

NumPy

`NumPy` 是科学计算的基础，定义了在科学计算中数据是如何存储的，如何访问的。

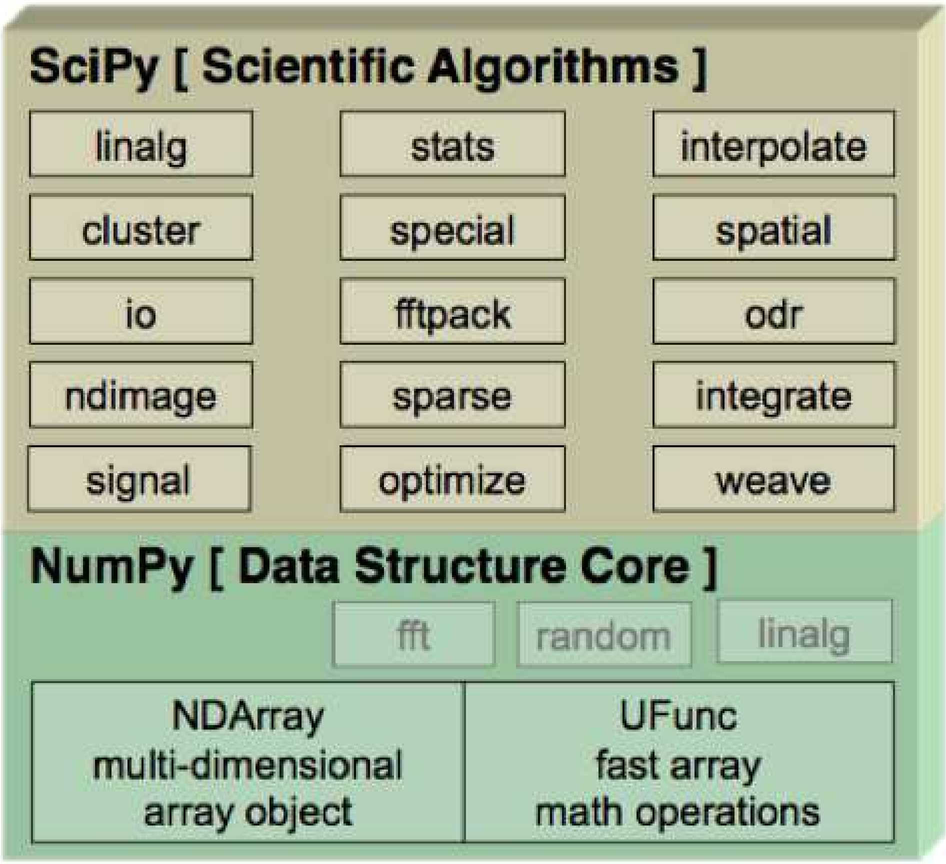
参考

- Quickstart tutorial (https://docs.scipy.org/doc/numpy-dev/user/quickstart.html)

In [2]:

```
import numpy as np
```

NumPy 和 SciPy



Numpy基础是数组对象 `ndarray`

- 数组的列类型是一致的

Numpy 数据类型

`np.dtype`

- bool
- inii
- int8
- int16
- int32
- int64
- uint8
- uint16
- uint32
- uint64
- float16
- float32
- float64, float
- complex128, complex

字符编码

- i 整数
- u 无符号整数
- f 浮点
- d 双精度浮点
- b bool
- D 复数
- S 字符串
- U unicode字符串
- V 空

In [3]:

```
np.dtype('i'), np.dtype(float)
```

Out[3]:

```
(dtype('int32'), dtype('float64'))
```

In [4]:

```
t = np.dtype('Float64')
t.char, t.type
```

Out[4]:

```
('d', numpy.float64)
```

创建 **np.array** 数组

In [5]:

```
a = np.array([1,2,3,4,5,6,7,8,9,10,11,12])
a
```

Out[5]:

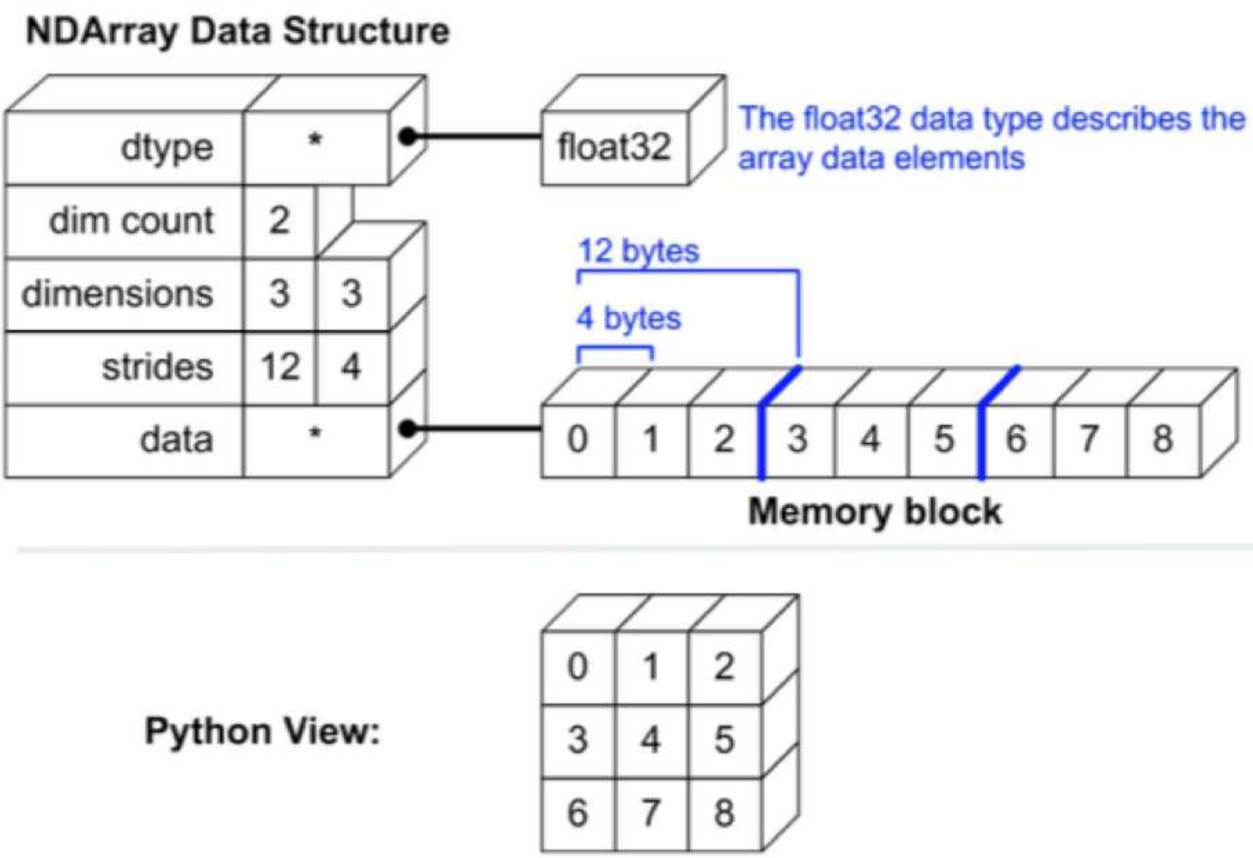
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

In [6]:

```
b = np.arange(10, dtype=np.int8)
b
```

Out[6]:

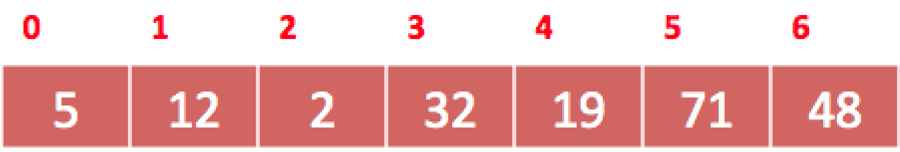
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9], dtype=int8)
```



Slice

一维

`var[lower:upper:step]`



In [7]:

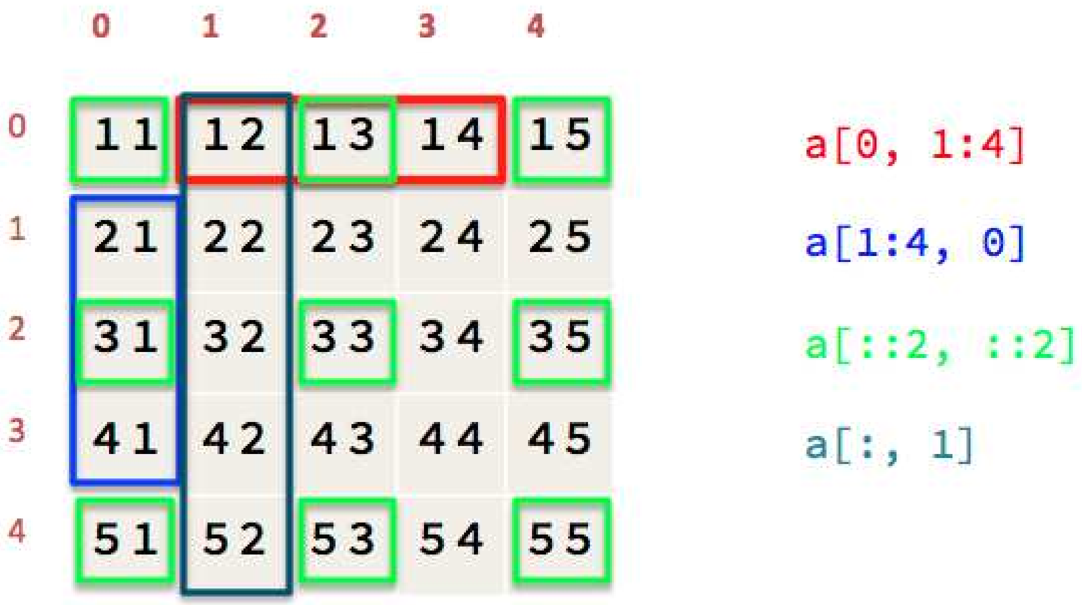
```
val = np.array([5, 12, 2, 32, 19, 71, 48])

val[1:5:2]
```

Out[7]:

```
array([12, 32])
```

二维

```
var[lower:upper:step, lower:upper:step]
```



In [8]:

```
# fancy indexing

a = np.arange(1,16).reshape(3,5)
a
```

Out[8]:

```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15]])
```

In [9]:

```
a[[1,2]]
```

Out[9]:

```
array([[ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15]])
```

In [10]:

```
a[[1,2], [3, 4]]
```

Out[10]:

```
array([ 9, 15])
```

In [11]:

```
a[a>5]
```

Out[11]:

```
array([ 6,  7,  8,  9, 10, 11, 12, 13, 14, 15])
```

In [12]:

```
# 生成函数

x = np.arange(-10,10,2)
x
```

Out[12]:

```
array([-10,  -8,  -6,  -4,  -2,   0,   2,   4,   6,   8])
```

In [13]:

```
x = np.linspace(-10, 10, 20)
x
```

Out[13]:

```
array([-10.        ,  -8.94736842,  -7.89473684,  -6.84210526,
        -5.78947368,  -4.73684211,  -3.68421053,  -2.63157895,
        -1.57894737,  -0.52631579,   0.52631579,   1.57894737,
         2.63157895,   3.68421053,   4.73684211,   5.78947368,
         6.84210526,   7.89473684,   8.94736842,  10.        ])
```

In [14]:

```
y = np.logspace(0, 10, 20, base=np.e)
y
```

Out[14]:

```
array([  1.00000000e+00,   1.69268460e+00,   2.86518116e+00,
         4.84984802e+00,   8.20926306e+00,   1.38956932e+01,
         2.35210258e+01,   3.98136782e+01,   6.73920000e+01,
         1.14073401e+02,   1.93090288e+02,   3.26840958e+02,
         5.53238656e+02,   9.36458553e+02,   1.58512897e+03,
         2.68312340e+03,   4.54168166e+03,   7.68763460e+03,
         1.30127407e+04,   2.20264658e+04])
```

In [15]:

```
# similar to meshgrid in MATLAB
x, y = np.mgrid[0:5, 0:5]
```

In [16]:

```
x
```

Out[16]:

```
array([[0, 0, 0, 0, 0],
       [1, 1, 1, 1, 1],
       [2, 2, 2, 2, 2],
       [3, 3, 3, 3, 3],
       [4, 4, 4, 4, 4]])
```

In [17]:

```
# uniform random
np.random.rand(5,5)
```

Out[17]:

```
array([[ 0.99708881,  0.32790373,  0.78421204,  0.46250811,  0.58650343],
       [ 0.12820361,  0.169632  ,  0.10251207,  0.69955435,  0.50615348],
       [ 0.82988333,  0.74002775,  0.62342001,  0.31515023,  0.9844216 ],
       [ 0.20368915,  0.90618349,  0.55240055,  0.44030397,  0.85502834],
       [ 0.89303322,  0.33488513,  0.701262  ,  0.41543455,  0.21345522]])
```

In [18]:

```
# 正态分布
np.random.randn(5,5)
```

Out[18]:

```
array([[ 0.6130893 ,  0.21470607,  0.07138132, -0.77000397,  0.26283988],
       [ 1.50457865, -0.66699852,  0.262513  , -1.5106911 ,  0.67808786],
       [-0.60650634,  0.88533886, -0.78460221, -0.291146  ,  1.83469553],
       [ 1.04451005,  0.34923151,  0.73092256, -0.17767853,  0.84140369],
       [ 0.23179844, -0.94431637, -1.4098484 ,  2.07534083,  0.86086853]])
```

np.array 操作

In [19]:

```
a.shape = (3,4)
a
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-19-f0af00c30b35> in <module>()
----> 1 a.shape = (3,4)
      2 a

ValueError: total size of new array must be unchanged
```

In [20]:

```
a.reshape(5,3)
```

Out[20]:

```
array([[ 1,  2,  3],
       [ 4,  5,  6],
       [ 7,  8,  9],
       [10, 11, 12],
       [13, 14, 15]])
```

In [21]:

```
# 展平

b = a.ravel()
```

In [22]:

```
a
```

Out[22]:

```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15]])
```

In [23]:

```
b
```

Out[23]:

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15])
```

In [24]:

```
b = a.flatten()
```

In [25]:

```
a
```

Out[25]:

```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15]])
```

In [26]:

```
b
```

Out[26]:

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15])
```

`ravel` 返回的是视图，而 `flatten` 返回的是重新分配内存的新结果

In [27]:

```
a
```

Out[27]:

```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15]])
```

In [28]:

```
a.transpose()
```

Out[28]:

```
array([[ 1,  6, 11],
       [ 2,  7, 12],
       [ 3,  8, 13],
       [ 4,  9, 14],
       [ 5, 10, 15]])
```

In [29]:

```
a
```

Out[29]:

```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15]])
```

In [30]:

```
a.T
```

Out[30]:

```
array([[ 1,  6, 11],
       [ 2,  7, 12],
       [ 3,  8, 13],
       [ 4,  9, 14],
       [ 5, 10, 15]])
```

In [31]:

```
a.resize((5,3))
```

In [32]:

```
a
```

Out[32]:

```
array([[ 1,  2,  3],
       [ 4,  5,  6],
       [ 7,  8,  9],
       [10, 11, 12],
       [13, 14, 15]])
```

与 `reshape` 不同，它就地更新结构

In [33]:

```
a.ndim
```

Out[33]:

```
2
```

In [34]:

```
a.size
```

Out[34]:

```
15
```

In [35]:

```
a.itemsize
```

Out[35]:

```
8
```

In [36]:

```
a.nbytes
```

Out[36]:

```
120
```

In [37]:

```
a.dtype
```

Out[37]:

```
dtype('int64')
```

In [38]:

```
a.T
```

Out[38]:

```
array([[ 1,  4,  7, 10, 13],
       [ 2,  5,  8, 11, 14],
       [ 3,  6,  9, 12, 15]])
```

In [39]:

```
a.tolist()
```

Out[39]:

```
[[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12], [13, 14, 15]]
```

In [40]:

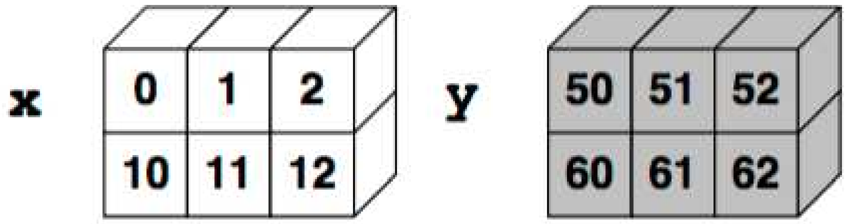
```
a.dot(a.T)
```

Out[40]:

```
array([[ 14,  32,  50,  68,  86],
       [ 32,  77, 122, 167, 212],
       [ 50, 122, 194, 266, 338],
       [ 68, 167, 266, 365, 464],
       [ 86, 212, 338, 464, 590]])
```

数组合并

`concatenate((a0,a1,...,aN),axis=0)`



In [41]:

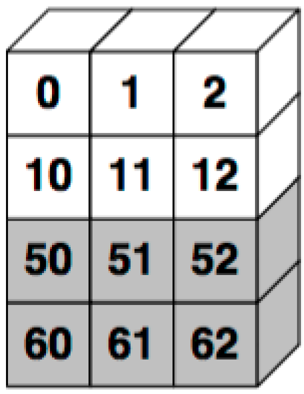
```
x = np.array([[0, 1, 2],[10, 11, 12]])
y = np.array([[50, 51, 52],[60, 61, 62]])
```

In [42]:

```
np.concatenate((x, y))
```

Out[42]:

```
array([[ 0,  1,  2],
       [10, 11, 12],
       [50, 51, 52],
       [60, 61, 62]])
```

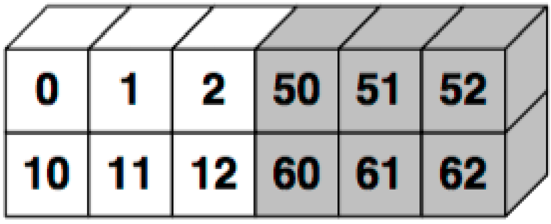


In [43]:

```
np.concatenate((x, y), 1)
```

Out[43]:

```
array([[ 0,  1,  2, 50, 51, 52],
       [10, 11, 12, 60, 61, 62]])
```

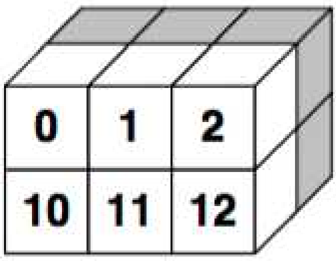


In [44]:

```
np.array((x,y))
```

Out[44]:

```
array([[[ 0,  1,  2],
        [10, 11, 12]],

       [[50, 51, 52],
        [60, 61, 62]]])
```



In [45]:

```
np.vstack((x,y))
```

Out[45]:

```
array([[ 0,  1,  2],
       [10, 11, 12],
       [50, 51, 52],
       [60, 61, 62]])
```

In [46]:

```
np.hstack((x,y))
```

Out[46]:

```
array([[ 0,  1,  2, 50, 51, 52],
       [10, 11, 12, 60, 61, 62]])
```

In [47]:

```
np.dstack((x,y))
```

Out[47]:

```
array([[[ 0, 50],
        [ 1, 51],
        [ 2, 52]],

       [[10, 60],
        [11, 61],
        [12, 62]]])
```

矩阵

在 `numpy` 中，矩阵是 `ndarray` 的子类。

矩阵函数

- mat
- matrix
- bmat 复合矩阵

In [48]:

```
A = np.mat('1 2 3; 4 5 6; 7 8 9')
A
```

Out[48]:

```
matrix([[1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]])
```

In [49]:

```
A = np.matrix('1 2 3; 4 5 6; 7 8 9')
A
```

Out[49]:

```
matrix([[1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]])
```

In [50]:

```
A = np.mat(np.arange(1,10).reshape(3,3))
A
```

Out[50]:

```
matrix([[1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]])
```

矩阵的计算

In [51]:

```
# 转置矩阵

A.T
```

Out[51]:

```
matrix([[1, 4, 7],
        [2, 5, 8],
        [3, 6, 9]])
```

In [52]:

```
# 逆矩阵

A.I
```

Out[52]:

```
matrix([[ -4.50359963e+15,   9.00719925e+15,  -4.50359963e+15],
        [  9.00719925e+15,  -1.80143985e+16,   9.00719925e+15],
        [ -4.50359963e+15,   9.00719925e+15,  -4.50359963e+15]])
```

In [53]:

```
# Hermitian
C = np.matrix([[1j, 2j], [3j, 4j]])
C
```

Out[53]:

```
matrix([[ 0.+1.j,  0.+2.j],
        [ 0.+3.j,  0.+4.j]])
```

In [54]:

```
C.H
```

Out[54]:

```
matrix([[ 0.-1.j,  0.-3.j],
        [ 0.-2.j,  0.-4.j]])
```

In [55]:

```
# 转化成一维
A.A1
```

Out[55]:

```
array([1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [56]:

```
np.linalg.det(A)
```

Out[56]:

```
6.6613381477509402e-16
```

In [57]:

```
# 单位矩阵

A = np.eye(3)
A
```

Out[57]:

```
array([[ 1.,  0.,  0.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.]])
```

In [58]:

```
B = A*10
B
```

Out[58]:

```
array([[ 10.,   0.,   0.],
       [  0.,  10.,   0.],
       [  0.,   0.,  10.]])
```

In [59]:

```
np.bmat("A B; B A")
```

Out[59]:

```
matrix([[  1.,   0.,   0.,  10.,   0.,   0.],
        [  0.,   1.,   0.,   0.,  10.,   0.],
        [  0.,   0.,   1.,   0.,   0.,  10.],
        [ 10.,   0.,   0.,   1.,   0.,   0.],
        [  0.,  10.,   0.,   0.,   1.,   0.],
        [  0.,   0.,  10.,   0.,   0.,   1.]])
```

In [60]:

```
# 零矩阵

A = np.zeros((3,3))
A
```

Out[60]:

```
array([[ 0.,  0.,  0.],
       [ 0.,  0.,  0.],
       [ 0.,  0.,  0.]])
```

In [61]:

```
A = np.mat(np.arange(1,10).reshape(3,3))
B = np.zeros_like(A)
B
```

Out[61]:

```
matrix([[0, 0, 0],
        [0, 0, 0],
        [0, 0, 0]])
```

In [62]:

```
# 对角阵

np.diag([1,2,3,4,5])
```

Out[62]:

```
array([[1, 0, 0, 0, 0],
       [0, 2, 0, 0, 0],
       [0, 0, 3, 0, 0],
       [0, 0, 0, 4, 0],
       [0, 0, 0, 0, 5]])
```

In [63]:

```
np.diag([1,2,3,4,5], k=1)
```

Out[63]:

```
array([[0, 1, 0, 0, 0, 0],
       [0, 0, 2, 0, 0, 0],
       [0, 0, 0, 3, 0, 0],
       [0, 0, 0, 0, 4, 0],
       [0, 0, 0, 0, 0, 5],
       [0, 0, 0, 0, 0, 0]])
```

矩阵计算

In [64]:

```
A = np.matrix('1 2 3; 4 5 6; 7 8 9')
```

In [65]:

```
A * 2
```

Out[65]:

```
matrix([[ 2,  4,  6],
        [ 8, 10, 12],
        [14, 16, 18]])
```

In [66]:

```
A + 2
```

Out[66]:

```
matrix([[ 3,  4,  5],
        [ 6,  7,  8],
        [ 9, 10, 11]])
```

In [67]:

```
A * A
```

Out[67]:

```
matrix([[ 30,  36,  42],
        [ 66,  81,  96],
        [102, 126, 150]])
```

In [68]:

```
# 求模

np.mod(A, 2)
```

Out[68]:

```
matrix([[1, 0, 1],
        [0, 1, 0],
        [1, 0, 1]])
```

In [69]:

```
A % 2
```

Out[69]:

```
matrix([[1, 0, 1],
        [0, 1, 0],
        [1, 0, 1]])
```

$A \times B$

A/B

$A^2 \times B$

In [70]:

```
A = np.array([1.0,6.0,2.0,5.0,8.0,9.0])
B = np.array([6.0,2.0,4.0,7.0,9.0,2.0])
A*B, A/B, A**2*B
```

Out[70]:

```
(array([  6.,  12.,   8.,  35.,  72.,  18.]),
 array([ 0.16666667,  3.        ,  0.5       ,  0.71428571,  0.88888889,
         4.5       ]),
 array([   6.,   72.,   16.,  175.,  576.,  162.]))
```

向量化函数

In [71]:

```
a = np.array([-3,-2,-1,0,1,2,3])

def Theta(x):
    if x >= 0:
        return 1
    else:
        return 0
```

In [72]:

```
Theta(a)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-72-06b326920517> in <module>()
----> 1 Theta(a)

<ipython-input-71-378beb789c60> in Theta(x)
      2 
      3 def Theta(x):
----> 4     if x >= 0:
      5         return 1
      6     else:

ValueError: The truth value of an array with more than one element is ambigu
ous. Use a.any() or a.all()
```

In [73]:

```
Theta_V = np.vectorize(Theta)

Theta_V(a)
```

Out[73]:

```
array([0, 0, 0, 1, 1, 1, 1])
```

多项式

$3x^2 + 2x - 1$

In [74]:

```
p = np.poly1d([3, 2, -1])
```

In [75]:

```
p(1)
```

Out[75]:

```
4
```

In [76]:

```
p.roots
```

Out[76]:

```
array([-1.        ,  0.33333333])
```

In [77]:

```
p.order
```

Out[77]:

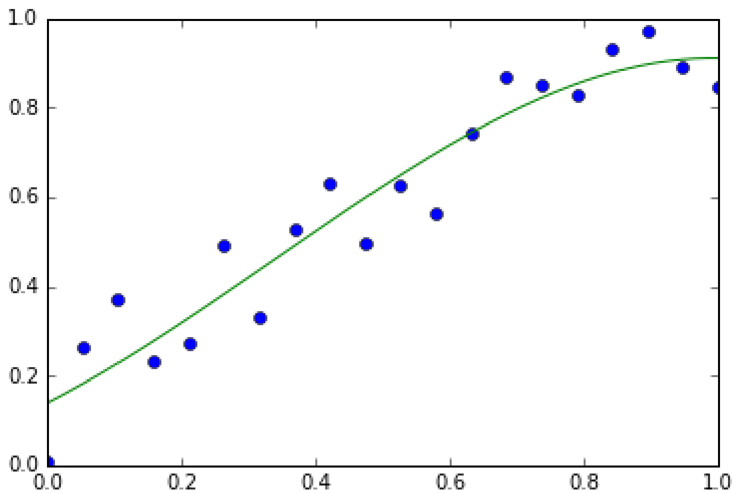
```
2
```

In [78]:

```
x = np.linspace(0, 1, 20)
y = np.sin(x) + 0.3*np.random.rand(20)
p = np.poly1d(np.polyfit(x, y, 3))

t = np.linspace(0,1,200)

plt.plot(x, y, 'o', t, p(t), '-')
```

Out[78]:

```
[<matplotlib.lines.Line2D at 0x106516898>,
 <matplotlib.lines.Line2D at 0x106516a58>]
```



数据文件读取

In [79]:

```
data = []

with open('data/mat.txt') as file:
    for line in file:
        fields = line.split()
        row_data = [float(x) for x in fields]

        data.append(row_data)

data = np.array(data)

data
```

Out[79]:

```
array([[ 0.84545347,  0.44725417,  0.47999929,  0.08579657],
       [ 0.00206587,  0.39216908,  0.4311762 ,  0.11400269],
       [ 0.47211792,  0.56315336,  0.36709996,  0.02245953],
       [ 0.02446823,  0.51626722,  0.90963311,  0.16006042],
       [ 0.82464247,  0.283328  ,  0.84850368,  0.3052179 ],
       [ 0.78429442,  0.18458249,  0.78490952,  0.65116394],
       [ 0.013527  ,  0.48211147,  0.28053254,  0.01123305],
       [ 0.03692758,  0.33033247,  0.93326603,  0.57492495],
       [ 0.53283285,  0.85009472,  0.7293702 ,  0.37974827],
       [ 0.54816185,  0.78522169,  0.3731202 ,  0.05622311]])
```

In [80]:

```
data = np.loadtxt("data/mat.txt")
data
```

Out[80]:

```
array([[ 0.84545347,  0.44725417,  0.47999929,  0.08579657],
       [ 0.00206587,  0.39216908,  0.4311762 ,  0.11400269],
       [ 0.47211792,  0.56315336,  0.36709996,  0.02245953],
       [ 0.02446823,  0.51626722,  0.90963311,  0.16006042],
       [ 0.82464247,  0.283328  ,  0.84850368,  0.3052179 ],
       [ 0.78429442,  0.18458249,  0.78490952,  0.65116394],
       [ 0.013527  ,  0.48211147,  0.28053254,  0.01123305],
       [ 0.03692758,  0.33033247,  0.93326603,  0.57492495],
       [ 0.53283285,  0.85009472,  0.7293702 ,  0.37974827],
       [ 0.54816185,  0.78522169,  0.3731202 ,  0.05622311]])
```

In [81]:

```
M = np.random.rand(10, 4)
np.savetxt("data/mat.txt", M)
```

In [82]:

```
M
```

Out[82]:

```
array([[ 0.56198339,  0.16553871,  0.57310649,  0.58528156],
       [ 0.363642  ,  0.73304361,  0.85445552,  0.36342131],
       [ 0.77505737,  0.63239828,  0.41010258,  0.87824707],
       [ 0.17446411,  0.52690052,  0.64765012,  0.82850736],
       [ 0.7261763 ,  0.04327556,  0.55110713,  0.21871242],
       [ 0.70179891,  0.25187961,  0.55744213,  0.76957906],
       [ 0.75812029,  0.99244128,  0.67686674,  0.68061882],
       [ 0.91686604,  0.59147   ,  0.36447999,  0.03570074],
       [ 0.79477932,  0.77501612,  0.08069673,  0.70004227],
       [ 0.27068581,  0.05866099,  0.06355585,  0.11082832]])
```

In [ ]:

```

```

In [83]:

```
np.savetxt("data/mat.csv", M, fmt="%.3f")
```

In [84]:

```
!cat data/mat.csv
```

```
0.562 0.166 0.573 0.585
0.364 0.733 0.854 0.363
0.775 0.632 0.410 0.878
0.174 0.527 0.648 0.829
0.726 0.043 0.551 0.219
0.702 0.252 0.557 0.770
0.758 0.992 0.677 0.681
0.917 0.591 0.364 0.036
0.795 0.775 0.081 0.700
0.271 0.059 0.064 0.111
```

In [85]:

```
dr1 = np.loadtxt('data/sample.txt')
dr1
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-85-06f8d6d7f309> in <module>()
----> 1 dr1 = np.loadtxt('data/sample.txt')
      2 dr1

/usr/local/lib/python3.5/site-packages/numpy/lib/npyio.py in loadtxt(fname,
 dtype, comments, delimiter, converters, skiprows, usecols, unpack, ndmin)
    928
    929             # Convert each value according to its column and store
--> 930             items = [conv(val) for (conv, val) in zip(converters, va
ls)]
    931             # Then pack it according to the dtype's nesting
    932             items = pack_items(items, packing)

/usr/local/lib/python3.5/site-packages/numpy/lib/npyio.py in <listcomp>(.0)
    928
    929             # Convert each value according to its column and store
--> 930             items = [conv(val) for (conv, val) in zip(converters, va
ls)]
    931             # Then pack it according to the dtype's nesting
    932             items = pack_items(items, packing)

/usr/local/lib/python3.5/site-packages/numpy/lib/npyio.py in floatconv(x)
    657         if b'0x' in x:
    658             return float.fromhex(asstr(x))
--> 659         return float(x)
    660
    661     typ = dtype.type

ValueError: could not convert string to float: b'obsid|designation|obsdate|l
mjd|planid|spid|fiberid|ra|dec|snru|snrg|snrr|snri|snrz|objtype|class|subcla
ss|magtype|mag1|mag2|mag3|mag4|mag5|mag6|mag7|tsource|fibertype|tfrom|t_info
|rv|z|z_err'
```

In [86]:

```
help(np.loadtxt)
```

```
Help on function loadtxt in module numpy.lib.npyio:

loadtxt(fname, dtype=<class 'float'>, comments='#', delimiter=None, converte
rs=None, skiprows=0, usecols=None, unpack=False, ndmin=0)
    Load data from a text file.

    Each row in the text file must have the same number of values.

    Parameters
    ----------
    fname : file or str
        File, filename, or generator to read.  If the filename extension is
        ``.gz`` or ``.bz2``, the file is first decompressed. Note that
        generators should return byte strings for Python 3k.
    dtype : data-type, optional
        Data-type of the resulting array; default: float.  If this is a
        structured data-type, the resulting array will be 1-dimensional, and
        each row will be interpreted as an element of the array.  In this
        case, the number of columns used must match the number of fields in
        the data-type.
    comments : str or sequence, optional
        The characters or list of characters used to indicate the start of a
        comment;
        default: '#'.
    delimiter : str, optional
        The string used to separate values.  By default, this is any
        whitespace.
    converters : dict, optional
        A dictionary mapping column number to a function that will convert
        that column to a float.  E.g., if column 0 is a date string:
        ``converters = {0: datestr2num}``.  Converters can also be used to
        provide a default value for missing data (but see also `genfromtxt
`):
        ``converters = {3: lambda s: float(s.strip() or 0)}``.  Default: Non
e.
    skiprows : int, optional
        Skip the first `skiprows` lines; default: 0.
    usecols : sequence, optional
        Which columns to read, with 0 being the first.  For example,
        ``usecols = (1,4,5)`` will extract the 2nd, 5th and 6th columns.
        The default, None, results in all columns being read.
    unpack : bool, optional
        If True, the returned array is transposed, so that arguments may be
        unpacked using ``x, y, z = loadtxt(...)``.  When used with a structu
red
        data-type, arrays are returned for each field.  Default is False.
    ndmin : int, optional
        The returned array will have at least `ndmin` dimensions.
        Otherwise mono-dimensional axes will be squeezed.
        Legal values: 0 (default), 1 or 2.

        .. versionadded:: 1.6.0

    Returns
    -------
    out : ndarray
        Data read from the text file.

    See Also
    --------
    load, fromstring, fromregex
```

```
genfromtxt : Load data with missing values handled as specified.
scipy.io.loadmat : reads MATLAB data files

Notes
-----
This function aims to be a fast reader for simply formatted files.  The
`genfromtxt` function provides more sophisticated handling of, e.g.,
lines with missing values.

.. versionadded:: 1.10.0

The strings produced by the Python float.hex method can be used as
input for floats.

Examples
--------
>>> from io import StringIO   # StringIO behaves like a file object
>>> c = StringIO("0 1\n2 3")
>>> np.loadtxt(c)
array([[ 0.,  1.],
       [ 2.,  3.]])

>>> d = StringIO("M 21 72\nF 35 58")
>>> np.loadtxt(d, dtype={'names': ('gender', 'age', 'weight'),
...                      'formats': ('S1', 'i4', 'f4')})
array([('M', 21, 72.0), ('F', 35, 58.0)],
      dtype=[('gender', '|S1'), ('age', '<i4'), ('weight', '<f4')])

>>> c = StringIO("1,0,2\n3,0,4")
>>> x, y = np.loadtxt(c, delimiter=',', usecols=(0, 2), unpack=True)
>>> x
array([ 1.,  3.])
>>> y
array([ 2.,  4.])
```

In [87]:

```
!head data/sample.txt
```

```
obsid|designation|obsdate|lmjd|planid|spid|fiberid|ra|dec|snru|snrg|snrr|snr
i|snrz|objtype|class|subclass|magtype|mag1|mag2|mag3|mag4|mag5|mag6|mag7|tso
urce|fibertype|tfrom|t_info|rv|z|z_err
101001|J220848.54-020324.3|2011-10-24|55859|F5902|1|1|332.2022740000|-2.0567
670000|2.23|10.69|17.99|23.07|13.93|Star|STAR|K1|ugriz|18.78|17.12|16.42|16.
15|15.97|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-23.06902964||0.00000297
101002|J220953.17-020506.0|2011-10-24|55859|F5902|1|2|332.4715760000|-2.0850
150000|2.00|5.52|14.19|20.30|14.05|Star|STAR|M0|ugriz|20.91|18.10|16.66|16.0
5|15.67|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||27.10000040||0.00017775
101008|J220928.49-015720.7|2011-10-24|55859|F5902|1|8|332.3687450000|-1.9557
710000|1.84|9.94|25.25|32.32|18.29|Star|STAR|G5|ugriz|18.25|16.64|15.97|15.7
7|15.64|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||25.03866609||0.00000287
101009|J220849.59-015207.1|2011-10-24|55859|F5902|1|9|332.2066650000|-1.8686
530000|1.86|9.13|18.80|25.28|14.18|Star|STAR|G0|ugriz|18.64|17.19|16.63|16.3
7|16.25|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-22.16965227||0.00000537
101016|J220923.69-020809.9|2011-10-24|55859|F5902|1|16|332.3487250000|-2.136
0960000|2.17|28.22|52.30|72.89|46.52|Star|STAR|K5|ugriz|18.64|16.21|15.23|1
4.85|14.62|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-6.63140917||0.00000130
101017|J220946.66-015526.5|2011-10-24|55859|F5902|1|17|332.4444170000|-1.924
0460000|2.60|16.56|29.63|38.19|22.15|Star|STAR|G0|ugriz|17.97|16.53|16.00|1
5.78|15.65|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-2.46129608||0.00000233
101020|J220853.37-015915.4|2011-10-24|55859|F5902|1|20|332.2223790000|-1.987
6260000|2.65|17.26|26.29|36.30|20.29|Star|STAR|F5|ugriz|17.01|15.98|15.51|1
5.35|15.27|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||10.84948906||0.00000751
101021|J220924.33-014833.5|2011-10-24|55859|F5902|1|21|332.3513810000|-1.809
3330000|6.05|34.57|53.87|62.42|37.85|Star|STAR|F5|ugriz|16.75|15.61|15.16|1
4.98|14.92|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||-17.91859521||0.00000354
101023|J221001.52-020100.8|2011-10-24|55859|F5902|1|23|332.5063740000|-2.016
9000000|2.35|12.14|22.38|27.72|16.25|Star|STAR|F9|ugriz|18.46|16.97|16.39|1
6.18|16.12|99.00|99.00|JF_LEGAS_S|Obj|SDSS_S||52.65854525||0.00000227
```

In [88]:

```
dr1 = np.loadtxt('data/sample.txt',  delimiter="|", skiprows=1)
dr1
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-88-3b88b6873464> in <module>()
----> 1 dr1 = np.loadtxt('data/sample.txt',  delimiter="|", skiprows=1)
      2 dr1

/usr/local/lib/python3.5/site-packages/numpy/lib/npyio.py in loadtxt(fname,
 dtype, comments, delimiter, converters, skiprows, usecols, unpack, ndmin)
    928
    929             # Convert each value according to its column and store
--> 930             items = [conv(val) for (conv, val) in zip(converters, va
ls)]
    931             # Then pack it according to the dtype's nesting
    932             items = pack_items(items, packing)

/usr/local/lib/python3.5/site-packages/numpy/lib/npyio.py in <listcomp>(.0)
    928
    929             # Convert each value according to its column and store
--> 930             items = [conv(val) for (conv, val) in zip(converters, va
ls)]
    931             # Then pack it according to the dtype's nesting
    932             items = pack_items(items, packing)

/usr/local/lib/python3.5/site-packages/numpy/lib/npyio.py in floatconv(x)
    657         if b'0x' in x:
    658             return float.fromhex(asstr(x))
--> 659         return float(x)
    660
    661     typ = dtype.type

ValueError: could not convert string to float: b'J220848.54-020324.3'
```

In [89]:

```
!cat data/sample2.txt
```

```
obsid|designation|obsdate|lmjd
101001|J220848.54-020324.3|2011-10-24|55859
101002|J220953.17-020506.0|2011-10-24|55859
101008|J220928.49-015720.7|2011-10-24|55859
101009|J220849.59-015207.1|2011-10-24|55859
101016|J220923.69-020809.9|2011-10-24|55859
101017|J220946.66-015526.5|2011-10-24|55859
101020|J220853.37-015915.4|2011-10-24|55859
101021|J220924.33-014833.5|2011-10-24|55859
101023|J221001.52-020100.8|2011-10-24|55859
```

In [90]:

```
dtype = np.dtype([('obsid', 'S6'), ('designation', 'S19'), ('obsdate', 'S10'), ('lmjd', i
nt)])

dr1 = np.loadtxt('data/sample2.txt',  dtype=dtype, delimiter="|", skiprows=1)

dr1
```

Out[90]:

```
array([(b'101001', b'J220848.54-020324.3', b'2011-10-24', 55859),
       (b'101002', b'J220953.17-020506.0', b'2011-10-24', 55859),
       (b'101008', b'J220928.49-015720.7', b'2011-10-24', 55859),
       (b'101009', b'J220849.59-015207.1', b'2011-10-24', 55859),
       (b'101016', b'J220923.69-020809.9', b'2011-10-24', 55859),
       (b'101017', b'J220946.66-015526.5', b'2011-10-24', 55859),
       (b'101020', b'J220853.37-015915.4', b'2011-10-24', 55859),
       (b'101021', b'J220924.33-014833.5', b'2011-10-24', 55859),
       (b'101023', b'J221001.52-020100.8', b'2011-10-24', 55859)],
      dtype=[('obsid', 'S6'), ('designation', 'S19'), ('obsdate', 'S10'),
 ('lmjd', '<i8')])
```

In [91]:

```
dr1['lmjd']
```

Out[91]:

```
array([55859, 55859, 55859, 55859, 55859, 55859, 55859, 55859, 55859])
```

File format	Package name(s)	Functions
txt	numpy	loadtxt, savetxt, genfromtxt, fromfile, tofile
csv	csv	reader, writer
Matlab	scipy.io	loadmat, savemat
hdf	pytables, h5py	
NetCDF	netCDF4, scipy.io.netcdf	netCDF4.Dataset, scipy.io.netcdf.netcdf_file

This includes many industry specific formats:

File format	Package name	Comments
wav	scipy.io.wavfile	Audio files
LAS/SEG-Y	Scipy cookbook, Obspy	Data files in Geophysics
jpeg, png, …	PIL, scipy.misc.pilutil	Common image formats
FITS	pyfits, astropy.io.fits	Image files in Astronomy

通用函数（ufunc）

元素级的数组函数，对 `ndarray` 中的数据进行计算和操作。

- 一元 `ufunc`
- 二元 `ufunc`
- 自定义 `ufunc`

In [92]:

```
# 一元ufunc
arr = np.arange(10)

np.cos(arr)
```

Out[92]:

```
array([ 1.        ,  0.54030231, -0.41614684, -0.9899925 , -0.65364362,
        0.28366219,  0.96017029,  0.75390225, -0.14550003, -0.91113026])
```

In [93]:

```
# 二元ufunc

x = np.random.randn(10)
y = np.random.randn(10)

x, y
```

Out[93]:

```
(array([ 1.79600881, -0.81414819,  2.1464102 ,  0.65948138,  1.42033628,
        -1.46328631,  1.8278442 ,  2.21153387,  0.66340491, -0.57464657]),
 array([ 2.16758348, -0.55589951,  0.41189805, -0.19602406,  0.54262162,
        -0.7196864 ,  0.10274757, -1.58080509,  1.39606345, -0.56942818]))
```

In [94]:

```
np.add(x, y)
```

Out[94]:

```
array([ 3.96359229, -1.3700477 ,  2.55830825,  0.46345731,  1.9629579 ,
       -2.18297271,  1.93059177,  0.63072878,  2.05946836, -1.14407475])
```

In [95]:

```
## 自定义

def sqrt2(x,y):
    return np.sqrt(x**2 + y**2)

# input: 2, output: 1
sqrt2_uf = np.frompyfunc(sqrt2, 2, 1)

sqrt2_uf(x,y)
```

Out[95]:

```
array([2.8149717177862432, 0.98583038151946334, 2.1855746933041336,
       0.6879979065810834, 1.5204582791937427, 1.6306916752957041,
       1.8307297687838613, 2.7184235900325704, 1.5456711280930153,
       0.80899142758923637], dtype=object)
```

In [96]:

```
sqrt2_uf2 = np.vectorize(sqrt2, otypes=[np.float64])
sqrt2_uf2(x,y)
```

Out[96]:

```
array([ 2.81497172,  0.98583038,  2.18557469,  0.68799791,  1.52045828,
        1.63069168,  1.83072977,  2.71842359,  1.54567113,  0.80899143])
```

结构化数组

各列的数据类型可能不一致

In [97]:

```
dtype=[('RA', np.float64), ('Dec', np.float64), ('Type', np.int16)]

sarr = np.array([(1.23293, 23.231234, 12), (12.3242, 332.47876, 34)], dtype=dtype)

sarr
```

Out[97]:

```
array([(1.23293, 23.231234, 12), (12.3242, 332.47876, 34)],
      dtype=[('RA', '<f8'), ('Dec', '<f8'), ('Type', '<i2')])
```

In [98]:

```
sarr['RA']
```

Out[98]:

```
array([  1.23293,  12.3242 ])
```

In [1]:

```
%matplotlib inline
import pylab as plt
```

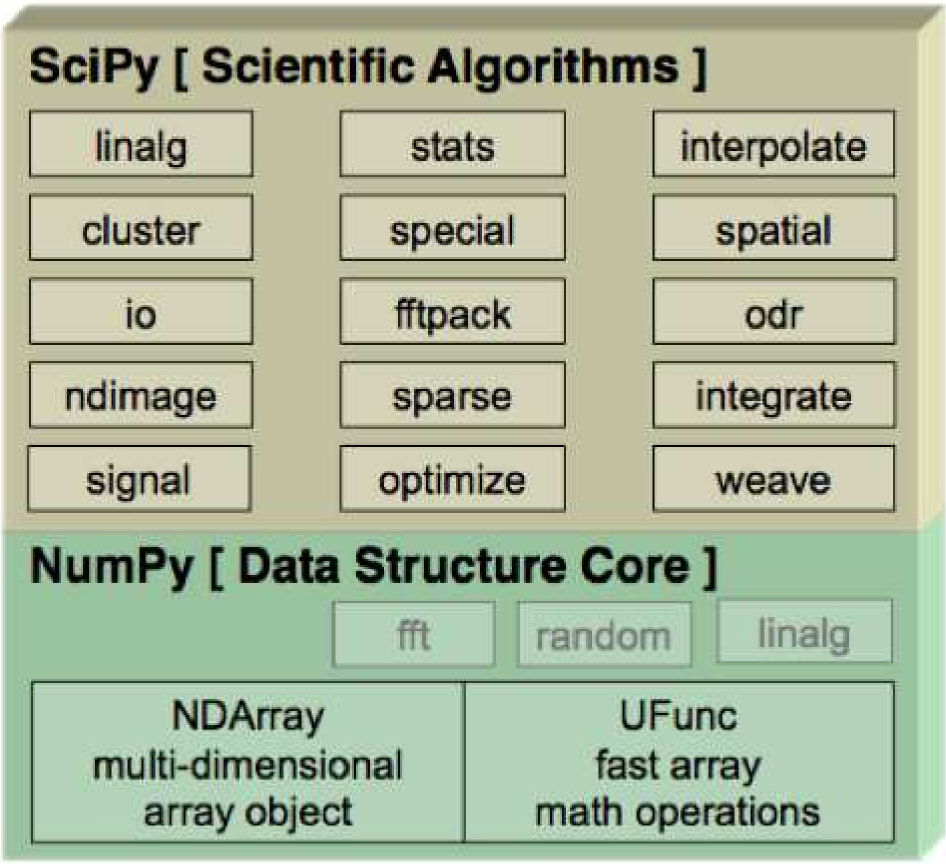
Python for Astronomy

SciPy

何勃亮
中国科学院国家天文台 中国虚拟天文台 (China-VO)

SciPy

NumPy 和 SciPy



SciPy

- Special functions (scipy.special (http://docs.scipy.org/doc/scipy/reference/special.html))
- Integration (scipy.integrate (http://docs.scipy.org/doc/scipy/reference/integrate.html))
- Optimization (scipy.optimize (http://docs.scipy.org/doc/scipy/reference/optimize.html))
- Interpolation (scipy.interpolate (http://docs.scipy.org/doc/scipy/reference/interpolate.html))
- Fourier Transforms (scipy.fftpack (http://docs.scipy.org/doc/scipy/reference/fftpack.html))
- Signal Processing (scipy.signal (http://docs.scipy.org/doc/scipy/reference/signal.html))
- Linear Algebra (scipy.linalg (http://docs.scipy.org/doc/scipy/reference/linalg.html))
- Sparse Eigenvalue Problems (scipy.sparse (http://docs.scipy.org/doc/scipy/reference/sparse.html))
- Statistics (scipy.stats (http://docs.scipy.org/doc/scipy/reference/stats.html))
- Multi-dimensional image processing (scipy.ndimage (http://docs.scipy.org/doc/scipy/reference/ndimage.html))
- File IO (scipy.io (http://docs.scipy.org/doc/scipy/reference/io.html))

In [2]:

```
import numpy as np
import scipy
```

Special functions

http://docs.scipy.org/doc/scipy/reference/special.html#module-scipy.special (http://docs.scipy.org/doc/scipy/reference/special.html#module-scipy.special)

In [3]:

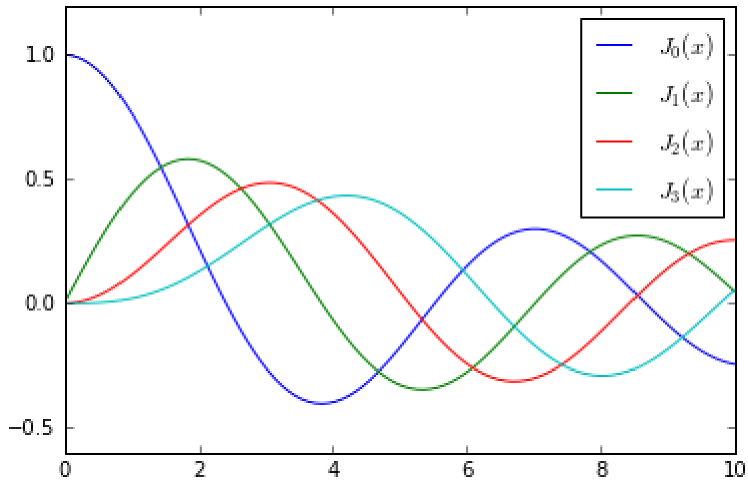
```
from scipy.special import jn

n = 0
x = 0.0

"J_{0:d}({1:f}) = {2:f}".format(n, x, jn(n, x))
```

Out[3]:

```
'J_0(0.000000) = 1.000000'
```

In [4]:

```
x = np.linspace(0, 10, 100)

fig, ax = plt.subplots()
for n in range(4):
    ax.plot(x, jn(n, x), label=r"$J_%d(x)$" % n)
ax.legend();
```



Integration

$$\int_a^b f(x)dx$$

In [5]:

```
from scipy.integrate import quad

# define a simple function for the integrand
def f(x):
    return x*x
```

In [6]:

```
x_lower = 0 # the lower limit of x
x_upper = 1 # the upper limit of x

val, abserr = quad(f, x_lower, x_upper)

"integral value ={}, absolute error ={}".format(val, abserr)
```

Out[6]:

```
'integral value =0.33333333333333337, absolute error =3.700743415417189e-15'
```

Fourier transform

SciPy使用的FFT函数来自于Fortran程序

In [8]:

```
from numpy.fft import fftfreq
from scipy.fftpack import *

t = np.random.rand(30)

N = len(t)
dt = t[1]-t[0]

# calculate the fast fourier transform
# y2 is the solution to the under-damped oscillator from the previous section
F = fft(y2[:,0])

# calculate the frequencies for the components in F
w = fftfreq(N, dt)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-8-f3c5af92dfd1> in <module>()
      9 # calculate the fast fourier transform
     10 # y2 is the solution to the under-damped oscillator from the previou
s section
---> 11 F = fft(y2[:,0])
     12 
     13 # calculate the frequencies for the components in F

NameError: name 'y2' is not defined
```

In [ ]:

```
fig, ax = plt.subplots(figsize=(9,3))
ax.plot(w_pos, abs(F_pos))
ax.set_xlim(0, 5);
```

Linear algebra

http://docs.scipy.org/doc/scipy/reference/linalg.html (http://docs.scipy.org/doc/scipy/reference/linalg.html)

线性方程组

$Ax = b$

In [9]:

```
from scipy.linalg import *

A = np.array([[1,2,3], [4,5,6], [7,8,9]])
b = np.array([1,2,3])
```

In [10]:

```
x = solve(A, b)

x
```

Out[10]:

```
array([-0.33333333,  0.66666667,  0.        ])
```

In [11]:

```
# check
np.dot(A, x) - b
```

Out[11]:

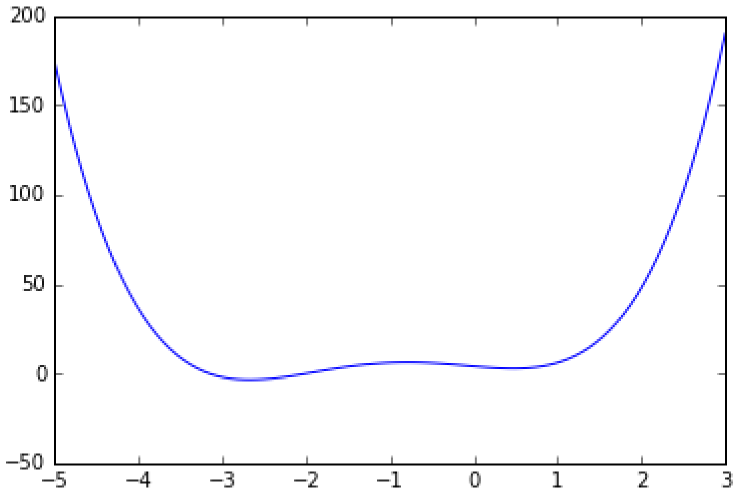
```
array([ -1.11022302e-16,   0.00000000e+00,   0.00000000e+00])
```

最优化 Optimization

寻找最小值

In [12]:

```
from scipy import optimize

def f(x):
    return 4*x**3 + (x-2)**2 + x**4

fig, ax  = plt.subplots()
x = np.linspace(-5, 3, 100)
ax.plot(x, f(x));
```



In [13]:

```
x_min = optimize.fmin_bfgs(f, -2)
x_min
```

```
Optimization terminated successfully.
         Current function value: -3.506641
         Iterations: 6
         Function evaluations: 30
         Gradient evaluations: 10
```

Out[13]:

```
array([-2.67298164])
```

In [14]:

```
x_min = optimize.fmin_bfgs(f, 0.5)
x_min
```

```
Optimization terminated successfully.
         Current function value: 2.804988
         Iterations: 3
         Function evaluations: 15
         Gradient evaluations: 5
```

Out[14]:

```
array([ 0.46961745])
```

In [15]:

```
optimize.brent(f)
```

Out[15]:

```
0.46961743402759754
```

In [16]:

```
optimize.fminbound(f, -4, 2)
```

Out[16]:

```
-2.6729822917513886
```

Interpolation

In [17]:

```
from scipy.interpolate import *
```

In [18]:

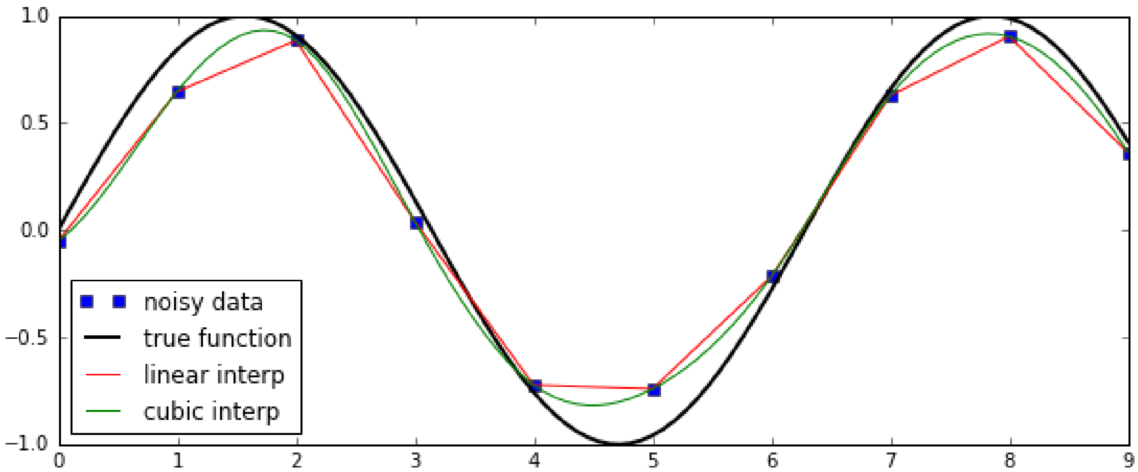
```
def f(x):
    return np.sin(x)
```

In [19]:

```
n = np.arange(0, 10)
x = np.linspace(0, 9, 100)

y_meas = f(n) + 0.1 * np.random.randn(len(n)) # simulate measurement with noise
y_real = f(x)

linear_interpolation = interp1d(n, y_meas)
y_interp1 = linear_interpolation(x)

cubic_interpolation = interp1d(n, y_meas, kind='cubic')
y_interp2 = cubic_interpolation(x)
```

In [20]:

```
fig, ax = plt.subplots(figsize=(10,4))
ax.plot(n, y_meas, 'bs', label='noisy data')
ax.plot(x, y_real, 'k', lw=2, label='true function')
ax.plot(x, y_interp1, 'r', label='linear interp')
ax.plot(x, y_interp2, 'g', label='cubic interp')
ax.legend(loc=3);
```



统计

http://docs.scipy.org/doc/scipy/reference/stats.html (http://docs.scipy.org/doc/scipy/reference/stats.html)

In [21]:

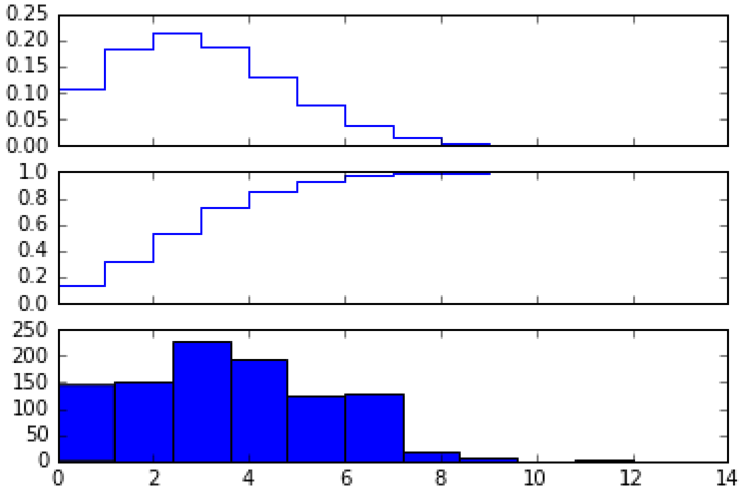
```
from scipy import stats
```

In [22]:

```
# create a (discreet) random variable with poissionian distribution

X = stats.poisson(3.5) # photon distribution for a coherent state with n=3.5 photons
```

In [23]:

```
n = np.arange(0,15)

fig, axes = plt.subplots(3,1, sharex=True)

# plot the probability mass function (PMF)
axes[0].step(n, X.pmf(n))

# plot the commulative distribution function (CDF)
axes[1].step(n, X.cdf(n))

# plot histogram of 1000 random realizations of the stochastic variable X
axes[2].hist(X.rvs(size=1000));
```

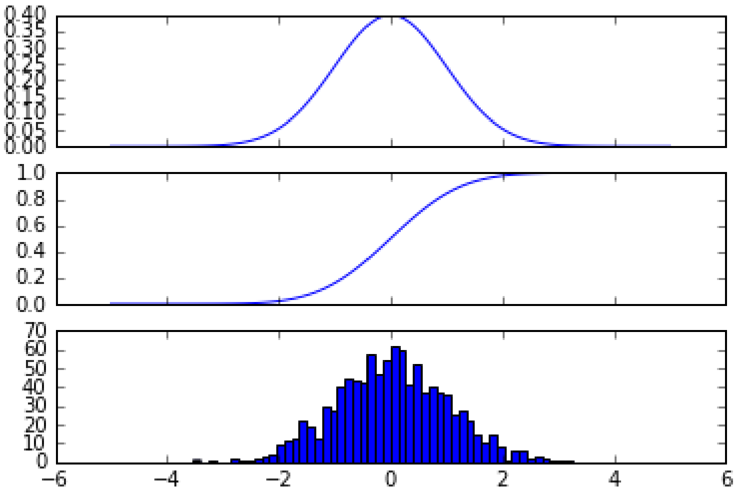


In [24]:

```
# create a (continous) random variable with normal distribution
Y = stats.norm()
```

In [25]:

```
x = np.linspace(-5,5,100)

fig, axes = plt.subplots(3,1, sharex=True)

# plot the probability distribution function (PDF)
axes[0].plot(x, Y.pdf(x))

# plot the commulative distributin function (CDF)
axes[1].plot(x, Y.cdf(x));

# plot histogram of 1000 random realizations of the stochastic variable Y
axes[2].hist(Y.rvs(size=1000), bins=50);
```



In [26]:

```
X.mean(), X.std(), X.var() # poission distribution
```

Out[26]:

```
(3.5, 1.8708286933869707, 3.5)
```

In [27]:

```
Y.mean(), Y.median(), Y.std(), Y.var() # normal distribution
```

Out[27]:

```
(0.0, 0.0, 1.0, 1.0)
```

In [28]:

```
#  Statistical tests

t_statistic, p_value = stats.ttest_ind(X.rvs(size=1000), X.rvs(size=1000))

print("t-statistic = {}".format(t_statistic))
print("p-value ={}".format(p_value))
```

```
t-statistic = 1.1286809118317047
p-value =0.25916796614330784
```

- rvs 生成随机数的采样函数
- pdf 概率密度函数 连续
- pmf 概率密度函数 离散
- cdf 累积密度函数
- ...

文件读写

http://docs.scipy.org/doc/scipy/reference/io.html (http://docs.scipy.org/doc/scipy/reference/io.html)

- Matlab
- IDL
- FortranFile
- NetCDF
- Wav sound
- Arff

In [1]:

```
%matplotlib inline
import numpy as np
```

Python for Astronomy

Matplotlib

何勃亮

中国科学院国家天文台 中国虚拟天文台 (China-VO)

In [2]:

```
import matplotlib
import matplotlib.pyplot as plt
```

In [3]:

```
%%HTML
<iframe width=100% height=600 src="http://matplotlib.org/gallery.html" ></iframe>
```

We're updating the default styles for Mat

Learn what to expect in the **new update**

home | examples | gallery | pyplot | docs »

Click on any image to see full size image and source code

- Gallery
  - Lines, bars, and markers
  - Shapes and collections
  - Statistical plots
  - Images, contours, and fields
  - Pie and polar charts
  - Color
  - Text, labels, and annotations
  - Ticks and spines
  - Axis scales
  - Subplots, axes, and figures

概念

plt

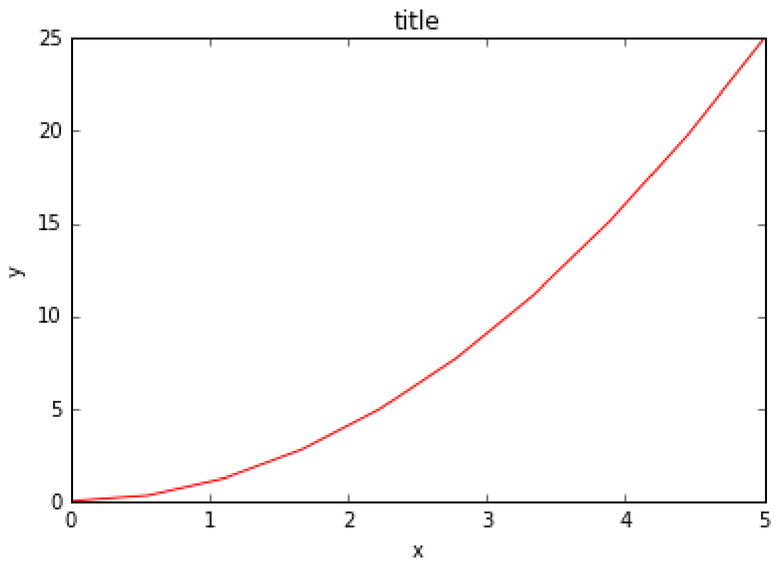
`import matplotlib.pyplot as plt`

matplotlib.pyplot is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. In matplotlib.pyplot various states are preserved across function calls, so that it keeps track of things like the current figure and plotting area, and the plotting functions are directed to the current axes (please note that “axes” here and in most places in the documentation refers to the axes part of a figure and not the strict mathematical term for more than one axis).

- `Matplotlib` 整个包
- `pyplot` 集成了方便用户进行绘图的一系列指令
- `pylab` 集成导入了 `pyplot` 和 `numpy` 在同一命名空间，但官方已经不推荐使用这个方式了。

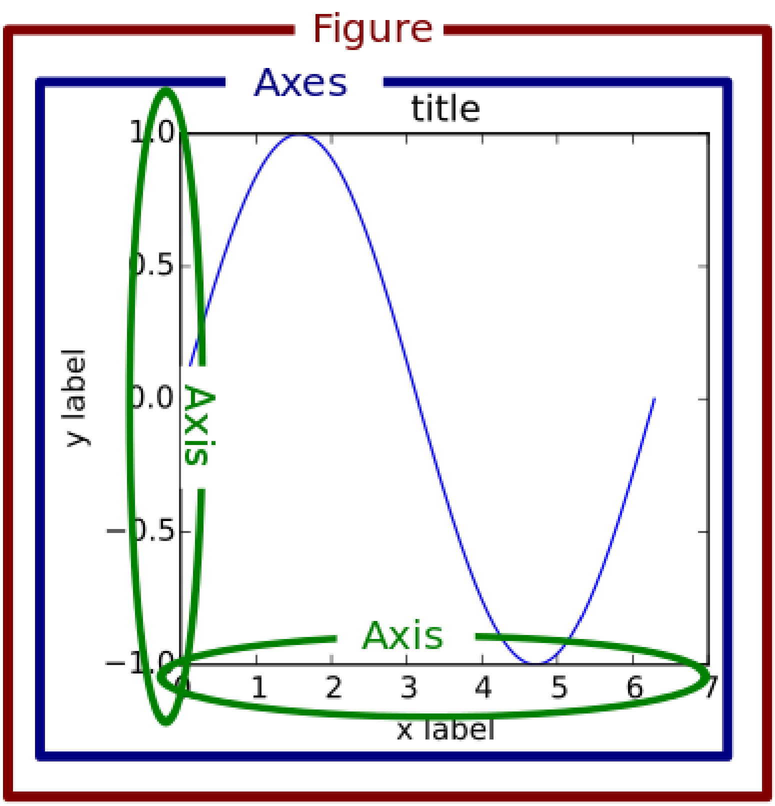
In [4]:

```
x = np.linspace(0, 5, 10)
y = x ** 2

fig = plt.figure()

axes = fig.add_axes([0.1, 0.1, 0.8, 0.8]) # left, bottom, width, height (range 0 to 1)

axes.plot(x, y, 'r')

axes.set_xlabel('x')
axes.set_ylabel('y')
axes.set_title('title');
```



Matplotlib 绘图步骤

1. 建立图像（`figure`）
2. 添加一个画布空间（`axes`）
3. 画图（`plot`）
4. 修改绘图参数（`label, title, etc`）



代码范式

1. 简单

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

x = ...
y = ...

fig = plt.figure()
ax = fig.add_subplot(111)
ax.plot(x, y)
plt.show()
```

定义绘图函数

```python
def my_plotter(ax, data1, data2, param_dict):
    """
    A helper function to make a graph

    Parameters
    ----------
    ax : Axes
        The axes to draw to

    data1 : array
       The x data

    data2 : array
       The y data

    param_dict : dict
       Dictionary of kwargs to pass to ax.plot

    Returns
    -------
    out : list
        list of artists added
    """
    out = ax.plot(data1, data2, **param_dict)
    return out
```

调用模式 1

```python
fig, ax = plt.subplots(1, 1)
my_plotter(ax, data1, data2, {'marker':'x'})
```

调用模式 2

```python
fig, (ax1, ax2) = plt.subplots(1, 2)
my_plotter(ax1, data1, data2, {'marker':'x'})
my_plotter(ax2, data3, data4, {'marker':'o'})
```

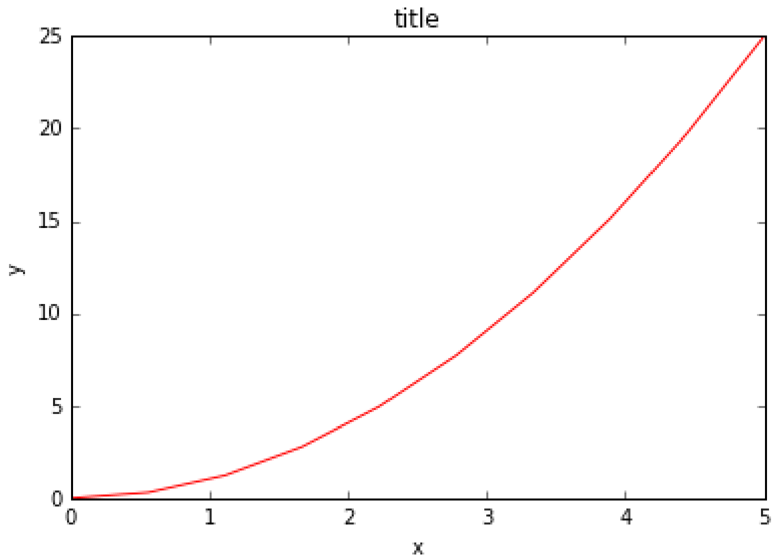
In [5]:

```
print(plt.style.available)
```

```
['ggplot', 'seaborn-bright', 'bmh', 'seaborn-darkgrid', 'seaborn-whitegrid',
 'seaborn-dark', 'seaborn-white', 'seaborn-colorblind', 'fivethirtyeight',
 'seaborn-notebook', 'classic', 'seaborn-pastel', 'dark_background', 'seabor
n-deep', 'seaborn-muted', 'seaborn-poster', 'seaborn-talk', 'seaborn-paper',
 'grayscale', 'seaborn-dark-palette', 'seaborn-ticks']
```

In [6]:

```
#plt.style.use('seaborn-paper')

x = np.linspace(0, 5, 10)
y = x ** 2

fig = plt.figure()

axes = fig.add_axes([0.1, 0.1, 0.8, 0.8]) # left, bottom, width, height (range 0 to 1)

axes.plot(x, y, 'r')

axes.set_xlabel('x')
axes.set_ylabel('y')
axes.set_title('title');
```



In [7]:

```
help(plt.plot)
```

```
Help on function plot in module matplotlib.pyplot:

plot(*args, **kwargs)
    Plot lines and/or markers to the
    :class:`~matplotlib.axes.Axes`.  *args* is a variable length
    argument, allowing for multiple *x*, *y* pairs with an
    optional format string.  For example, each of the following is
    legal::

        plot(x, y)        # plot x and y using default line style and color
        plot(x, y, 'bo')  # plot x and y using blue circle markers
        plot(y)           # plot y using x as index array 0..N-1
        plot(y, 'r+')     # ditto, but with red plusses

    If *x* and/or *y* is 2-dimensional, then the corresponding columns
    will be plotted.

    If used with labeled data, make sure that the color spec is not
    included as an element in data, as otherwise the last case
    ``plot("v","r", data={"v":..., "r":...)``
    can be interpreted as the first case which would do ``plot(v, r)``
    using the default line style and color.

    If not used with labeled data (i.e., without a data argument),
    an arbitrary number of *x*, *y*, *fmt* groups can be specified, as in::

        a.plot(x1, y1, 'g^', x2, y2, 'g-')

    Return value is a list of lines that were added.

    By default, each line is assigned a different style specified by a
    'style cycle'.  To change this behavior, you can edit the
    axes.prop_cycle rcParam.

    The following format string characters are accepted to control
    the line style or marker:

    ================    ===============================
    character           description
    ================    ===============================
    ``'-'``             solid line style
    ``'--'``            dashed line style
    ``'-.'``            dash-dot line style
    ``':'``             dotted line style
    ``'.'``             point marker
    ``','``             pixel marker
    ``'o'``             circle marker
    ``'v'``             triangle_down marker
    ``'^'``             triangle_up marker
    ``'<'``             triangle_left marker
    ``'>'``             triangle_right marker
    ``'1'``             tri_down marker
    ``'2'``             tri_up marker
    ``'3'``             tri_left marker
    ``'4'``             tri_right marker
    ``'s'``             square marker
    ``'p'``             pentagon marker
    ``'*'``             star marker
    ``'h'``             hexagon1 marker
    ``'H'``             hexagon2 marker
    ``'+'``             plus marker
```

```
``'x'``             x marker
``'D'``             diamond marker
``'d'``             thin_diamond marker
``'|'``             vline marker
``'_'``             hline marker
================    ===============================
```

The following color abbreviations are supported:

```
==========  ========
character   color
==========  ========
'b'         blue
'g'         green
'r'         red
'c'         cyan
'm'         magenta
'y'         yellow
'k'         black
'w'         white
==========  ========
```

In addition, you can specify colors in many weird and
wonderful ways, including full names (``'green'``), hex
strings (``'#008000'``), RGB or RGBA tuples (``(0,1,0,1)``) or
grayscale intensities as a string (``'0.8'``). Of these, the
string specifications can be used in place of a ``fmt`` group,
but the tuple forms can be used only as ``kwargs``.

Line styles and colors are combined in a single format string, as in
``'bo'`` for blue circles.

The *kwargs* can be used to set line properties (any property that has
a ``set_*`` method). You can use this to set a line label (for auto
legends), linewidth, anitialising, marker face color, etc. Here is an
example::

```
    plot([1,2,3], [1,2,3], 'go-', label='line 1', linewidth=2)
    plot([1,2,3], [1,4,9], 'rs',  label='line 2')
    axis([0, 4, 0, 10])
    legend()
```

If you make multiple lines with one plot command, the kwargs
apply to all those lines, e.g.::

```
    plot(x1, y1, x2, y2, antialiased=False)
```

Neither line will be antialiased.

You do not need to use format strings, which are just
abbreviations. All of the line properties can be controlled
by keyword arguments. For example, you can set the color,
marker, linestyle, and markercolor with::

```
    plot(x, y, color='green', linestyle='dashed', marker='o',
         markerfacecolor='blue', markersize=12).
```

See :class:`~matplotlib.lines.Line2D` for details.

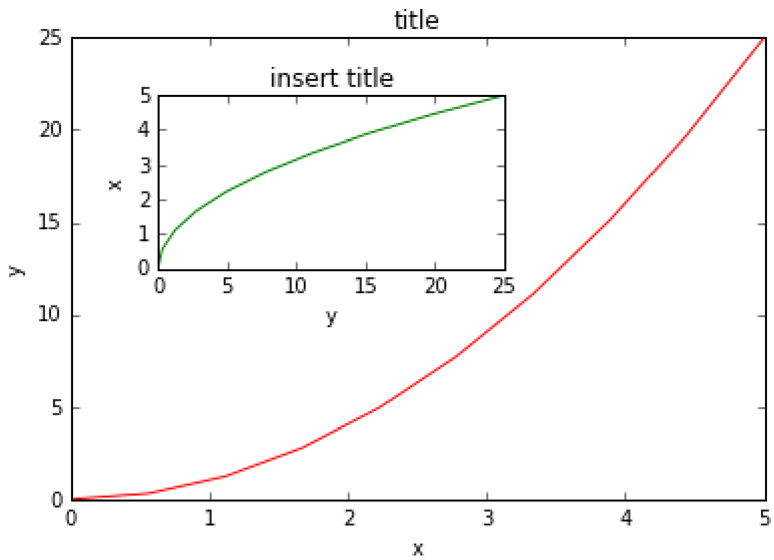
The kwargs are :class:`~matplotlib.lines.Line2D` properties:

```
      agg_filter: unknown
      alpha: float (0.0 transparent through 1.0 opaque)
      animated: [True | False]
      antialiased or aa: [True | False]
      axes: an :class:`~matplotlib.axes.Axes` instance
      clip_box: a :class:`matplotlib.transforms.Bbox` instance
      clip_on: [True | False]
      clip_path: [ (:class:`~matplotlib.path.Path`, :class:`~matplotlib.tran
sforms.Transform`) | :class:`~matplotlib.patches.Patch` | None ]
      color or c: any matplotlib color
      contains: a callable function
      dash_capstyle: ['butt' | 'round' | 'projecting']
      dash_joinstyle: ['miter' | 'round' | 'bevel']
      dashes: sequence of on/off ink in points
      drawstyle: ['default' | 'steps' | 'steps-pre' | 'steps-mid' | 'steps-p
ost']
      figure: a :class:`matplotlib.figure.Figure` instance
      fillstyle: ['full' | 'left' | 'right' | 'bottom' | 'top' | 'none']
      gid: an id string
      label: string or anything printable with '%s' conversion.
      linestyle or ls: ['solid' | 'dashed', 'dashdot', 'dotted' | (offset, o
n-off-dash-seq) | ``'-'`` | ``'--'`` | ``'-.'`` | ``':'`` | ``'None'`` | ``'
 '`` | ``''``]
      linewidth or lw: float value in points
      marker: :mod:`A valid marker style <matplotlib.markers>`
      markeredgecolor or mec: any matplotlib color
      markeredgewidth or mew: float value in points
      markerfacecolor or mfc: any matplotlib color
      markerfacecoloralt or mfcalt: any matplotlib color
      markersize or ms: float
      markevery: [None | int | length-2 tuple of int | slice | list/array of
 int | float | length-2 tuple of float]
      path_effects: unknown
      picker: float distance in points or callable pick function ``fn(artis
t, event)``
      pickradius: float distance in points
      rasterized: [True | False | None]
      sketch_params: unknown
      snap: unknown
      solid_capstyle: ['butt' | 'round' |  'projecting']
      solid_joinstyle: ['miter' | 'round' | 'bevel']
      transform: a :class:`matplotlib.transforms.Transform` instance
      url: a url string
      visible: [True | False]
      xdata: 1D array
      ydata: 1D array
      zorder: any number

    kwargs *scalex* and *scaley*, if defined, are passed on to
    :meth:`~matplotlib.axes.Axes.autoscale_view` to determine
    whether the *x* and *y* axes are autoscaled; the default is
    *True*.

    Notes
    -----

    In addition to the above described arguments, this function can take a
    **data** keyword argument. If such a **data** argument is given, the
    following arguments are replaced by **data[<arg>]**:
```

```
* All arguments with the following names: 'y', 'x'.



Additional kwargs: hold = [True|False] overrides default hold state
```

In [8]:

```
fig = plt.figure()

axes1 = fig.add_axes([0.1, 0.1, 0.8, 0.8]) # main axes
axes2 = fig.add_axes([0.2, 0.5, 0.4, 0.3]) # inset axes

# main figure
axes1.plot(x, y, 'r')
axes1.set_xlabel('x')
axes1.set_ylabel('y')
axes1.set_title('title')

# insert
axes2.plot(y, x, 'g')
axes2.set_xlabel('y')
axes2.set_ylabel('x')
axes2.set_title('insert title');
```



图像尺寸，比例和分辨率

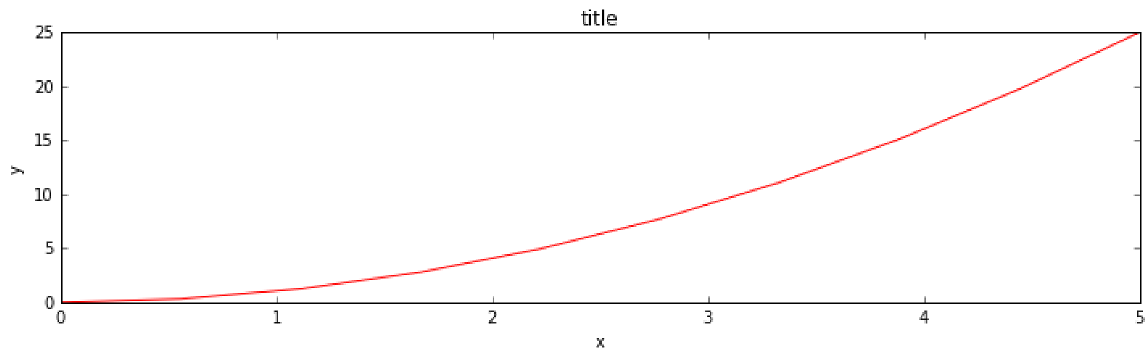
In [9]:

```
# 800 x 400, 100dpi

fig = plt.figure(figsize=(8,4), dpi=100)
```

```
<matplotlib.figure.Figure at 0x1020f9c18>
```

In [10]:

```
fig, ax = plt.subplots(figsize=(12,3))

ax.plot(x, y, 'r')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_title('title');
```



In [11]:

```
# 保存图像

fig.savefig("filename.png")

fig.savefig("filename200.png", dpi=200)
```

图例，标题

In [12]:

```
ax.set_title("title");
```

In [13]:

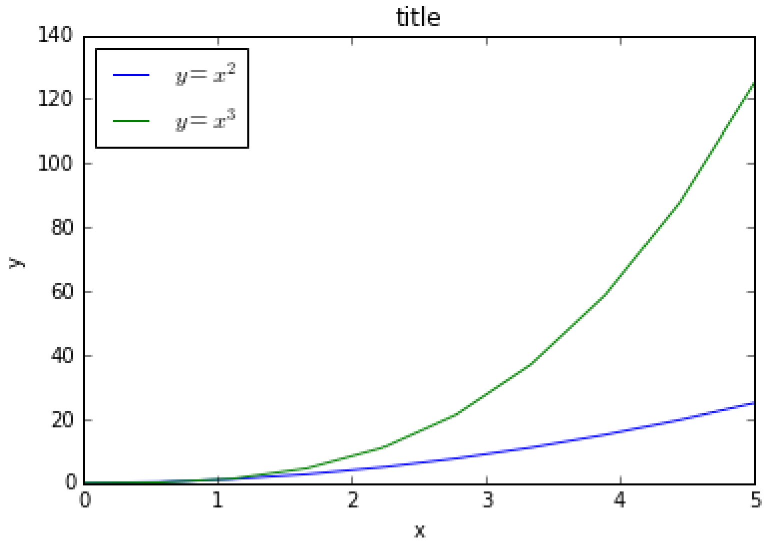
```
ax.set_xlabel("x")
ax.set_ylabel("y");
```

In [14]:

```
ax.plot(x, x**2, label="curve1")
ax.plot(x, x**3, label="curve2")
ax.legend();
```

In [15]:

```
fig, ax = plt.subplots()

ax.plot(x, x**2, label="$y = x^2$")
ax.plot(x, x**3, label="$y = x^3$")
ax.legend(loc=2); # upper left corner
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_title('title');
```

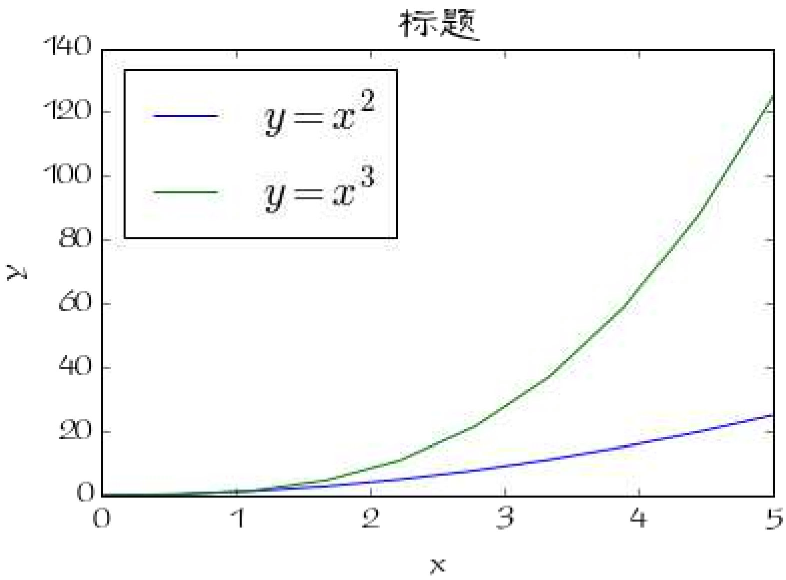


In [16]:

```
# Update the matplotlib configuration parameters:
matplotlib.rcParams.update({'font.size': 18, 'font.family': 'FZGuLi-S12S'})
```

In [17]:

```
fig, ax = plt.subplots()

ax.plot(x, x**2, label="$y = x^2$")
ax.plot(x, x**3, label="$y = x^3$")
ax.legend(loc=2); # upper left corner
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_title('标题');
```



In [18]:

```
# restore
matplotlib.rcParams.update({'font.size': 12, 'font.family': 'sans', 'text.usetex': False})
```

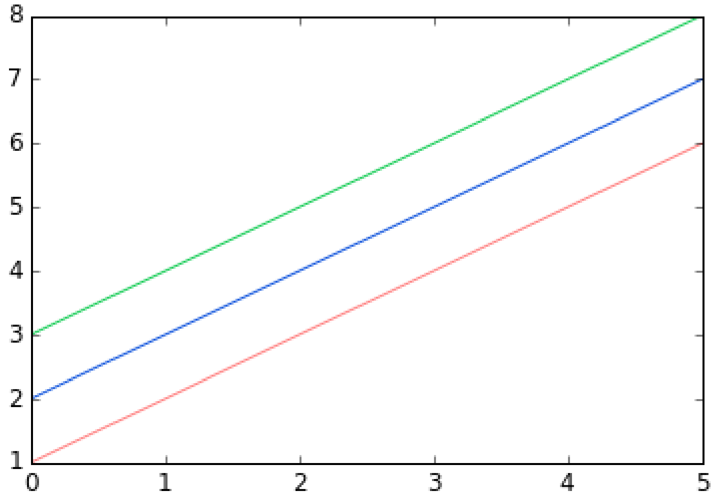
颜色，线宽，线型

In [19]:

```
fig, ax = plt.subplots()

ax.plot(x, x+1, color="red", alpha=0.5) # half-transparant red
ax.plot(x, x+2, color="#1155dd")        # RGB hex code for a bluish color
ax.plot(x, x+3, color="#15cc55")        # RGB hex code for a greenish color
```

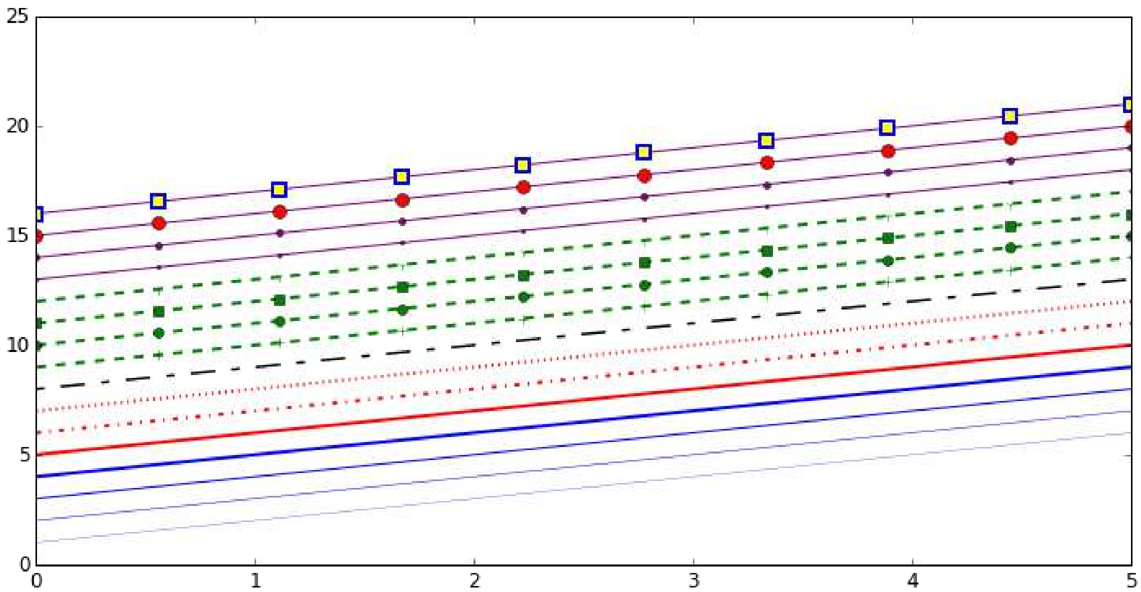
Out[19]:

```
[<matplotlib.lines.Line2D at 0x103c93f98>]
```



In [20]:

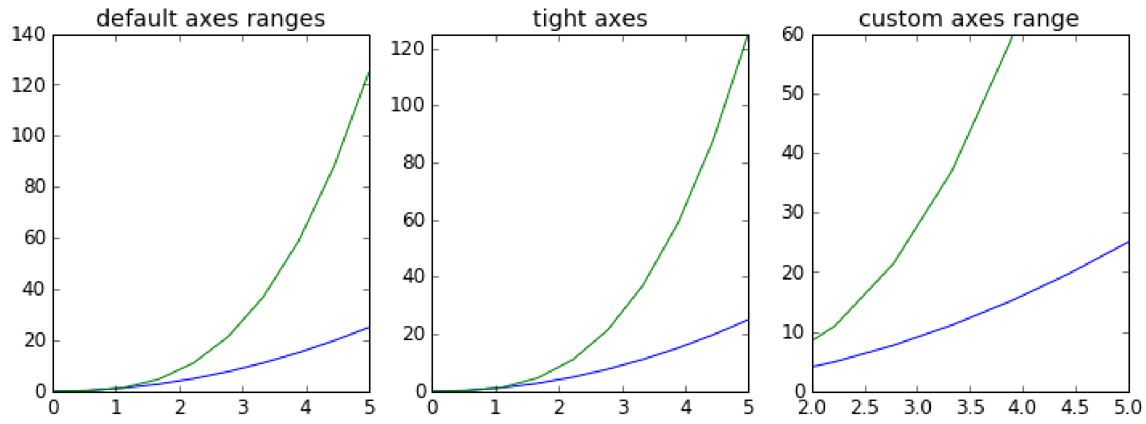
```
fig, ax = plt.subplots(figsize=(12,6))

ax.plot(x, x+1, color="blue", linewidth=0.25)
ax.plot(x, x+2, color="blue", linewidth=0.50)
ax.plot(x, x+3, color="blue", linewidth=1.00)
ax.plot(x, x+4, color="blue", linewidth=2.00)

# possible linestype options ‘-‘, ‘--’, ‘-.’, ‘:’, ‘steps’
ax.plot(x, x+5, color="red", lw=2, linestyle='-')
ax.plot(x, x+6, color="red", lw=2, ls='-.')
ax.plot(x, x+7, color="red", lw=2, ls=':')

# custom dash
line, = ax.plot(x, x+8, color="black", lw=1.50)
line.set_dashes([5, 10, 15, 10]) # format: line length, space length, ...

# possible marker symbols: marker = '+', 'o', '*', 's', ',', '.', '1', '2', '3', '4', ...
ax.plot(x, x+ 9, color="green", lw=2, ls='--', marker='+')
ax.plot(x, x+10, color="green", lw=2, ls='--', marker='o')
ax.plot(x, x+11, color="green", lw=2, ls='--', marker='s')
ax.plot(x, x+12, color="green", lw=2, ls='--', marker='1')

# marker size and color
ax.plot(x, x+13, color="purple", lw=1, ls='-', marker='o', markersize=2)
ax.plot(x, x+14, color="purple", lw=1, ls='-', marker='o', markersize=4)
ax.plot(x, x+15, color="purple", lw=1, ls='-', marker='o', markersize=8,
markerfacecolor="red")
ax.plot(x, x+16, color="purple", lw=1, ls='-', marker='s', markersize=8,
        markerfacecolor="yellow", markeredgewidth=2, markeredgecolor="blue");
```



坐标轴

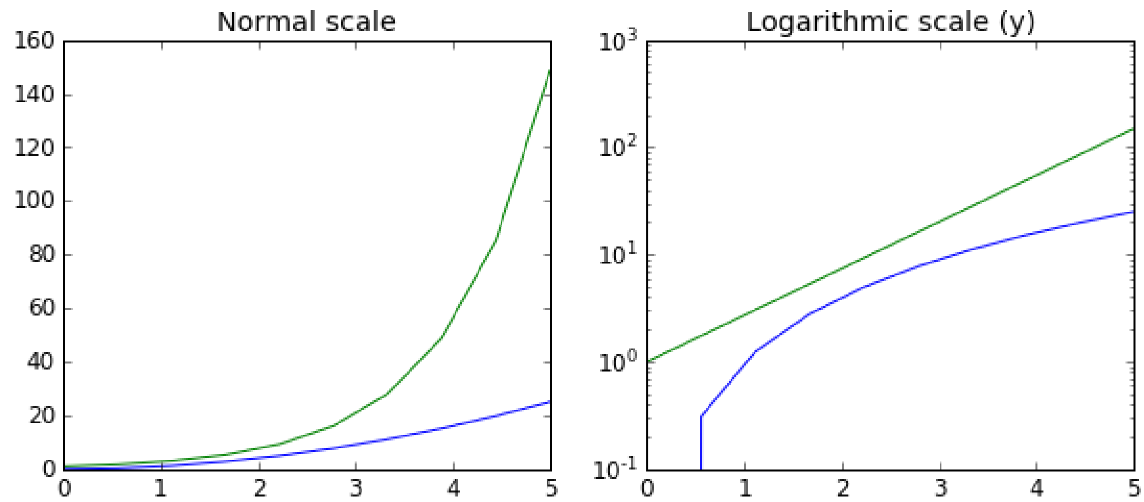
In [21]:

```
fig, axes = plt.subplots(1, 3, figsize=(12, 4))

axes[0].plot(x, x**2, x, x**3)
axes[0].set_title("default axes ranges")

axes[1].plot(x, x**2, x, x**3)
axes[1].axis('tight')
axes[1].set_title("tight axes")

axes[2].plot(x, x**2, x, x**3)
axes[2].set_ylim([0, 60])
axes[2].set_xlim([2, 5])
axes[2].set_title("custom axes range");
```



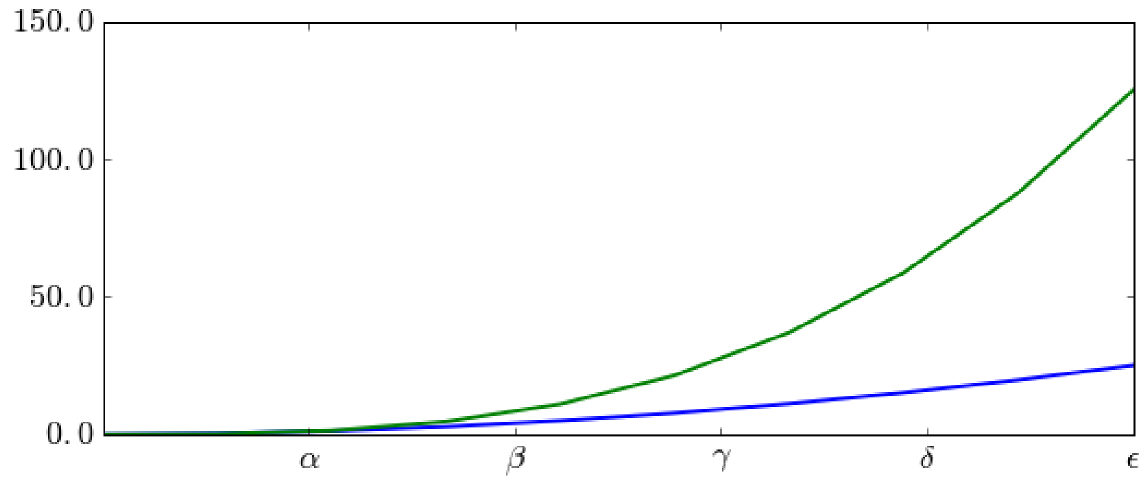
In [22]:

```
## Logarithmic scale

fig, axes = plt.subplots(1, 2, figsize=(10,4))

axes[0].plot(x, x**2, x, np.exp(x))
axes[0].set_title("Normal scale")

axes[1].plot(x, x**2, x, np.exp(x))
axes[1].set_yscale("log")
axes[1].set_title("Logarithmic scale (y)");
```



In [23]:

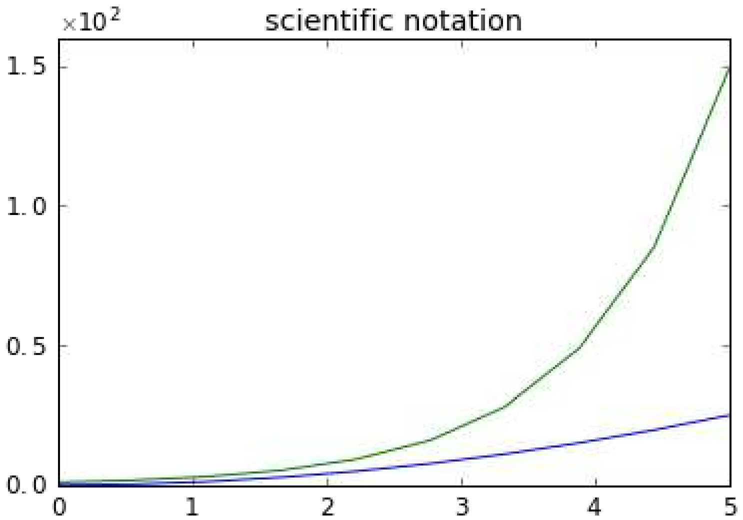
```
#  定制刻度显示

fig, ax = plt.subplots(figsize=(10, 4))

ax.plot(x, x**2, x, x**3, lw=2)

ax.set_xticks([1, 2, 3, 4, 5])
ax.set_xticklabels([r'$\alpha$', r'$\beta$', r'$\gamma$', r'$\delta$', r'$\epsilon$'], fo
ntsize=18)

yticks = [0, 50, 100, 150]
ax.set_yticks(yticks)
ax.set_yticklabels(["$%.1f$" % y for y in yticks], fontsize=18); # use LaTeX formatted la
bels
```



http://matplotlib.org/api/ticker_api.html (http://matplotlib.org/api/ticker_api.html)

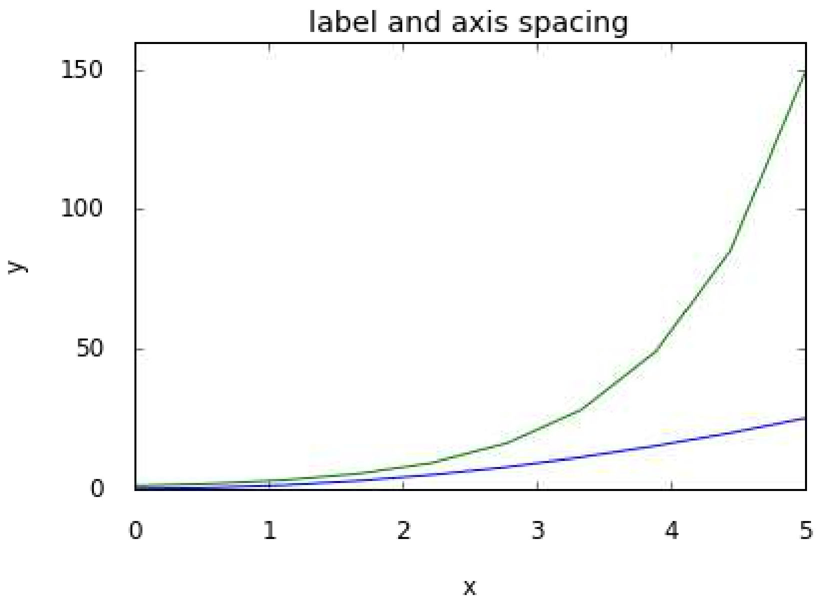
In [24]:

```
# 科学记数法

fig, ax = plt.subplots(1, 1)

ax.plot(x, x**2, x, np.exp(x))
ax.set_title("scientific notation")

ax.set_yticks([0, 50, 100, 150])

from matplotlib import ticker
formatter = ticker.ScalarFormatter(useMathText=True)
formatter.set_scientific(True)
formatter.set_powerlimits((-1,1))
ax.yaxis.set_major_formatter(formatter)
```



In [25]:

```
# distance between x and y axis and the numbers on the axes
matplotlib.rcParams['xtick.major.pad'] = 15
matplotlib.rcParams['ytick.major.pad'] = 15

fig, ax = plt.subplots(1, 1)

ax.plot(x, x**2, x, np.exp(x))
ax.set_yticks([0, 50, 100, 150])

ax.set_title("label and axis spacing")

# padding between axis label and axis numbers
ax.xaxis.labelpad = 15
ax.yaxis.labelpad = 15

ax.set_xlabel("x")
ax.set_ylabel("y");
```

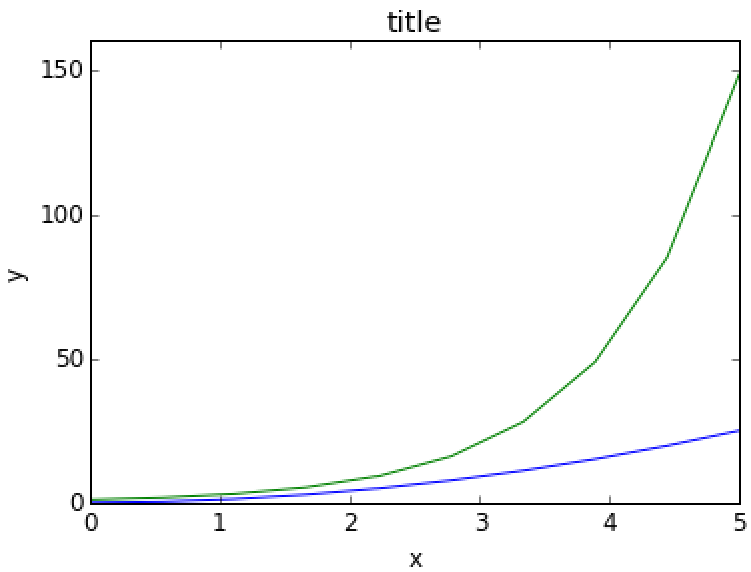


In [26]:

```
# restore defaults
matplotlib.rcParams['xtick.major.pad'] = 3
matplotlib.rcParams['ytick.major.pad'] = 3
```

In [27]:

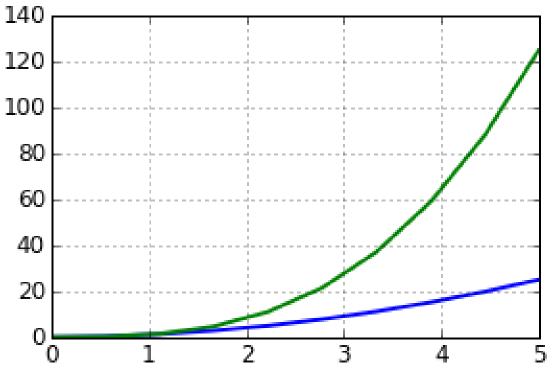
```
# Axis position adjustments

fig, ax = plt.subplots(1, 1)

ax.plot(x, x**2, x, np.exp(x))
ax.set_yticks([0, 50, 100, 150])

ax.set_title("title")
ax.set_xlabel("x")
ax.set_ylabel("y")

fig.subplots_adjust(left=0.15, right=.9, bottom=0.1, top=0.9);
```

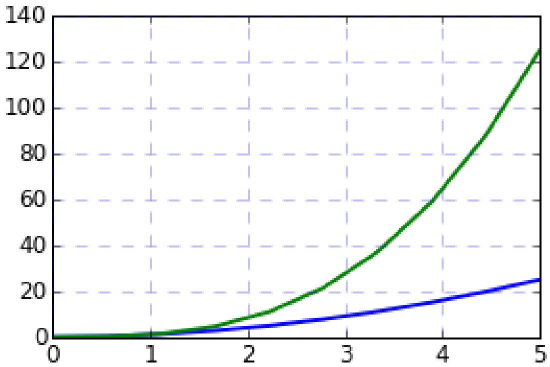


网格

In [28]:

```
fig, axes = plt.subplots(1, 2, figsize=(10,3))

# default grid appearance
axes[0].plot(x, x**2, x, x**3, lw=2)
axes[0].grid(True)

# custom grid appearance
axes[1].plot(x, x**2, x, x**3, lw=2)
axes[1].grid(color='b', alpha=0.5, linestyle='dashed', linewidth=0.5)
```

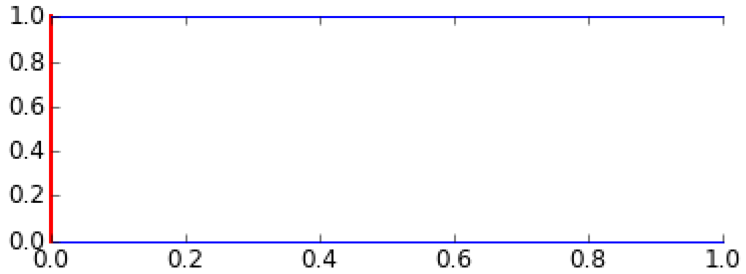




坐标轴颜色

In [29]:

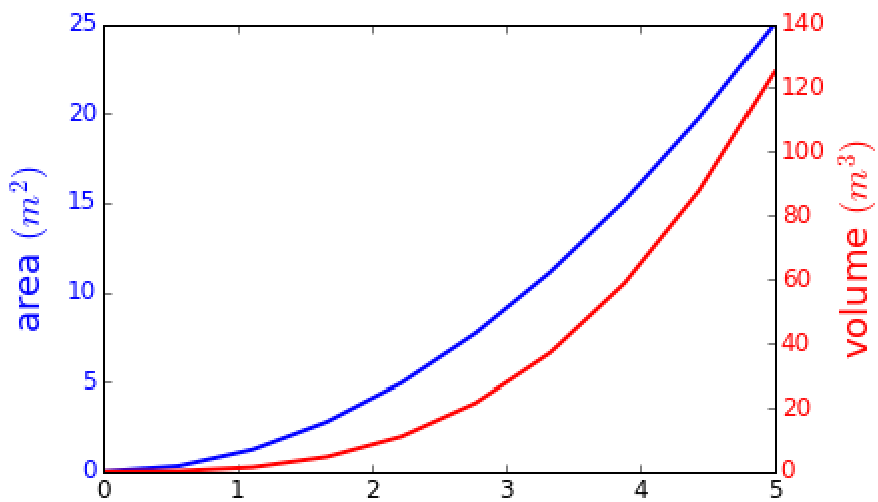
```
fig, ax = plt.subplots(figsize=(6,2))

ax.spines['bottom'].set_color('blue')
ax.spines['top'].set_color('blue')

ax.spines['left'].set_color('red')
ax.spines['left'].set_linewidth(2)

# turn off axis spine to the right
ax.spines['right'].set_color("none")
ax.yaxis.tick_left() # only ticks on the left side
```



双坐标轴

In [30]:

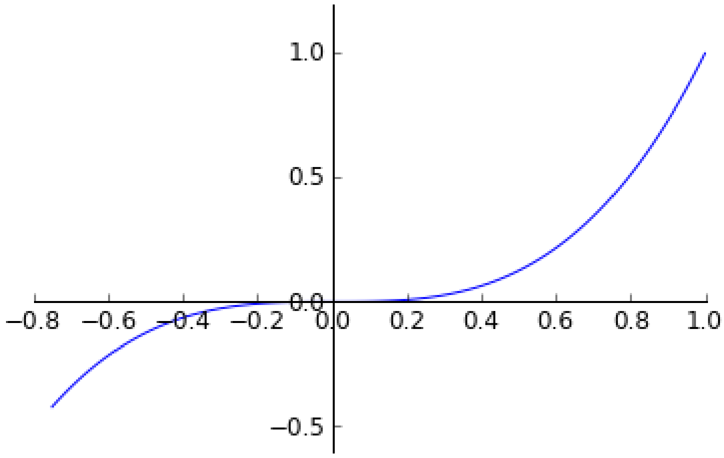
```
fig, ax1 = plt.subplots()

ax1.plot(x, x**2, lw=2, color="blue")
ax1.set_ylabel(r"area $(m^2)$", fontsize=18, color="blue")
for label in ax1.get_yticklabels():
    label.set_color("blue")

ax2 = ax1.twinx()
ax2.plot(x, x**3, lw=2, color="red")
ax2.set_ylabel(r"volume $(m^3)$", fontsize=18, color="red")
for label in ax2.get_yticklabels():
    label.set_color("red")
```



去掉边框

In [31]:

```
fig, ax = plt.subplots()

ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')

ax.xaxis.set_ticks_position('bottom')
ax.spines['bottom'].set_position(('data',0)) # set position of x spine to x=0

ax.yaxis.set_ticks_position('left')
ax.spines['left'].set_position(('data',0))   # set position of y spine to y=0

xx = np.linspace(-0.75, 1., 100)
ax.plot(xx, xx**3);
```



各种图型

[http://matplotlib.org/gallery.html (http://matplotlib.org/gallery.html)](http://matplotlib.org/gallery.html)

散点图等

In [32]:

```
help(plt.scatter)
```

```
Help on function scatter in module matplotlib.pyplot:

scatter(x, y, s=20, c=None, marker='o', cmap=None, norm=None, vmin=None, vma
x=None, alpha=None, linewidths=None, verts=None, edgecolors=None, hold=None,
 data=None, **kwargs)
    Make a scatter plot of x vs y, where x and y are sequence like objects
    of the same lengths.

    Parameters
    ----------
    x, y : array_like, shape (n, )
        Input data

    s : scalar or array_like, shape (n, ), optional, default: 20
        size in points^2.

    c : color or sequence of color, optional, default : 'b'
        `c` can be a single color format string, or a sequence of color
        specifications of length `N`, or a sequence of `N` numbers to be
        mapped to colors using the `cmap` and `norm` specified via kwargs
        (see below). Note that `c` should not be a single numeric RGB or
        RGBA sequence because that is indistinguishable from an array of
        values to be colormapped.  `c` can be a 2-D array in which the
        rows are RGB or RGBA, however, including the case of a single
        row to specify the same color for all points.

    marker : `~matplotlib.markers.MarkerStyle`, optional, default: 'o'
        See `~matplotlib.markers` for more information on the different
        styles of markers scatter supports. `marker` can be either
        an instance of the class or the text shorthand for a particular
        marker.

    cmap : `~matplotlib.colors.Colormap`, optional, default: None
        A `~matplotlib.colors.Colormap` instance or registered name.
        `cmap` is only used if `c` is an array of floats. If None,
        defaults to rc `image.cmap`.

    norm : `~matplotlib.colors.Normalize`, optional, default: None
        A `~matplotlib.colors.Normalize` instance is used to scale
        luminance data to 0, 1. `norm` is only used if `c` is an array of
        floats. If `None`, use the default :func:`normalize`.

    vmin, vmax : scalar, optional, default: None
        `vmin` and `vmax` are used in conjunction with `norm` to normalize
        luminance data.  If either are `None`, the min and max of the
        color array is used.  Note if you pass a `norm` instance, your
        settings for `vmin` and `vmax` will be ignored.

    alpha : scalar, optional, default: None
        The alpha blending value, between 0 (transparent) and 1 (opaque)

    linewidths : scalar or array_like, optional, default: None
        If None, defaults to (lines.linewidth,).

    edgecolors : color or sequence of color, optional, default: None
        If None, defaults to (patch.edgecolor).
        If 'face', the edge color will always be the same as
        the face color.  If it is 'none', the patch boundary will not
        be drawn.  For non-filled markers, the `edgecolors` kwarg
        is ignored; color is determined by `c`.
```

```
    Returns
    -------
    paths : `~matplotlib.collections.PathCollection`

    Other parameters
    ----------------
    kwargs : `~matplotlib.collections.Collection` properties

    Notes
    ------
    Any or all of `x`, `y`, `s`, and `c` may be masked arrays, in
    which case all masks will be combined and only unmasked points
    will be plotted.

    Fundamentally, scatter works with 1-D arrays; `x`, `y`, `s`,
    and `c` may be input as 2-D arrays, but within scatter
    they will be flattened. The exception is `c`, which
    will be flattened only if its size matches the size of `x`
    and `y`.

    Examples
    --------
    .. plot:: mpl_examples/shapes_and_collections/scatter_demo.py

    Notes
    -----

    In addition to the above described arguments, this function can take a
    **data** keyword argument. If such a **data** argument is given, the
    following arguments are replaced by **data[<arg>]**:

    * All arguments with the following names: 'linewidths', 'y', 'edgecolor
s', 'facecolor', 's', 'facecolors', 'c', 'x', 'color'.




    Additional kwargs: hold = [True|False] overrides default hold state
```

In [33]:

```
help(plt.step)
```

```
Help on function step in module matplotlib.pyplot:

step(x, y, *args, **kwargs)
    Make a step plot.

    Call signature::

      step(x, y, *args, **kwargs)

    Additional keyword args to :func:`step` are the same as those
    for :func:`~matplotlib.pyplot.plot`.

    *x* and *y* must be 1-D sequences, and it is assumed, but not checked,
    that *x* is uniformly increasing.

    Keyword arguments:

    *where*: [ 'pre' | 'post' | 'mid'  ]
      If 'pre' (the default), the interval from x[i] to x[i+1] has level
      y[i+1].

      If 'post', that interval has level y[i].

      If 'mid', the jumps in *y* occur half-way between the
      *x*-values.

    Return value is a list of lines that were added.

    Notes
    -----

    In addition to the above described arguments, this function can take a
    **data** keyword argument. If such a **data** argument is given, the
    following arguments are replaced by **data[<arg>]**:

    * All arguments with the following names: 'y', 'x'.




    Additional kwargs: hold = [True|False] overrides default hold state
```

In [34]:

```
help(plt.bar)
```

```
Help on function bar in module matplotlib.pyplot:

bar(left, height, width=0.8, bottom=None, hold=None, data=None, **kwargs)
    Make a bar plot.

    Make a bar plot with rectangles bounded by:

      `left`, `left` + `width`, `bottom`, `bottom` + `height`
            (left, right, bottom and top edges)

    Parameters
    ----------
    left : sequence of scalars
        the x coordinates of the left sides of the bars

    height : sequence of scalars
        the heights of the bars

    width : scalar or array-like, optional
        the width(s) of the bars
        default: 0.8

    bottom : scalar or array-like, optional
        the y coordinate(s) of the bars
        default: None

    color : scalar or array-like, optional
        the colors of the bar faces

    edgecolor : scalar or array-like, optional
        the colors of the bar edges

    linewidth : scalar or array-like, optional
        width of bar edge(s). If None, use default
        linewidth; If 0, don't draw edges.
        default: None

    tick_label : string or array-like, optional
        the tick labels of the bars
        default: None

    xerr : scalar or array-like, optional
        if not None, will be used to generate errorbar(s) on the bar chart
        default: None

    yerr : scalar or array-like, optional
        if not None, will be used to generate errorbar(s) on the bar chart
        default: None

    ecolor : scalar or array-like, optional
        specifies the color of errorbar(s)
        default: None

    capsize : scalar, optional
       determines the length in points of the error bar caps
       default: None, which will take the value from the
       ``errorbar.capsize`` :data:`rcParam<matplotlib.rcParams>`.

    error_kw : dict, optional
        dictionary of kwargs to be passed to errorbar method. *ecolor* and
        *capsize* may be specified here rather than as independent kwargs.
```

```
    align : {'edge',  'center'}, optional
        If 'edge', aligns bars by their left edges (for vertical bars) and
        by their bottom edges (for horizontal bars). If 'center', interpret
        the `left` argument as the coordinates of the centers of the bars.
        To align on the align bars on the right edge pass a negative
        `width`.

    orientation : {'vertical',  'horizontal'}, optional
        The orientation of the bars.

    log : boolean, optional
        If true, sets the axis to be log scale.
        default: False

    Returns
    -------
    bars : matplotlib.container.BarContainer
        Container with all of the bars + errorbars

    Notes
    -----
    The optional arguments `color`, `edgecolor`, `linewidth`,
    `xerr`, and `yerr` can be either scalars or sequences of
    length equal to the number of bars.  This enables you to use
    bar as the basis for stacked bar charts, or candlestick plots.
    Detail: `xerr` and `yerr` are passed directly to
    :meth:`errorbar`, so they can also have shape 2xN for
    independent specification of lower and upper errors.

    Other optional kwargs:

      agg_filter: unknown
      alpha: float or None
      animated: [True | False]
      antialiased or aa: [True | False]  or None for default
      axes: an :class:`~matplotlib.axes.Axes` instance
      capstyle: ['butt' | 'round' | 'projecting']
      clip_box: a :class:`matplotlib.transforms.Bbox` instance
      clip_on: [True | False]
      clip_path: [ (:class:`~matplotlib.path.Path`, :class:`~matplotlib.tran
sforms.Transform`) | :class:`~matplotlib.patches.Patch` | None ]
      color: matplotlib color spec
      contains: a callable function
      edgecolor or ec: mpl color spec, or None for default, or 'none' for no
 color
      facecolor or fc: mpl color spec, or None for default, or 'none' for no
 color
      figure: a :class:`matplotlib.figure.Figure` instance
      fill: [True | False]
      gid: an id string
      hatch: ['/' | '\\' | '|' | '-' | '+' | 'x' | 'o' | 'O' | '.' | '*']
      joinstyle: ['miter' | 'round' | 'bevel']
      label: string or anything printable with '%s' conversion.
      linestyle or ls: ['solid' | 'dashed', 'dashdot', 'dotted' | (offset, o
n-off-dash-seq) | ``'-'`` | ``'--'`` | ``'-.'`` | ``':'`` | ``'None'`` | ``'
 '`` | ``''``]
      linewidth or lw: float or None for default
      path_effects: unknown
      picker: [None|float|boolean|callable]
      rasterized: [True | False | None]
```

```
      sketch_params: unknown
      snap: unknown
      transform: :class:`~matplotlib.transforms.Transform` instance
      url: a url string
      visible: [True | False]
      zorder: any number

    See also
    --------
    barh: Plot a horizontal bar plot.

    Examples
    --------

    **Example:** A stacked bar chart.

    .. plot:: mpl_examples/pylab_examples/bar_stacked.py

    Notes
    -----

    In addition to the above described arguments, this function can take a
    **data** keyword argument. If such a **data** argument is given, the
    following arguments are replaced by **data[<arg>]**:

    * All arguments with the following names: 'height', 'linewidth', 'tick_l
abel', 'left', 'width', 'xerr', 'ecolor', 'bottom', 'yerr', 'edgecolor', 'co
lor'.




    Additional kwargs: hold = [True|False] overrides default hold state
```

In [35]:

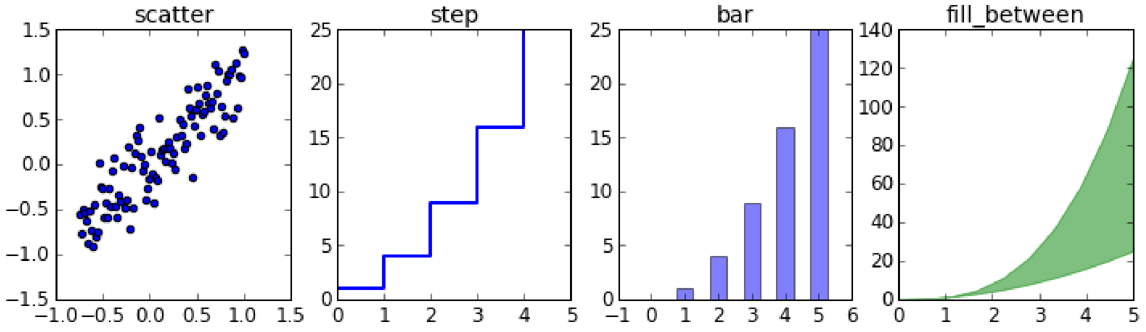
```
help(plt.fill_between)
```

```
Help on function fill_between in module matplotlib.pyplot:

fill_between(x, y1, y2=0, where=None, interpolate=False, step=None, hold=Non
e, data=None, **kwargs)
    Make filled polygons between two curves.


    Create a :class:`~matplotlib.collections.PolyCollection`
    filling the regions between *y1* and *y2* where
    ``where==True``

    Parameters
    ----------
    x : array
        An N-length array of the x data

    y1 : array
        An N-length array (or scalar) of the y data

    y2 : array
        An N-length array (or scalar) of the y data

    where : array, optional
        If `None`, default to fill between everywhere.  If not `None`,
        it is an N-length numpy boolean array and the fill will
        only happen over the regions where ``where==True``.

    interpolate : bool, optional
        If `True`, interpolate between the two lines to find the
        precise point of intersection.  Otherwise, the start and
        end points of the filled region will only occur on explicit
        values in the *x* array.

    step : {'pre', 'post', 'mid'}, optional
        If not None, fill with step logic.


    Notes
    -----

    Additional Keyword args passed on to the
    :class:`~matplotlib.collections.PolyCollection`.

    kwargs control the :class:`~matplotlib.patches.Polygon` properties:

      agg_filter: unknown
      alpha: float or None
      animated: [True | False]
      antialiased or antialiaseds: Boolean or sequence of booleans
      array: unknown
      axes: an :class:`~matplotlib.axes.Axes` instance
      clim: a length 2 sequence of floats
      clip_box: a :class:`matplotlib.transforms.Bbox` instance
      clip_on: [True | False]
      clip_path: [ (:class:`~matplotlib.path.Path`, :class:`~matplotlib.tran
sforms.Transform`) | :class:`~matplotlib.patches.Patch` | None ]
      cmap: a colormap or registered colormap name
      color: matplotlib color arg or sequence of rgba tuples
      contains: a callable function
      edgecolor or edgecolors: matplotlib color spec or sequence of specs
      facecolor or facecolors: matplotlib color spec or sequence of specs
```

```
      figure: a :class:`matplotlib.figure.Figure` instance
      gid: an id string
      hatch: [ '/' | '\\' | '|' | '-' | '+' | 'x' | 'o' | 'O' | '.' | '*' ]
      label: string or anything printable with '%s' conversion.
      linestyle or dashes or linestyles: ['solid' | 'dashed', 'dashdot', 'do
tted' | (offset, on-off-dash-seq) | ``'-'`` | ``'--'`` | ``'-.'`` | ``':'``
 | ``'None'`` | ``' '`` | ``''``]
      linewidth or linewidths or lw: float or sequence of floats
      norm: unknown
      offset_position: unknown
      offsets: float or sequence of floats
      path_effects: unknown
      picker: [None|float|boolean|callable]
      pickradius: unknown
      rasterized: [True | False | None]
      sketch_params: unknown
      snap: unknown
      transform: :class:`~matplotlib.transforms.Transform` instance
      url: a url string
      urls: unknown
      visible: [True | False]
      zorder: any number

    Examples
    --------

    .. plot:: mpl_examples/pylab_examples/fill_between_demo.py

    See Also
    --------

        :meth:`fill_betweenx`
            for filling between two sets of x-values

    Notes
    -----

    In addition to the above described arguments, this function can take a
    **data** keyword argument. If such a **data** argument is given, the
    following arguments are replaced by **data[<arg>]**:

    * All arguments with the following names: 'where', 'y1', 'y2', 'x'.




    Additional kwargs: hold = [True|False] overrides default hold state
```

In [36]:

```
n = np.array([0,1,2,3,4,5])

fig, axes = plt.subplots(1, 4, figsize=(12,3))

axes[0].scatter(xx, xx + 0.25*np.random.randn(len(xx)))
axes[0].set_title("scatter")

axes[1].step(n, n**2, lw=2)
axes[1].set_title("step")

axes[2].bar(n, n**2, align="center", width=0.5, alpha=0.5)
axes[2].set_title("bar")

axes[3].fill_between(x, x**2, x**3, color="green", alpha=0.5);
axes[3].set_title("fill_between");
```

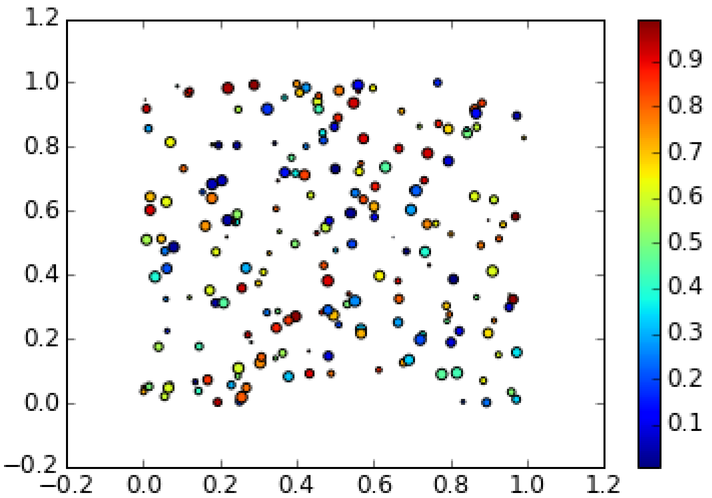


In [37]:

```
## Scatter
fig, ax = plt.subplots(1, 1)

x = np.random.rand(200)
y = np.random.rand(200)

size = np.random.rand(200)*30
color = np.random.rand(200)

im = ax.scatter(x, y, size, color)
fig.colorbar(im,ax=ax)
```

Out[37]:

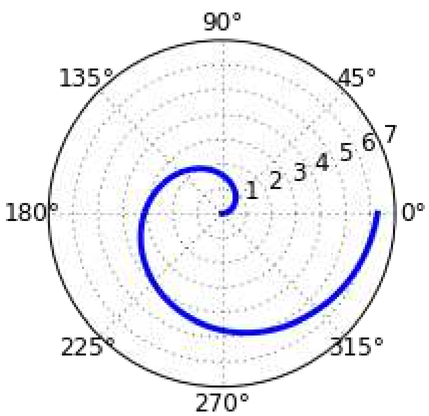
<matplotlib.colorbar.Colorbar at 0x1051a57f0>



极坐标

In [38]:

```
# polar plot using add_axes and polar projection
fig = plt.figure()
ax = fig.add_axes([0.0, 0.0, .6, .6], polar=True)
t = np.linspace(0, 2 * np.pi, 100)
ax.plot(t, t, color='blue', lw=3);
```



直方图

In [39]:

```
help(plt.hist)
```

```
Help on function hist in module matplotlib.pyplot:

hist(x, bins=10, range=None, normed=False, weights=None, cumulative=False, b
ottom=None, histtype='bar', align='mid', orientation='vertical', rwidth=Non
e, log=False, color=None, label=None, stacked=False, hold=None, data=None, *
*kwargs)
    Plot a histogram.

    Compute and draw the histogram of *x*. The return value is a
    tuple (*n*, *bins*, *patches*) or ([*n0*, *n1*, ...], *bins*,
    [*patches0*, *patches1*,...]) if the input contains multiple
    data.

    Multiple data can be provided via *x* as a list of datasets
    of potentially different length ([*x0*, *x1*, ...]), or as
    a 2-D ndarray in which each column is a dataset.  Note that
    the ndarray form is transposed relative to the list form.

    Masked arrays are not supported at present.

    Parameters
    ----------
    x : (n,) array or sequence of (n,) arrays
        Input values, this takes either a single array or a sequency of
        arrays which are not required to be of the same length

    bins : integer or array_like, optional
        If an integer is given, `bins + 1` bin edges are returned,
        consistently with :func:`numpy.histogram` for numpy version >=
        1.3.

        Unequally spaced bins are supported if `bins` is a sequence.

        default is 10

    range : tuple or None, optional
        The lower and upper range of the bins. Lower and upper outliers
        are ignored. If not provided, `range` is (x.min(), x.max()). Range
        has no effect if `bins` is a sequence.

        If `bins` is a sequence or `range` is specified, autoscaling
        is based on the specified bin range instead of the
        range of x.

        Default is ``None``

    normed : boolean, optional
        If `True`, the first element of the return tuple will
        be the counts normalized to form a probability density, i.e.,
        ``n/(len(x)`dbin)``, i.e., the integral of the histogram will sum
        to 1. If *stacked* is also *True*, the sum of the histograms is
        normalized to 1.

        Default is ``False``

    weights : (n, ) array_like or None, optional
        An array of weights, of the same shape as `x`.  Each value in `x`
        only contributes its associated weight towards the bin count
        (instead of 1).  If `normed` is True, the weights are normalized,
        so that the integral of the density over the range remains 1.
```

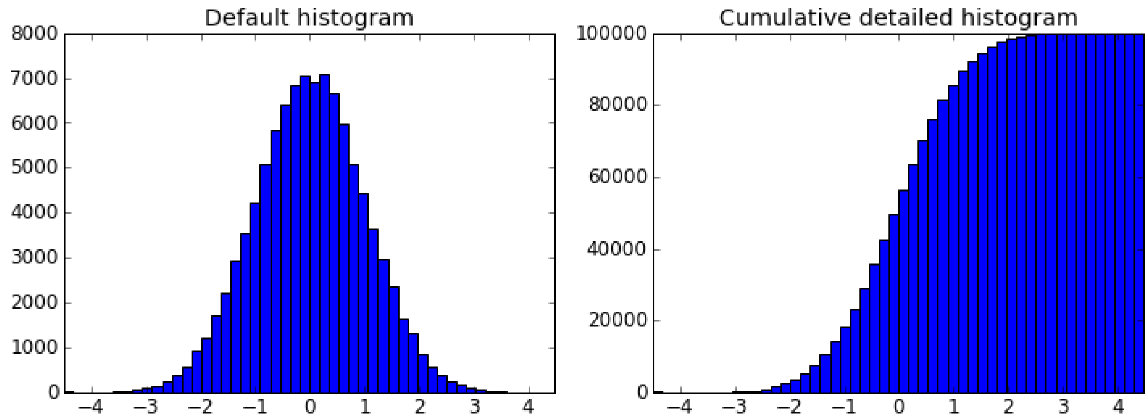
```
    Default is ``None``

cumulative : boolean, optional
    If `True`, then a histogram is computed where each bin gives the
    counts in that bin plus all bins for smaller values. The last bin
    gives the total number of datapoints.  If `normed` is also `True`
    then the histogram is normalized such that the last bin equals 1.
    If `cumulative` evaluates to less than 0 (e.g., -1), the direction
    of accumulation is reversed.  In this case, if `normed` is also
    `True`, then the histogram is normalized such that the first bin
    equals 1.

    Default is ``False``

bottom : array_like, scalar, or None
    Location of the bottom baseline of each bin.  If a scalar,
    the base line for each bin is shifted by the same amount.
    If an array, each bin is shifted independently and the length
    of bottom must match the number of bins.  If None, defaults to 0.

    Default is ``None``

histtype : {'bar', 'barstacked', 'step',  'stepfilled'}, optional
    The type of histogram to draw.

    - 'bar' is a traditional bar-type histogram.  If multiple data
      are given the bars are aranged side by side.

    - 'barstacked' is a bar-type histogram where multiple
      data are stacked on top of each other.

    - 'step' generates a lineplot that is by default
      unfilled.

    - 'stepfilled' generates a lineplot that is by default
      filled.

    Default is 'bar'

align : {'left', 'mid', 'right'}, optional
    Controls how the histogram is plotted.

        - 'left': bars are centered on the left bin edges.

        - 'mid': bars are centered between the bin edges.

        - 'right': bars are centered on the right bin edges.

    Default is 'mid'

orientation : {'horizontal', 'vertical'}, optional
    If 'horizontal', `~matplotlib.pyplot.barh` will be used for
    bar-type histograms and the *bottom* kwarg will be the left edges.

rwidth : scalar or None, optional
    The relative width of the bars as a fraction of the bin width.  If
    `None`, automatically compute the width.

    Ignored if `histtype` is 'step' or 'stepfilled'.

    Default is ``None``
```

```
log : boolean, optional
    If `True`, the histogram axis will be set to a log scale. If `log`
    is `True` and `x` is a 1D array, empty bins will be filtered out
    and only the non-empty (`n`, `bins`, `patches`) will be returned.

    Default is ``False``

color : color or array_like of colors or None, optional
    Color spec or sequence of color specs, one per dataset.  Default
    (`None`) uses the standard line color sequence.

    Default is ``None``

label : string or None, optional
    String, or sequence of strings to match multiple datasets.  Bar
    charts yield multiple patches per dataset, but only the first gets
    the label, so that the legend command will work as expected.

    default is ``None``

stacked : boolean, optional
    If `True`, multiple data are stacked on top of each other If
    `False` multiple data are aranged side by side if histtype is
    'bar' or on top of each other if histtype is 'step'

    Default is ``False``

Returns
-------
n : array or list of arrays
    The values of the histogram bins. See **normed** and **weights**
    for a description of the possible semantics. If input **x** is an
    array, then this is an array of length **nbins**. If input is a
    sequence arrays ``[data1, data2,..]``, then this is a list of
    arrays with the values of the histograms for each of the arrays
    in the same order.

bins : array
    The edges of the bins. Length nbins + 1 (nbins left edges and right
    edge of last bin).  Always a single array even when multiple data
    sets are passed in.

patches : list or list of lists
    Silent list of individual patches used to create the histogram
    or list of such list if multiple input datasets.

Other Parameters
----------------
kwargs : `~matplotlib.patches.Patch` properties

See also
--------
hist2d : 2D histograms

Notes
-----
Until numpy release 1.5, the underlying numpy histogram function was
incorrect with `normed`=`True` if bin sizes were unequal.  MPL
inherited that error.  It is now corrected within MPL when using
earlier numpy versions.
```

```
Examples
--------
.. plot:: mpl_examples/statistics/histogram_demo_features.py

Notes
-----

In addition to the above described arguments, this function can take a
**data** keyword argument. If such a **data** argument is given, the
following arguments are replaced by **data[<arg>]**:

* All arguments with the following names: 'weights', 'x'.



Additional kwargs: hold = [True|False] overrides default hold state
```

In [62]:

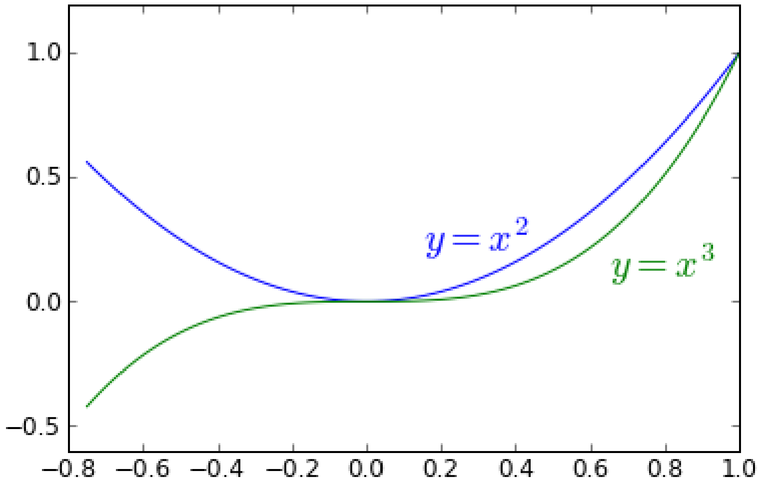
```
n = np.random.randn(100000)
fig, axes = plt.subplots(1, 2, figsize=(12,4))

axes[0].hist(n, bins=50)
axes[0].set_title("Default histogram")
axes[0].set_xlim((min(n), max(n)))

axes[1].hist(n, cumulative=True, bins=50)
axes[1].set_title("Cumulative detailed histogram")
axes[1].set_xlim((min(n), max(n)));
```



文字标注

In [41]:

```
fig, ax = plt.subplots()

ax.plot(xx, xx**2, xx, xx**3)

ax.text(0.15, 0.2, r"$y=x^2$", fontsize=20, color="blue")
ax.text(0.65, 0.1, r"$y=x^3$", fontsize=20, color="green");
```



多图模式

- subplot
- subplot2grid
- gridspec
- add_axes

subplot

In [42]:

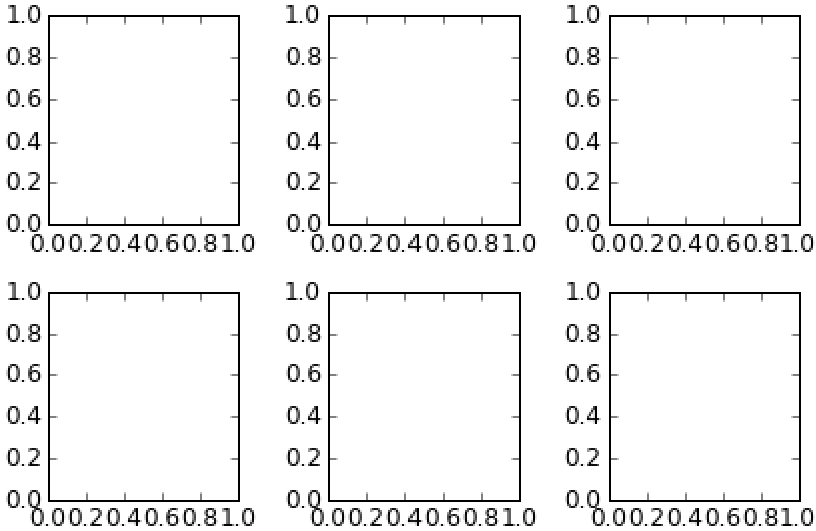
```
help(plt.subplots)
```

```
Help on function subplots in module matplotlib.pyplot:

subplots(nrows=1, ncols=1, sharex=False, sharey=False, squeeze=True, subplot
_kw=None, gridspec_kw=None, **fig_kw)
    Create a figure with a set of subplots already made.

    This utility wrapper makes it convenient to create common layouts of
    subplots, including the enclosing figure object, in a single call.

    Keyword arguments:

      *nrows* : int
        Number of rows of the subplot grid.  Defaults to 1.

      *ncols* : int
        Number of columns of the subplot grid.  Defaults to 1.

      *sharex* : string or bool
        If *True*, the X axis will be shared amongst all subplots.  If
        *True* and you have multiple rows, the x tick labels on all but
        the last row of plots will have visible set to *False*
        If a string must be one of "row", "col", "all", or "none".
        "all" has the same effect as *True*, "none" has the same effect
        as *False*.
        If "row", each subplot row will share a X axis.
        If "col", each subplot column will share a X axis and the x tick
        labels on all but the last row will have visible set to *False*.

      *sharey* : string or bool
        If *True*, the Y axis will be shared amongst all subplots. If
        *True* and you have multiple columns, the y tick labels on all but
        the first column of plots will have visible set to *False*
        If a string must be one of "row", "col", "all", or "none".
        "all" has the same effect as *True*, "none" has the same effect
        as *False*.
        If "row", each subplot row will share a Y axis and the y tick
        labels on all but the first column will have visible set to *False*.
        If "col", each subplot column will share a Y axis.

      *squeeze* : bool
        If *True*, extra dimensions are squeezed out from the
        returned axis object:

        - if only one subplot is constructed (nrows=ncols=1), the
          resulting single Axis object is returned as a scalar.

        - for Nx1 or 1xN subplots, the returned object is a 1-d numpy
          object array of Axis objects are returned as numpy 1-d
          arrays.

        - for NxM subplots with N>1 and M>1 are returned as a 2d
          array.

        If *False*, no squeezing at all is done: the returned axis
        object is always a 2-d array containing Axis instances, even if it
        ends up being 1x1.

      *subplot_kw* : dict
        Dict with keywords passed to the
        :meth:`~matplotlib.figure.Figure.add_subplot` call used to
        create each subplots.
```

```
  *gridspec_kw* : dict
    Dict with keywords passed to the
    :class:`~matplotlib.gridspec.GridSpec` constructor used to create
    the grid the subplots are placed on.

  *fig_kw* : dict
    Dict with keywords passed to the :func:`figure` call.  Note that all
    keywords not recognized above will be automatically included here.

Returns:

fig, ax : tuple

  - *fig* is the :class:`matplotlib.figure.Figure` object

  - *ax* can be either a single axis object or an array of axis
    objects if more than one subplot was created.  The dimensions
    of the resulting array can be controlled with the squeeze
    keyword, see above.

Examples::

    x = np.linspace(0, 2*np.pi, 400)
    y = np.sin(x**2)

    # Just a figure and one subplot
    f, ax = plt.subplots()
    ax.plot(x, y)
    ax.set_title('Simple plot')

    # Two subplots, unpack the output array immediately
    f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)
    ax1.plot(x, y)
    ax1.set_title('Sharing Y axis')
    ax2.scatter(x, y)

    # Four polar axes
    plt.subplots(2, 2, subplot_kw=dict(polar=True))

    # Share a X axis with each column of subplots
    plt.subplots(2, 2, sharex='col')

    # Share a Y axis with each row of subplots
    plt.subplots(2, 2, sharey='row')

    # Share a X and Y axis with all subplots
    plt.subplots(2, 2, sharex='all', sharey='all')
    # same as
    plt.subplots(2, 2, sharex=True, sharey=True)
```

In [43]:

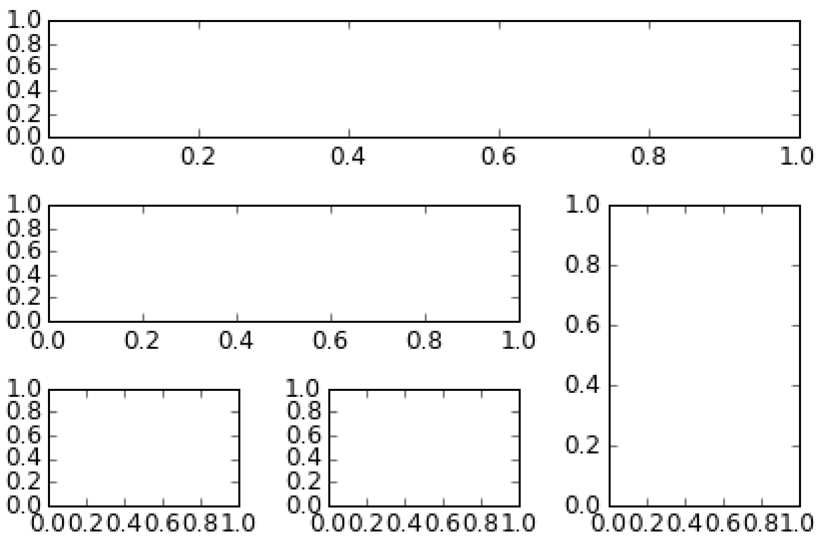
```
fig, ax = plt.subplots(2, 3)
fig.tight_layout()
```



subplot2grid

In [44]:

```
fig = plt.figure()
ax1 = plt.subplot2grid((3,3), (0,0), colspan=3)
ax2 = plt.subplot2grid((3,3), (1,0), colspan=2)
ax3 = plt.subplot2grid((3,3), (1,2), rowspan=2)
ax4 = plt.subplot2grid((3,3), (2,0))
ax5 = plt.subplot2grid((3,3), (2,1))
fig.tight_layout()
```



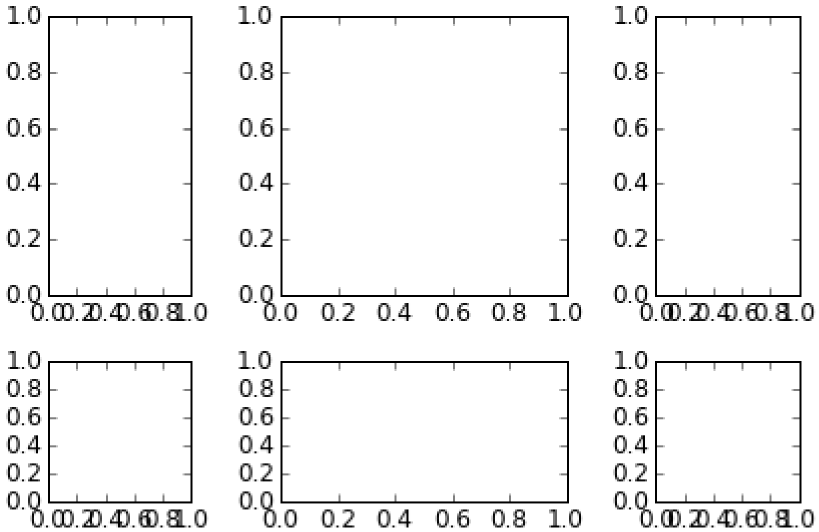
gridspec

In [45]:

```
import matplotlib.gridspec as gridspec
```

In [46]:

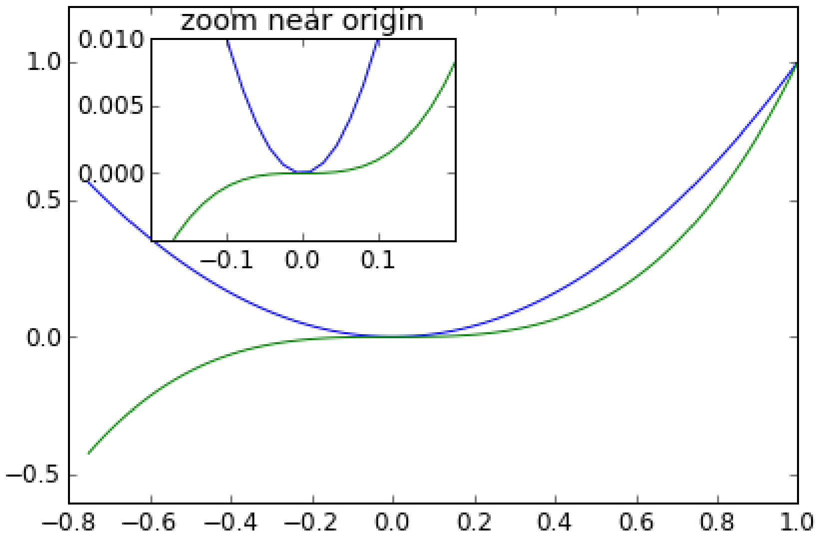
```
fig = plt.figure()

gs = gridspec.GridSpec(2, 3, height_ratios=[2,1], width_ratios=[1,2,1])
for g in gs:
    ax = fig.add_subplot(g)

fig.tight_layout()
```



gridspec

In [47]:

```
fig, ax = plt.subplots()

ax.plot(xx, xx**2, xx, xx**3)
fig.tight_layout()

# inset
inset_ax = fig.add_axes([0.2, 0.55, 0.35, 0.35]) # X, Y, width, height

inset_ax.plot(xx, xx**2, xx, xx**3)
inset_ax.set_title('zoom near origin')

# set axis range
inset_ax.set_xlim(-.2, .2)
inset_ax.set_ylim(-.005, .01)

# set axis tick locations
inset_ax.set_yticks([0, 0.005, 0.01])
inset_ax.set_xticks([-0.1,0,.1]);
```



颜色图和等高线图

[http://www.scipy.org/Cookbook/Matplotlib/Show_colormaps (http://www.scipy.org/Cookbook/Matplotlib/Show_colormaps)](http://www.scipy.org/Cookbook/Matplotlib/Show_colormaps)

In [48]:

```
alpha = 0.7
phi_ext = 2 * np.pi * 0.5

def flux_qubit_potential(phi_m, phi_p):
    return 2 + alpha - 2 * np.cos(phi_p) * np.cos(phi_m) - alpha * np.cos(phi_ext - 2*phi
_p)
```

In [49]:

```
phi_m = np.linspace(0, 2*np.pi, 100)
phi_p = np.linspace(0, 2*np.pi, 100)
X,Y = np.meshgrid(phi_p, phi_m)
Z = flux_qubit_potential(X, Y).T
```

pcolor

In [50]:

```
help(plt.pcolor)
```

```
Help on function pcolor in module matplotlib.pyplot:

pcolor(*args, **kwargs)
    Create a pseudocolor plot of a 2-D array.

    .. note::

        pcolor can be very slow for large arrays; consider
        using the similar but much faster
        :func:`~matplotlib.pyplot.pcolormesh` instead.

    Call signatures::

      pcolor(C, **kwargs)
      pcolor(X, Y, C, **kwargs)

    *C* is the array of color values.

    *X* and *Y*, if given, specify the (*x*, *y*) coordinates of
    the colored quadrilaterals; the quadrilateral for C[i,j] has
    corners at::

      (X[i,   j],   Y[i,   j]),
      (X[i,   j+1], Y[i,   j+1]),
      (X[i+1, j],   Y[i+1, j]),
      (X[i+1, j+1], Y[i+1, j+1]).

    Ideally the dimensions of *X* and *Y* should be one greater
    than those of *C*; if the dimensions are the same, then the
    last row and column of *C* will be ignored.

    Note that the column index corresponds to the
    *x*-coordinate, and the row index corresponds to *y*; for
    details, see the :ref:`Grid Orientation
    <axes-pcolor-grid-orientation>` section below.

    If either or both of *X* and *Y* are 1-D arrays or column vectors,
    they will be expanded as needed into the appropriate 2-D arrays,
    making a rectangular grid.

    *X*, *Y* and *C* may be masked arrays.  If either C[i, j], or one
    of the vertices surrounding C[i,j] (*X* or *Y* at [i, j], [i+1, j],
    [i, j+1],[i+1, j+1]) is masked, nothing is plotted.

    Keyword arguments:

      *cmap*: [ *None* | Colormap ]
        A :class:`matplotlib.colors.Colormap` instance. If *None*, use
        rc settings.

      *norm*: [ *None* | Normalize ]
        An :class:`matplotlib.colors.Normalize` instance is used
        to scale luminance data to 0,1. If *None*, defaults to
        :func:`normalize`.

      *vmin*/*vmax*: [ *None* | scalar ]
        *vmin* and *vmax* are used in conjunction with *norm* to
        normalize luminance data.  If either is *None*, it
        is autoscaled to the respective min or max
        of the color array *C*.  If not *None*, *vmin* or
        *vmax* passed in here override any pre-existing values
```

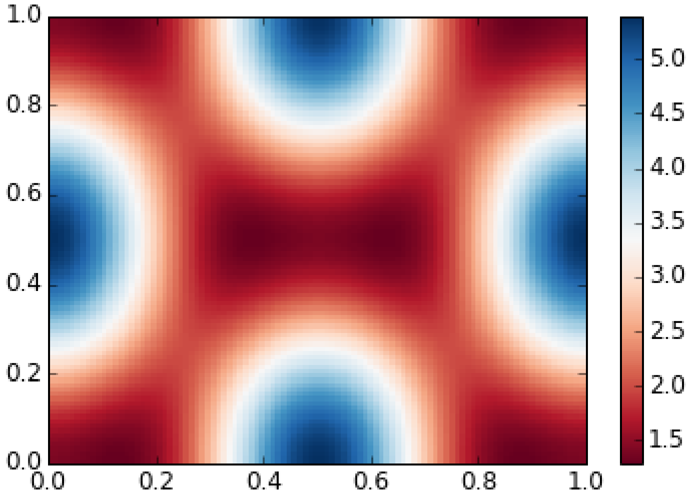
```
    supplied in the *norm* instance.

  *shading*: [ 'flat' | 'faceted' ]
    If 'faceted', a black grid is drawn around each rectangle; if
    'flat', edges are not drawn. Default is 'flat', contrary to
    MATLAB.

    This kwarg is deprecated; please use 'edgecolors' instead:
      * shading='flat' -- edgecolors='none'
      * shading='faceted  -- edgecolors='k'

  *edgecolors*: [ *None* | ``'none'`` | color | color sequence]
    If *None*, the rc setting is used by default.

    If ``'none'``, edges will not be visible.

    An mpl color or sequence of colors will set the edge color

  *alpha*: ``0 <= scalar <= 1``   or *None*
    the alpha blending value

  *snap*: bool
    Whether to snap the mesh to pixel boundaries.

Return value is a :class:`matplotlib.collections.Collection`
instance.

.. _axes-pcolor-grid-orientation:

The grid orientation follows the MATLAB convention: an
array *C* with shape (*nrows*, *ncolumns*) is plotted with
the column number as *X* and the row number as *Y*, increasing
up; hence it is plotted the way the array would be printed,
except that the *Y* axis is reversed.  That is, *C* is taken
as *C*(*y*, *x*).

Similarly for :func:`meshgrid`::

  x = np.arange(5)
  y = np.arange(3)
  X, Y = np.meshgrid(x, y)

is equivalent to::

  X = array([[0, 1, 2, 3, 4],
             [0, 1, 2, 3, 4],
             [0, 1, 2, 3, 4]])

  Y = array([[0, 0, 0, 0, 0],
             [1, 1, 1, 1, 1],
             [2, 2, 2, 2, 2]])

so if you have::

  C = rand(len(x), len(y))

then you need to transpose C::

  pcolor(X, Y, C.T)

or::
```

```
      pcolor(C.T)

    MATLAB :func:`pcolor` always discards the last row and column
    of *C*, but matplotlib displays the last row and column if *X* and
    *Y* are not specified, or if *X* and *Y* have one more row and
    column than *C*.

    kwargs can be used to control the
    :class:`~matplotlib.collections.PolyCollection` properties:

      agg_filter: unknown
      alpha: float or None
      animated: [True | False]
      antialiased or antialiaseds: Boolean or sequence of booleans
      array: unknown
      axes: an :class:`~matplotlib.axes.Axes` instance
      clim: a length 2 sequence of floats
      clip_box: a :class:`matplotlib.transforms.Bbox` instance
      clip_on: [True | False]
      clip_path: [ (:class:`~matplotlib.path.Path`, :class:`~matplotlib.tran
sforms.Transform`) | :class:`~matplotlib.patches.Patch` | None ]
      cmap: a colormap or registered colormap name
      color: matplotlib color arg or sequence of rgba tuples
      contains: a callable function
      edgecolor or edgecolors: matplotlib color spec or sequence of specs
      facecolor or facecolors: matplotlib color spec or sequence of specs
      figure: a :class:`matplotlib.figure.Figure` instance
      gid: an id string
      hatch: [ '/' | '\\' | '|' | '-' | '+' | 'x' | 'o' | 'O' | '.' | '*' ]
      label: string or anything printable with '%s' conversion.
      linestyle or dashes or linestyles: ['solid' | 'dashed', 'dashdot', 'do
tted' | (offset, on-off-dash-seq) | ``'-'`` | ``'--'`` | ``'-.'`` | ``':'``
 | ``'None'`` | ``' '`` | ``''``]
      linewidth or linewidths or lw: float or sequence of floats
      norm: unknown
      offset_position: unknown
      offsets: float or sequence of floats
      path_effects: unknown
      picker: [None|float|boolean|callable]
      pickradius: unknown
      rasterized: [True | False | None]
      sketch_params: unknown
      snap: unknown
      transform: :class:`~matplotlib.transforms.Transform` instance
      url: a url string
      urls: unknown
      visible: [True | False]
      zorder: any number

    .. note::

        The default *antialiaseds* is False if the default
        *edgecolors*="none" is used.  This eliminates artificial lines
        at patch boundaries, and works regardless of the value of
        alpha.  If *edgecolors* is not "none", then the default
        *antialiaseds* is taken from
        rcParams['patch.antialiased'], which defaults to *True*.
        Stroking the edges may be preferred if *alpha* is 1, but
        will cause artifacts otherwise.
```

```
.. seealso::

    :func:`~matplotlib.pyplot.pcolormesh`
        For an explanation of the differences between
        pcolor and pcolormesh.

Notes
-----

In addition to the above described arguments, this function can take a
**data** keyword argument. If such a **data** argument is given, the
following arguments are replaced by **data[<arg>]**:

* All positional and all keyword arguments.




Additional kwargs: hold = [True|False] overrides default hold state
```

In [51]:

```
fig, ax = plt.subplots()

p = ax.pcolor(X/(2*np.pi), Y/(2*np.pi), Z, cmap=matplotlib.cm.RdBu, vmin=abs(Z).min(), vm
ax=abs(Z).max())
cb = fig.colorbar(p, ax=ax)
```



imshow

In [52]:

```
help(plt.imshow)
```

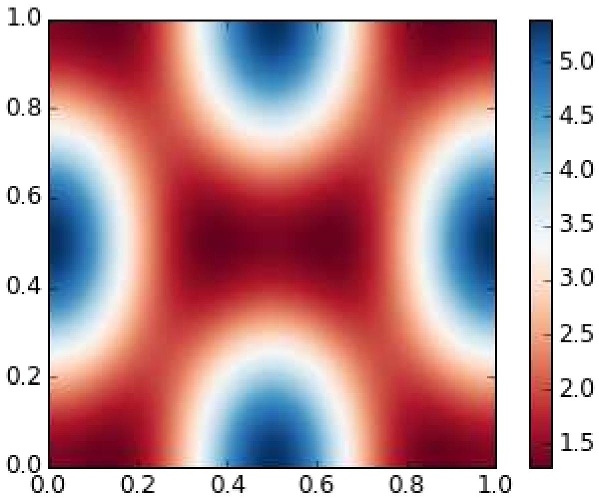
```
Help on function imshow in module matplotlib.pyplot:

imshow(X, cmap=None, norm=None, aspect=None, interpolation=None, alpha=None,
 vmin=None, vmax=None, origin=None, extent=None, shape=None, filternorm=1, f
ilterrad=4.0, imlim=None, resample=None, url=None, hold=None, data=None, **k
wargs)
    Display an image on the axes.

    Parameters
    -----------
    X : array_like, shape (n, m) or (n, m, 3) or (n, m, 4)
        Display the image in `X` to current axes.  `X` may be a float
        array, a uint8 array or a PIL image. If `X` is an array, it
        can have the following shapes:

        - MxN -- luminance (grayscale, float array only)
        - MxNx3 -- RGB (float or uint8 array)
        - MxNx4 -- RGBA (float or uint8 array)

        The value for each component of MxNx3 and MxNx4 float arrays
        should be in the range 0.0 to 1.0; MxN float arrays may be
        normalised.

    cmap : `~matplotlib.colors.Colormap`, optional, default: None
        If None, default to rc `image.cmap` value. `cmap` is ignored when
        `X` has RGB(A) information

    aspect : ['auto' | 'equal' | scalar], optional, default: None
        If 'auto', changes the image aspect ratio to match that of the
        axes.

        If 'equal', and `extent` is None, changes the axes aspect ratio to
        match that of the image. If `extent` is not `None`, the axes
        aspect ratio is changed to match that of the extent.

        If None, default to rc ``image.aspect`` value.

    interpolation : string, optional, default: None
        Acceptable values are 'none', 'nearest', 'bilinear', 'bicubic',
        'spline16', 'spline36', 'hanning', 'hamming', 'hermite', 'kaiser',
        'quadric', 'catrom', 'gaussian', 'bessel', 'mitchell', 'sinc',
        'lanczos'

        If `interpolation` is None, default to rc `image.interpolation`.
        See also the `filternorm` and `filterrad` parameters.
        If `interpolation` is 'none', then no interpolation is performed
        on the Agg, ps and pdf backends. Other backends will fall back to
        'nearest'.

    norm : `~matplotlib.colors.Normalize`, optional, default: None
        A `~matplotlib.colors.Normalize` instance is used to scale
        luminance data to 0, 1. If `None`, use the default
        func:`normalize`. `norm` is only used if `X` is an array of
        floats.

    vmin, vmax : scalar, optional, default: None
        `vmin` and `vmax` are used in conjunction with norm to normalize
        luminance data.  Note if you pass a `norm` instance, your
        settings for `vmin` and `vmax` will be ignored.

    alpha : scalar, optional, default: None
```

```
    The alpha blending value, between 0 (transparent) and 1 (opaque)

origin : ['upper' | 'lower'], optional, default: None
    Place the [0,0] index of the array in the upper left or lower left
    corner of the axes. If None, default to rc `image.origin`.

extent : scalars (left, right, bottom, top), optional, default: None
    The location, in data-coordinates, of the lower-left and
    upper-right corners. If `None`, the image is positioned such that
    the pixel centers fall on zero-based (row, column) indices.

shape : scalars (columns, rows), optional, default: None
    For raw buffer images

filternorm : scalar, optional, default: 1
    A parameter for the antigrain image resize filter.  From the
    antigrain documentation, if `filternorm` = 1, the filter
    normalizes integer values and corrects the rounding errors. It
    doesn't do anything with the source floating point values, it
    corrects only integers according to the rule of 1.0 which means
    that any sum of pixel weights must be equal to 1.0.  So, the
    filter function must produce a graph of the proper shape.

filterrad : scalar, optional, default: 4.0
    The filter radius for filters that have a radius parameter, i.e.
    when interpolation is one of: 'sinc', 'lanczos' or 'blackman'

Returns
--------
image : `~matplotlib.image.AxesImage`

Other parameters
----------------
kwargs : `~matplotlib.artist.Artist` properties.

See also
--------
matshow : Plot a matrix or an array as an image.

Notes
-----
Unless *extent* is used, pixel centers will be located at integer
coordinates. In other words: the origin will coincide with the center
of pixel (0, 0).

Examples
--------

.. plot:: mpl_examples/pylab_examples/image_demo.py

Notes
-----

In addition to the above described arguments, this function can take a
**data** keyword argument. If such a **data** argument is given, the
following arguments are replaced by **data[<arg>]**:

* All positional and all keyword arguments.
```

```
Additional kwargs: hold = [True|False] overrides default hold state
```

In [53]:

```
fig, ax = plt.subplots()

im = ax.imshow(Z, cmap=matplotlib.cm.RdBu, vmin=abs(Z).min(), vmax=abs(Z).max(), extent=
[0, 1, 0, 1])
im.set_interpolation('bilinear')

cb = fig.colorbar(im, ax=ax)
```



contour

In [54]:

```
help(plt.contour)
```

```
Help on function contour in module matplotlib.pyplot:

contour(*args, **kwargs)
    Plot contours.

    :func:`~matplotlib.pyplot.contour` and
    :func:`~matplotlib.pyplot.contourf` draw contour lines and
    filled contours, respectively.  Except as noted, function
    signatures and return values are the same for both versions.

    :func:`~matplotlib.pyplot.contourf` differs from the MATLAB
    version in that it does not draw the polygon edges.
    To draw edges, add line contours with
    calls to :func:`~matplotlib.pyplot.contour`.


    Call signatures::

      contour(Z)

    make a contour plot of an array *Z*. The level values are chosen
    automatically.

    ::

      contour(X,Y,Z)

    *X*, *Y* specify the (x, y) coordinates of the surface

    ::

      contour(Z,N)
      contour(X,Y,Z,N)

    contour up to *N* automatically-chosen levels.

    ::

      contour(Z,V)
      contour(X,Y,Z,V)

    draw contour lines at the values specified in sequence *V*,
    which must be in increasing order.

    ::

      contourf(..., V)

    fill the ``len(V)-1`` regions between the values in *V*,
    which must be in increasing order.

    ::

      contour(Z, **kwargs)

    Use keyword args to control colors, linewidth, origin, cmap ... see
    below for more details.

    *X* and *Y* must both be 2-D with the same shape as *Z*, or they
    must both be 1-D such that ``len(X)`` is the number of columns in
    *Z* and ``len(Y)`` is the number of rows in *Z*.
```

```
``C = contour(...)`` returns a
:class:`~matplotlib.contour.QuadContourSet` object.

Optional keyword arguments:

  *corner_mask*: [ *True* | *False* | 'legacy' ]
    Enable/disable corner masking, which only has an effect if *Z* is
    a masked array.  If *False*, any quad touching a masked point is
    masked out.  If *True*, only the triangular corners of quads
    nearest those points are always masked out, other triangular
    corners comprising three unmasked points are contoured as usual.
    If 'legacy', the old contouring algorithm is used, which is
    equivalent to *False* and is deprecated, only remaining whilst the
    new algorithm is tested fully.

    If not specified, the default is taken from
    rcParams['contour.corner_mask'], which is True unless it has
    been modified.

  *colors*: [ *None* | string | (mpl_colors) ]
    If *None*, the colormap specified by cmap will be used.

    If a string, like 'r' or 'red', all levels will be plotted in this
    color.

    If a tuple of matplotlib color args (string, float, rgb, etc),
    different levels will be plotted in different colors in the order
    specified.

  *alpha*: float
    The alpha blending value

  *cmap*: [ *None* | Colormap ]
    A cm :class:`~matplotlib.colors.Colormap` instance or
    *None*. If *cmap* is *None* and *colors* is *None*, a
    default Colormap is used.

  *norm*: [ *None* | Normalize ]
    A :class:`matplotlib.colors.Normalize` instance for
    scaling data values to colors. If *norm* is *None* and
    *colors* is *None*, the default linear scaling is used.

  *vmin*, *vmax*: [ *None* | scalar ]
    If not *None*, either or both of these values will be
    supplied to the :class:`matplotlib.colors.Normalize`
    instance, overriding the default color scaling based on
    *levels*.

  *levels*: [level0, level1, ..., leveln]
    A list of floating point numbers indicating the level
    curves to draw, in increasing order; e.g., to draw just
    the zero contour pass ``levels=[0]``

  *origin*: [ *None* | 'upper' | 'lower' | 'image' ]
    If *None*, the first value of *Z* will correspond to the
    lower left corner, location (0,0). If 'image', the rc
    value for ``image.origin`` will be used.

    This keyword is not active if *X* and *Y* are specified in
    the call to contour.
```

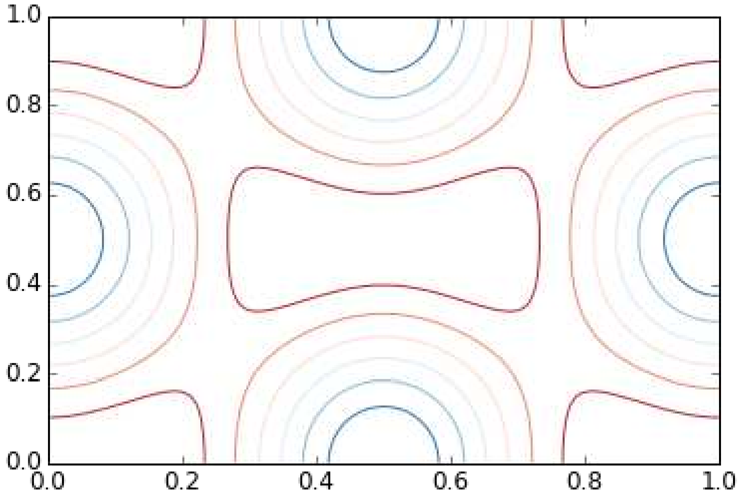
```
*extent*: [ *None* | (x0,x1,y0,y1) ]

  If *origin* is not *None*, then *extent* is interpreted as
  in :func:`matplotlib.pyplot.imshow`: it gives the outer
  pixel boundaries. In this case, the position of Z[0,0]
  is the center of the pixel, not a corner. If *origin* is
  *None*, then (*x0*, *y0*) is the position of Z[0,0], and
  (*x1*, *y1*) is the position of Z[-1,-1].

  This keyword is not active if *X* and *Y* are specified in
  the call to contour.

*locator*: [ *None* | ticker.Locator subclass ]
  If *locator* is *None*, the default
  :class:`~matplotlib.ticker.MaxNLocator` is used. The
  locator is used to determine the contour levels if they
  are not given explicitly via the *V* argument.

*extend*: [ 'neither' | 'both' | 'min' | 'max' ]
  Unless this is 'neither', contour levels are automatically
  added to one or both ends of the range so that all data
  are included. These added ranges are then mapped to the
  special colormap values which default to the ends of the
  colormap range, but can be set via
  :meth:`matplotlib.colors.Colormap.set_under` and
  :meth:`matplotlib.colors.Colormap.set_over` methods.

*xunits*, *yunits*: [ *None* | registered units ]
  Override axis units by specifying an instance of a
  :class:`matplotlib.units.ConversionInterface`.

*antialiased*: [ *True* | *False* ]
  enable antialiasing, overriding the defaults.  For
  filled contours, the default is *True*.  For line contours,
  it is taken from rcParams['lines.antialiased'].

*nchunk*: [ 0 | integer ]
  If 0, no subdivision of the domain.  Specify a positive integer to
  divide the domain into subdomains of *nchunk* by *nchunk* quads.
  Chunking reduces the maximum length of polygons generated by the
  contouring algorithm which reduces the rendering workload passed
  on to the backend and also requires slightly less RAM.  It can
  however introduce rendering artifacts at chunk boundaries depending
  on the backend, the *antialiased* flag and value of *alpha*.
```

```
contour-only keyword arguments:

  *linewidths*: [ *None* | number | tuple of numbers ]
    If *linewidths* is *None*, the default width in
    ``lines.linewidth`` in ``matplotlibrc`` is used.

    If a number, all levels will be plotted with this linewidth.

    If a tuple, different levels will be plotted with different
    linewidths in the order specified.

  *linestyles*: [ *None* | 'solid' | 'dashed' | 'dashdot' | 'dotted' ]
    If *linestyles* is *None*, the default is 'solid' unless
    the lines are monochrome.  In that case, negative
    contours will take their linestyle from the ``matplotlibrc``
```

```
        ``contour.negative_linestyle`` setting.

        *linestyles* can also be an iterable of the above strings
        specifying a set of linestyles to be used. If this
        iterable is shorter than the number of contour levels
        it will be repeated as necessary.

    contourf-only keyword arguments:

      *hatches*:
        A list of cross hatch patterns to use on the filled areas.
        If None, no hatching will be added to the contour.
        Hatching is supported in the PostScript, PDF, SVG and Agg
        backends only.


    Note: contourf fills intervals that are closed at the top; that
    is, for boundaries *z1* and *z2*, the filled region is::

        z1 < z <= z2

    There is one exception: if the lowest boundary coincides with
    the minimum value of the *z* array, then that minimum value
    will be included in the lowest interval.

    **Examples:**

    .. plot:: mpl_examples/pylab_examples/contour_demo.py

    .. plot:: mpl_examples/pylab_examples/contourf_demo.py

    .. plot:: mpl_examples/pylab_examples/contour_corner_mask.py



    Additional kwargs: hold = [True|False] overrides default hold state
```

In [55]:

```
fig, ax = plt.subplots()

cnt = ax.contour(Z, cmap=matplotlib.cm.RdBu, vmin=abs(Z).min(), vmax=abs(Z).max(),
extent=[0, 1, 0, 1])
```



3D

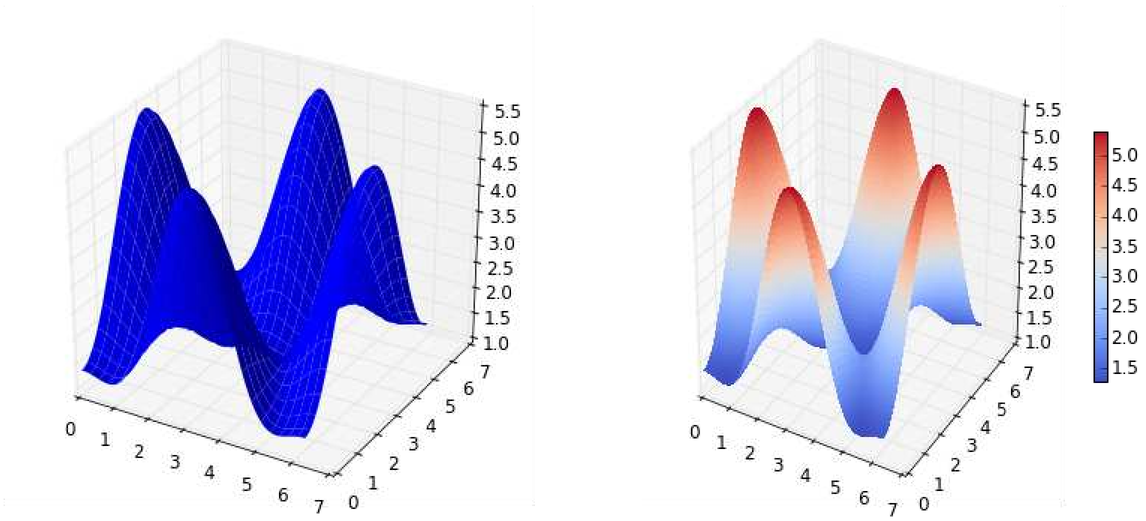
In [56]:

```
from mpl_toolkits.mplot3d.axes3d import Axes3D
```

Surface plot

In [57]:

```
fig = plt.figure(figsize=(14,6))

# `ax` is a 3D-aware axis instance because of the projection='3d' keyword argument to add
_subplot
ax = fig.add_subplot(1, 2, 1, projection='3d')

p = ax.plot_surface(X, Y, Z, rstride=4, cstride=4, linewidth=0)

# surface_plot with color grading and color bar
ax = fig.add_subplot(1, 2, 2, projection='3d')
p = ax.plot_surface(X, Y, Z, rstride=1, cstride=1, cmap=matplotlib.cm.coolwarm,
linewidth=0, antialiased=False)
cb = fig.colorbar(p, shrink=0.5)
```



In [58]:

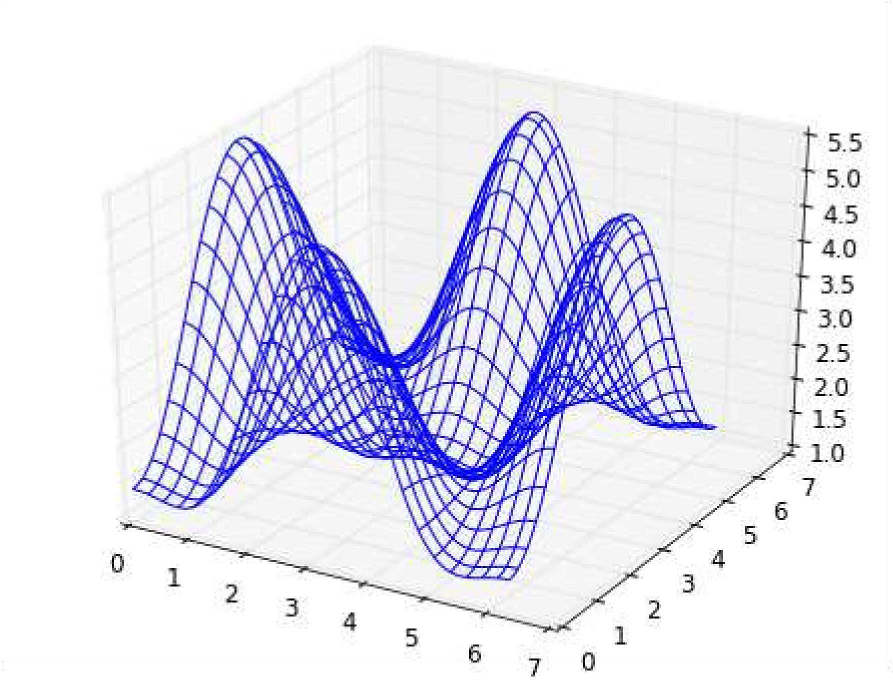
```
help(ax.plot_surface)
```

```
Help on method plot_surface in module mpl_toolkits.mplot3d.axes3d:

plot_surface(X, Y, Z, *args, **kwargs) method of matplotlib.axes._subplots.A
xes3DSubplot instance
    Create a surface plot.

    By default it will be colored in shades of a solid color,
    but it also supports color mapping by supplying the *cmap*
    argument.

    The `rstride` and `cstride` kwargs set the stride used to
    sample the input data to generate the graph.  If 1k by 1k
    arrays are passed in the default values for the strides will
    result in a 100x100 grid being plotted.

    ============= ================================================
    Argument      Description
    ============= ================================================
    *X*, *Y*, *Z* Data values as 2D arrays
    *rstride*     Array row stride (step size), defaults to 10
    *cstride*     Array column stride (step size), defaults to 10
    *color*       Color of the surface patches
    *cmap*        A colormap for the surface patches.
    *facecolors*  Face colors for the individual patches
    *norm*        An instance of Normalize to map values to colors
    *vmin*        Minimum value to map
    *vmax*        Maximum value to map
    *shade*       Whether to shade the facecolors
    ============= ================================================

    Other arguments are passed on to
    :class:`~mpl_toolkits.mplot3d.art3d.Poly3DCollection`
```

Wire-frame plot

In [59]:

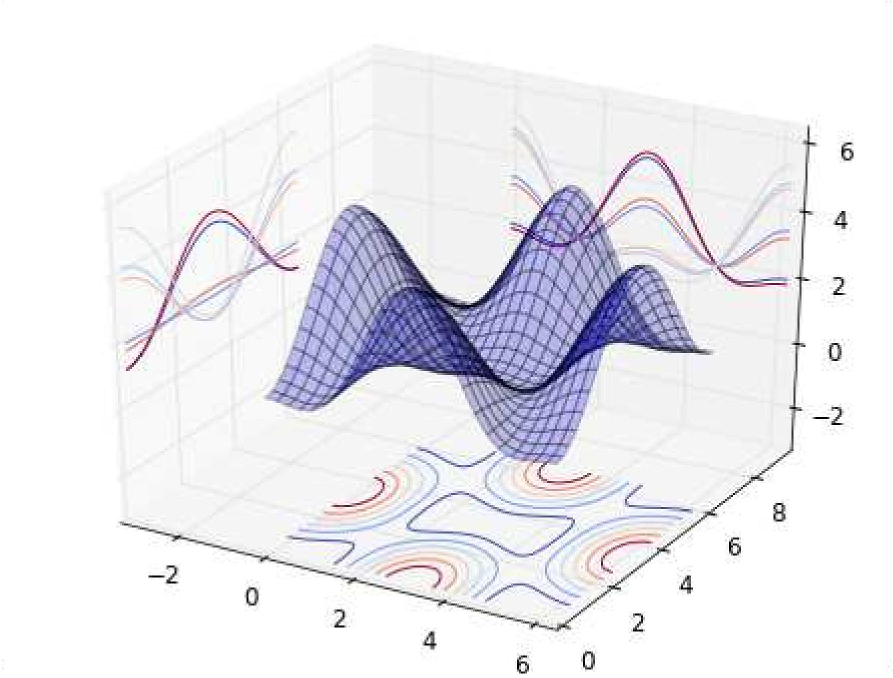
```
fig = plt.figure(figsize=(8,6))

ax = fig.add_subplot(1, 1, 1, projection='3d')

p = ax.plot_wireframe(X, Y, Z, rstride=4, cstride=4)
```



复杂点的

In [60]:

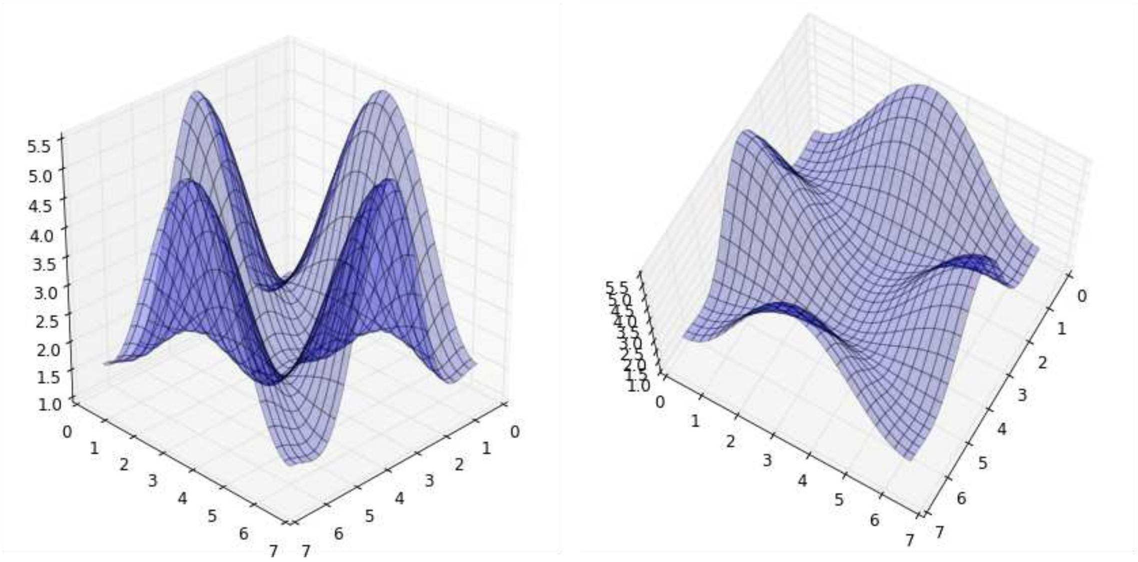
```
fig = plt.figure(figsize=(8,6))

ax = fig.add_subplot(1,1,1, projection='3d')

ax.plot_surface(X, Y, Z, rstride=4, cstride=4, alpha=0.25)
cset = ax.contour(X, Y, Z, zdir='z', offset=-np.pi, cmap=matplotlib.cm.coolwarm)
cset = ax.contour(X, Y, Z, zdir='x', offset=-np.pi, cmap=matplotlib.cm.coolwarm)
cset = ax.contour(X, Y, Z, zdir='y', offset=3*np.pi, cmap=matplotlib.cm.coolwarm)

ax.set_xlim3d(-np.pi, 2*np.pi);
ax.set_ylim3d(0, 3*np.pi);
ax.set_zlim3d(-np.pi, 2*np.pi);
```



换个角度

In [61]:

```
fig = plt.figure(figsize=(12,6))

ax = fig.add_subplot(1,2,1, projection='3d')
ax.plot_surface(X, Y, Z, rstride=4, cstride=4, alpha=0.25)
ax.view_init(30, 45)

ax = fig.add_subplot(1,2,2, projection='3d')
ax.plot_surface(X, Y, Z, rstride=4, cstride=4, alpha=0.25)
ax.view_init(70, 30)

fig.tight_layout()
```



重要参考

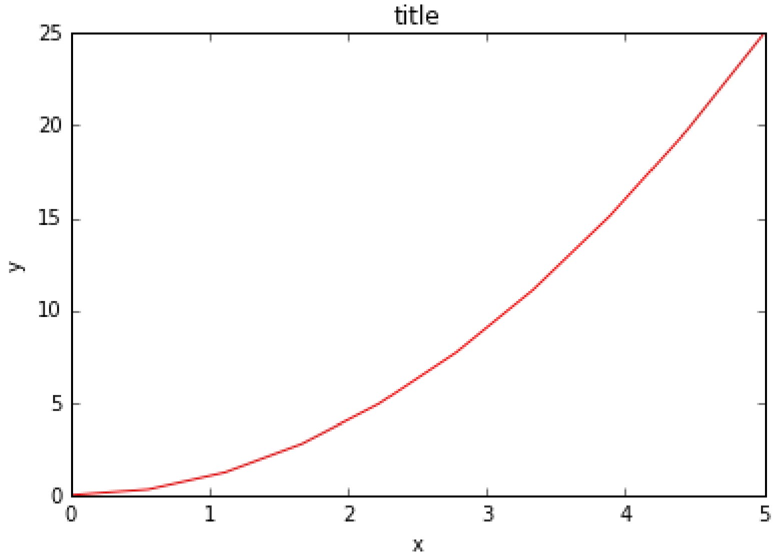
- [初学者指南 (http://matplotlib.org/users/beginner.html)](http://matplotlib.org/users/beginner.html)
- [高级指南 (http://matplotlib.org/users/developer.html)](http://matplotlib.org/users/developer.html)

In [6]:

```
%matplotlib inline
import numpy as np

import matplotlib
import matplotlib.pyplot as plt
```

In [7]:

```
x = np.linspace(0, 5, 10)
y = x ** 2

fig = plt.figure()

axes = fig.add_axes([0.1, 0.1, 0.8, 0.8]) # left, bottom, width, height (range 0 to 1)

axes.plot(x, y, 'r')

axes.set_xlabel('x')
axes.set_ylabel('y')
axes.set_title('title');
```



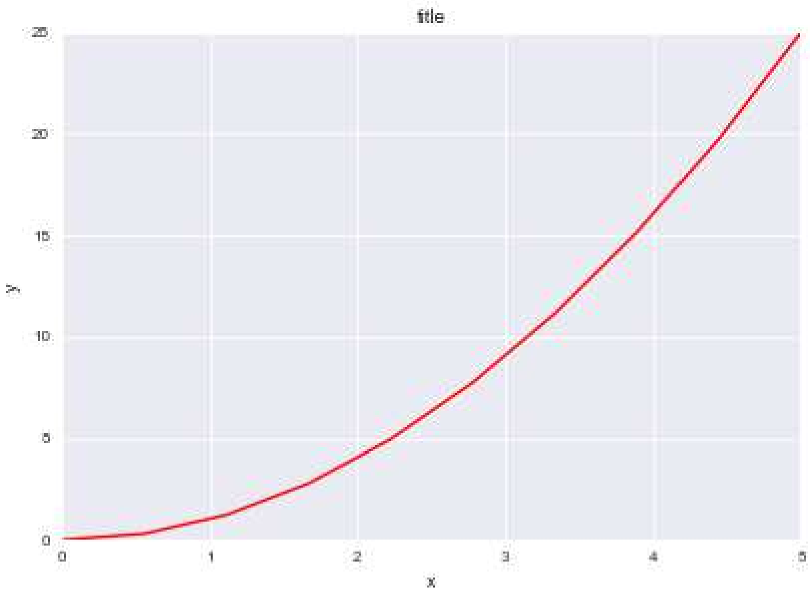
In [8]:

```
print(plt.style.available)
```

```
['seaborn-deep', 'seaborn-whitegrid', 'grayscale', 'seaborn-notebook', 'five
thirtyeight', 'seaborn-darkgrid', 'seaborn-white', 'seaborn-dark-palette',
 'seaborn-pastel', 'seaborn-poster', 'classic', 'bmh', 'seaborn-dark', 'seab
orn-paper', 'seaborn-talk', 'seaborn-ticks', 'seaborn-colorblind', 'seaborn-
bright', 'dark_background', 'seaborn-muted', 'ggplot']
```

In [13]:

```
plt.style.use('seaborn-paper')

x = np.linspace(0, 5, 10)
y = x ** 2

fig = plt.figure()

axes = fig.add_axes([0.1, 0.1, 0.8, 0.8]) # left, bottom, width, height (range 0 to 1)

axes.plot(x, y, 'r')

axes.set_xlabel('x')
axes.set_ylabel('y')
axes.set_title('title');
```



In [ ]:

Python for Astronomy

Pandas

何勃亮
中国科学院国家天文台 中国虚拟天文台 (China-VO)

In [1]:

```
%matplotlib inline

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

In [2]:

```
data = pd.read_table('data/dr1.sample', delimiter=',')
```

In [3]:

```
data.head()
```

Out[3]:

	ra	dec	mag2
0	332.202274	-2.056767	17.12
1	332.471576	-2.085015	18.10
2	332.368745	-1.955771	16.64
3	332.206665	-1.868653	17.19
4	332.348725	-2.136096	16.21

In [4]:

```
data.ra
```

Out[4]:

0	332.202274
1	332.471576
2	332.368745
3	332.206665
4	332.348725
5	332.444417
6	332.222379
7	332.351381
8	332.506374
9	332.244417
10	331.551234
11	331.768314
12	331.621652
13	331.772340
14	331.656890
15	331.753596
16	331.721996
17	331.645975
18	331.777682
19	331.755536
20	331.573480
21	331.829413
22	331.568160
23	331.854516
24	331.812125
25	331.730088
26	331.789860
27	331.764592
28	331.672794
29	331.698572
	...
969	333.422293
970	333.780378
971	333.612375
972	333.578496
973	333.416844
974	333.516416
975	333.360288
976	333.105776
977	333.337832
978	333.367444
979	333.250518
980	333.140347
981	333.132170
982	332.824680
983	333.047730
984	332.794817
985	332.925200
986	333.022310
987	332.929870
988	333.052310
989	332.842742
990	332.937900
991	333.088170
992	332.975800
993	332.964376
994	332.904375
995	332.857210
996	332.891050
997	333.020110
998	333.113113

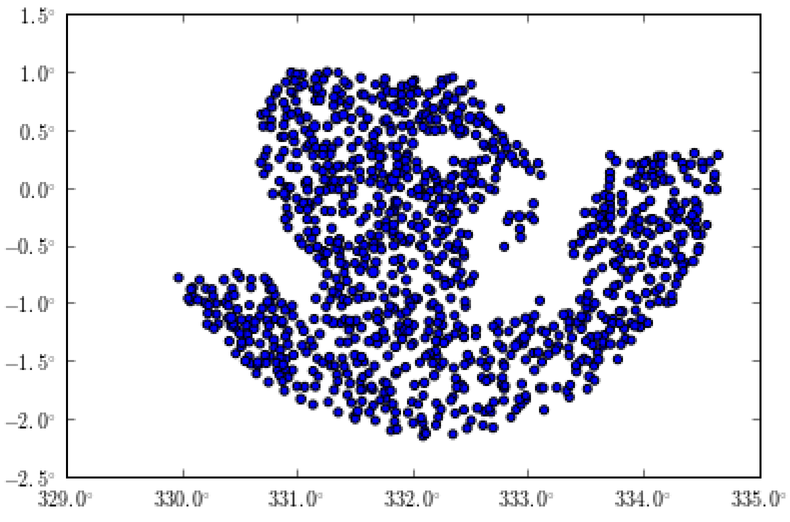
Name: ra, dtype: float64

In [5]:

```
from astropy import units as u

from astropy.coordinates import Angle

fig = plt.figure()

axes = fig.add_axes([0.1, 0.1, 0.8, 0.8]) # left, bottom, width, height (range 0 to 1)

axes.scatter(data['ra'], data['dec'], s = data['mag2'])

xticks = axes.get_xticks()
yticks = axes.get_yticks()

xt = ["$%.1f^{\circ}$"%n for n in xticks.tolist()]
yt = ["$%.1f^{\circ}$"%n for n in yticks.tolist()]

xt,yt
axes.set_xticklabels(xt)
axes.set_yticklabels(yt)
```

Out[5]:

```
[<matplotlib.text.Text at 0x10d447198>,
 <matplotlib.text.Text at 0x10d44bc50>,
 <matplotlib.text.Text at 0x10d472a20>,
 <matplotlib.text.Text at 0x10d455048>,
 <matplotlib.text.Text at 0x10d4474a8>,
 <matplotlib.text.Text at 0x10d43dcc0>,
 <matplotlib.text.Text at 0x10d4ac748>,
 <matplotlib.text.Text at 0x10d4b20f0>,
 <matplotlib.text.Text at 0x10d4b2b00>]
```



In [ ]:

[AstroML Sample (http://www.astroml.org/sklearn_tutorial/auto_examples/plot_sdss_images.html)](http://www.astroml.org/sklearn_tutorial/auto_examples/plot_sdss_images.html)

In [2]:

```
%matplotlib inline
```

In [3]:

```python
import os
from urllib.request import urlopen

import pylab as pl
from matplotlib import image

def _fetch(outfile, RA, DEC, scale=0.2, width=400, height=400):
    """Fetch the image at the given RA, DEC from the SDSS server"""
    url = ("http://casjobs.sdss.org/ImgCutoutDR7/"
           "getjpeg.aspx?ra=%.8f&dec=%.8f&scale=%.2f&width=%i&height=%i"
           % (RA, DEC, scale, width, height))
    print("downloading %s" % url)
    print(" -> %s" % outfile)
    fhandle = urlopen(url)
    open(outfile, 'wb').write(fhandle.read())


def fetch_image(object_type):
    """Return the data array for the image of object type"""
    if not os.path.exists('downloads'):
        os.makedirs('downloads')

    filename = os.path.join('downloads', '%s_image.jpg' % object_type)
    if not os.path.exists(filename):
        RA = image_locations[object_type]['RA']
        DEC = image_locations[object_type]['DEC']
        _fetch(filename, RA, DEC)

    return image.imread(filename)


image_locations = dict(star=dict(RA=180.63040108,
                                 DEC=64.96767375),
                       galaxy=dict(RA=197.51943983,
                                   DEC=0.94881436),
                       quasar=dict(RA=226.18451462,
                                   DEC=4.07456639))


# Plot the images
fig = pl.figure(figsize=(9, 3))

# Check that PIL is installed for jpg support
if 'jpg' not in fig.canvas.get_supported_filetypes():
    raise ValueError("PIL required to load SDSS jpeg images")

object_types = ['star', 'galaxy', 'quasar']

for i, object_type in enumerate(object_types):
    ax = pl.subplot(131 + i, xticks=[], yticks=[])
    I = fetch_image(object_type)
    ax.imshow(I)
    if object_type != 'galaxy':
        pl.arrow(0.65, 0.65, -0.1, -0.1, width=0.005, head_width=0.03,
                 length_includes_head=True,
                 color='w', transform=ax.transAxes)
    pl.text(0.99, 0.01, object_type, fontsize='large', color='w', ha='right',
            transform=ax.transAxes)

pl.subplots_adjust(bottom=0.04, top=0.94, left=0.02, right=0.98, wspace=0.04)
```

```
pl.show()
```

In [ ]:

```

```

In [2]:

```
import numpy as np
```

Python for Astronomy

AstroPy

何勃亮
中国科学院国家天文台 中国虚拟天文台 (China-VO)

AstroPy 项目

首先区分两个概念，**Astropy Project**和 **astropy** 包，前者是一个宏大的计划，后者则特指前者的核心软件包。

Astropy Project 由 **astropy**核心包和附属包组成。

Astropy Project项目自2011年开始发起，得到了多个国家的研究单位的支持，并且以开源社区的模式运行，目前的最新版本是 `v1.2` 。项目网站 http://www.astropy.org (http://www.astropy.org)

AstroPy 核心

AstroPy 核心包由一系列的基础组件组成，比如核心数据结构与算法、文件与数据I/O单位、天文计算与工具以及软件开发配置相关等。

核心数据结构与算法

- 天文常数 `astropy.constants`
- 单位与数量 `astropy.units`
- N维数据集 `astropy.nddata`
- 数据表格 `astropy.table`
- 时间与日期 `astropy.time`
- 天文坐标系统 `astropy.coordinates`
- 世界坐标系统 `astropy.wcs`
- 模型与拟合 `astropy.modeling`
- 数据分析程序 `astropy.analytic_functions`

文件与数据 I/O

- 通用文件读写接口
- FITS 文件操作 `astropy.io.fits`
- ASCII 表格操作 `astropy.io.ascii`
- VOTable 文件操作 `astropy.io.votable`
- I/O 杂项 `astropy.io.misc`

天文计算与工具

- 卷积与滤波 `astropy.convolution`
- 数据可视化 `astropy.visualization`
- 宇宙学计算 `astropy.cosmology`
- 天文统计学工具 `astropy.stats`
- 虚拟天文台访问工具 `astropy.vo`

软件开发配置相关

- 配置系统 `astropy.config`
- I/O注册 `astropy.io.registry`
- 日志系统
- Python 告警系统
- AstroPy 核心包工具集 `astropy.utils`
- AstroPy 测试助手 `astropy.tests.helper`

AstroPy 附属包

AstroPy成员包是指遵循 AstroPy开发、互操作和接口标准规范的一系列附属包,这些包有些是有一定历史的软件包,还有一些是在 AstroPy 基础上新开发出来的软件包,适合在不同场景下使用。在 http://www.astropy.org/affiliated/ (http://www.astropy.org/affiliated/) 可以查阅完整的附属包目录。

In [9]:

```
import astropy
```

天文常数

`AstroPy` 天文常数包： `astropy.constants` 。

常数值列表

名称	值	单位	描述
`G`	6.67384×10^{-11}	$m^3 kg^{-1} s^{-2}$	*重力常数*
`L_sun`	3.846×10^{26}	W	*太阳光度*
`M_earth`	5.9742×10^{24}	kg	*地球质量*
`M_jup`	1.8987×10^{27}	kg	*木星质量*
`M_sun`	1.9891×10^{30}	kg	*太阳质量*
`N_A`	$6.02214129 \times 10^{23}$	mol^{-1}	*阿伏伽德罗常数*
`R`	8.3144621	$JK^{-1} mol^{-1}$	*气体常数*
`R_earth`	6378136	m	*地球赤道半径*
`R_jup`	71492000	m	*木星赤道半径*
`R_sun`	695508000	m	*太阳赤道半径*
`Ryd`	10973731.6	m^{-1}	*里德伯常数 Rydberg constant*
`a0`	$5.29177211 \times 10^{-11}$	m	*玻尔半径 Bohr radius*
`alpha`	0.00729735257		*精细结构常数*
`atmosphere`	101325	Pa	*大气压*
`au`	$1.49597871 \times 10^{11}$	m	*天文单位*
`b_wien`	0.0028977721	mK	*维恩位移定律常数*
`c`	299792458	ms^{-1}	*真空光速*
`e`	$1.60217657 \times 10^{-19}$	C	*电子电荷*
`eps0`	$8.85418782 \times 10^{-12}$	Fm^{-1}	*介电常数*
`g0`	9.80665	ms^{-2}	*标准重力加速度*
`h`	$6.62606957 \times 10^{-34}$	Js	*普朗克常数*
`hbar`	$1.05457173 \times 10^{-34}$	Js	*约化普朗克常数*
`k_B`	$1.3806488 \times 10^{-23}$	JK^{-1}	*玻尔兹曼常数*
`kpc`	$3.08567758 \times 10^{19}$	m	*千秒差距*

m_e	$9.10938291 \times 10^{-31}$	kg	电子质量
m_n	$1.67492735 \times 10^{-27}$	kg	中子质量
m_p	$1.67262178 \times 10^{-27}$	kg	质子质量
mu0	$1.25663706 \times 10^{-6}$	NA^{-2}	磁性常数
muB	$9.27400968 \times 10^{-24}$	JT^{-1}	玻尔磁子
pc	$3.08567758 \times 10^{16}$	m	秒差距
sigma_T	$6.65245873 \times 10^{-29}$	m^2	汤姆逊散射截面
sigma_sb	5.670373×10^{-8}	$WK^{-4}m^{-2}$	斯蒂芬 - 玻尔兹曼常数
u	$1.66053892 \times 10^{-27}$	kg	原子质量单位

简单使用：

In [5]:

```
from astropy.constants import M_earth, M_sun, R_earth, R_sun
```

In [6]:

```
M_sun / M_earth
```

Out[6]:

332948.34

In [7]:

```
R_sun / R_earth
```

Out[7]:

109.04565

In [8]:

```
print(M_earth)
```

```
  Name   = Earth mass
  Value  = 5.9742e+24
  Uncertainty  = 5e+19
  Unit  = kg
  Reference = Allen's Astrophysical Quantities 4th Ed.
```

In [10]:

```
print(M_earth.cgs) # cgs格式

# 关于`Quantity`将在`astropy.units`中介绍，`constants`包搭配`unit`包，可以进行一些基本的计算转换。
```

5.9742e+27 g

Unit

`AstroPy Unit` ： `astropy.units` 。

`astropy.units` 定义物理量的单位，并可实现物理量的一些转换，比如 `KMS` 到 `cgs` 的转换等。

- `astropy.units.si` SI Unit
- `astropy.units.cgs` CGS Unit
- `astropy.units.astrophys` 天体物理相关单位
- `astropy.units.function.units`
- `astropy.units.imperial`
- `astropy.units.cds` CDS format unit
- `astropy.units.equivalencies` 一组标准的天文等价公式

In [10]:

```
from astropy import units as u
```

Quantity

In [18]:

```
k = 100 * u.AU

type(k)
```

Out[18]:

astropy.units.quantity.Quantity

In [19]:

```
k.value, k.unit
```

Out[19]:

(100.0, Unit("AU"))

In [24]:

```
# 单位转化

k.to(u.km)
```

Out[24]:

1.4959787×10^{10} km

In [20]:

```
15.1 * u.meter / (32.0 * u.second)
```

Out[20]:

$0.471875 \ \frac{\mathrm{m}}{\mathrm{s}}$

In [21]:

```
3.0 * u.kilometer / (130.51 * u.meter / u.second)
```

Out[21]:

$0.022986744 \ \frac{\mathrm{km\,s}}{\mathrm{m}}$

In [22]:

```
(3.0 * u.kilometer / (130.51 * u.meter / u.second)).decompose()
```

Out[22]:

$22.986744 \ \mathrm{s}$

In [25]:

```
# 自定义单位，并进行转换

from astropy.units import imperial

cms = u.cm / u.s

mph = imperial.mile / u.hour

q = 42.0 * cms

q.to(mph)
```

Out[25]:

$0.93951324 \ \frac{\mathrm{mi}}{\mathrm{h}}$

In [26]:

```
# 分析量纲

(u.s ** -1).compose()
```

Out[26]:

```
[Unit("Bq"), Unit("Hz"), Unit("2.7027e-11 Ci")]
```

In [29]:

```
# SI , CGS

k.si, k.cgs
```

Out[29]:

```
(<Quantity 14959787070000.0 m>, <Quantity 1495978707000000.0 cm>)
```

In [30]:

```
# 天文学等价公式

(1000 * u.nm).to(u.Hz)
```

```
---------------------------------------------------------------------------
UnitConversionError                       Traceback (most recent call last)
<ipython-input-30-65e17937747d> in <module>()
      1 # 天文学等价公式
      2 
----> 3 (1000 * u.nm).to(u.Hz)

/usr/local/lib/python3.5/site-packages/astropy/units/quantity.py in to(self,
 unit, equivalencies)
    651         unit = Unit(unit)
    652         new_val = self.unit.to(unit, self.view(np.ndarray),
--> 653                                equivalencies=equivalencies)
    654         return self._new_view(new_val, unit)
    655 

/usr/local/lib/python3.5/site-packages/astropy/units/core.py in to(self, oth
er, value, equivalencies)
    987             If units are inconsistent
    988         """
--> 989         return self._get_converter(other, equivalencies=equivalencie
s)(value)
    990 
    991     def in_units(self, other, value=1.0, equivalencies=[]):

/usr/local/lib/python3.5/site-packages/astropy/units/core.py in _get_convert
er(self, other, equivalencies)
    889                             pass
    890 
--> 891             raise exc
    892 
    893     @deprecated('1.0')

/usr/local/lib/python3.5/site-packages/astropy/units/core.py in _get_convert
er(self, other, equivalencies)
    875         try:
    876             return self._apply_equivalencies(
--> 877                 self, other, self._normalize_equivalencies(equivalen
cies))
    878         except UnitsError as exc:
    879             # Last hope: maybe other knows how to do it?

/usr/local/lib/python3.5/site-packages/astropy/units/core.py in _apply_equiv
alencies(self, unit, other, equivalencies)
    859         raise UnitConversionError(
    860             "{0} and {1} are not convertible".format(
--> 861                 unit_str, other_str))
    862 
    863     def _get_converter(self, other, equivalencies=[]):

UnitConversionError: 'nm' (length) and 'Hz' (frequency) are not convertible
```

In [31]:

```
(1000 * u.nm).to(u.Hz, equivalencies=u.spectral())
```

Out[31]:

2.9979246×10^{14} Hz

In [32]:

```
# 格式化

q = 15.1 * u.meter / (32.0 * u.second)

"{0.value:0.03f} {0.unit:FITS}".format(q)
```

Out[32]:

```
'0.472 m s-1'
```

N-dimensional datasets (`astropy.nddata`)¶

与 `numpy` 的 `ndarray` 类似，是对其的一个针对天文学应该的高级包装。

In [3]:

```
from astropy.nddata import NDData

array = np.zeros((12, 12, 12))

ndd1 = NDData(array)

ndd1
```

Out[3]:

```
NDData([[[ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         ...,
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.]],

        [[ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         ...,
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.]],

        [[ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         ...,
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.]],


        ...,
        [[ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         ...,
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.]],

        [[ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         ...,
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.]],

        [[ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         ...,
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.],
         [ 0.,  0.,  0., ...,  0.,  0.,  0.]]])
```

In [5]:

```
ndd1.meta
```

Out[5]:

```
OrderedDict()
```

In [6]:

```
ndd1.unit
```

In [7]:

```
ndd1.uncertainty
```

In [8]:

```
ndd1.wcs
```

In [11]:

```
ndd = NDData([1, 2, 3, 4], unit="meter")
ndd.unit
```

Out[11]:

m

In [12]:

```
ndd.meta['exposure_time'] = 340.

ndd.meta
```

Out[12]:

```
OrderedDict([('exposure_time', 340.0)])
```

Time

`from astropy.time import Time`

In [13]:

```
from astropy.time import Time
```

In [14]:

```
times = ['2016-01-01T00:00:00.123456789', '2015-01-01T00:00:00']

t = Time(times, format='isot', scale='utc')

t
```

Out[14]:

```
<Time object: scale='utc' format='isot' value=['2016-01-01T00:00:00.123' '20
15-01-01T00:00:00.000']>
```

Format

Format	Class	E
byear	`TimeBesselianEpoch` (../api/astropy.time.TimeBesselianEpoch.html#astropy.time.TimeBesselianEpoch)	1
byear_str	`TimeBesselianEpochString` (../api/astropy.time.TimeBesselianEpochString.html#astropy.time.TimeBesselianEpochString)	‘
cxcsec	`TimeCxcSec` (../api/astropy.time.TimeCxcSec.html#astropy.time.TimeCxcSec)	6
datetime	`TimeDatetime` (../api/astropy.time.TimeDatetime.html#astropy.time.TimeDatetime)	c 1
decimalyear	`TimeDecimalYear` (../api/astropy.time.TimeDecimalYear.html#astropy.time.TimeDecimalYear)	2
fits	`TimeFITS` (../api/astropy.time.TimeFITS.html#astropy.time.TimeFITS)	“ C
gps	`TimeGPS` (../api/astropy.time.TimeGPS.html#astropy.time.TimeGPS)	6
iso	`TimeISO` (../api/astropy.time.TimeISO.html#astropy.time.TimeISO)	“ C
isot	`TimeISOT` (../api/astropy.time.TimeISOT.html#astropy.time.TimeISOT)	“ C
jd	`TimeJD` (../api/astropy.time.TimeJD.html#astropy.time.TimeJD)	2
jyear	`TimeJulianEpoch` (../api/astropy.time.TimeJulianEpoch.html#astropy.time.TimeJulianEpoch)	2
jyear_str	`TimeJulianEpochString` (../api/astropy.time.TimeJulianEpochString.html#astropy.time.TimeJulianEpochString)	‘
mjd	`TimeMJD` (../api/astropy.time.TimeMJD.html#astropy.time.TimeMJD)	5
plot_date	`TimePlotDate` (../api/astropy.time.TimePlotDate.html#astropy.time.TimePlotDate)	7
unix	`TimeUnix` (../api/astropy.time.TimeUnix.html#astropy.time.TimeUnix)	9
yday	`TimeYearDayTime` (../api/astropy.time.TimeYearDayTime.html#astropy.time.TimeYearDayTime)	2

In [18]:

```
from datetime import datetime

time2 = datetime.now()

t = Time(time2, format='datetime', scale='utc')
```

In [19]:

```
t
```

Out[19]:

```
<Time object: scale='utc' format='datetime' value=2016-06-28 06:55:07.811364
>
```

In [27]:

```
t = Time.now()
t
```

Out[27]:

```
<Time object: scale='utc' format='datetime' value=2016-06-27 23:00:00.794208
>
```

In [20]:

```
t.iso
```

Out[20]:

```
'2016-06-28 06:55:07.811'
```

In [21]:

```
t.jd
```

Out[21]:

```
2457567.7882848536
```

In [22]:

```
t.mjd
```

Out[22]:

```
57567.28828485375
```

In [23]:

```
t.yday
```

Out[23]:

```
'2016:180:06:55:07.811'
```

In [24]:

```
t.byear
```

Out[24]:

```
2016.4911253597104
```

In [25]:

```
t.format
```

Out[25]:

```
'datetime'
```

In [26]:

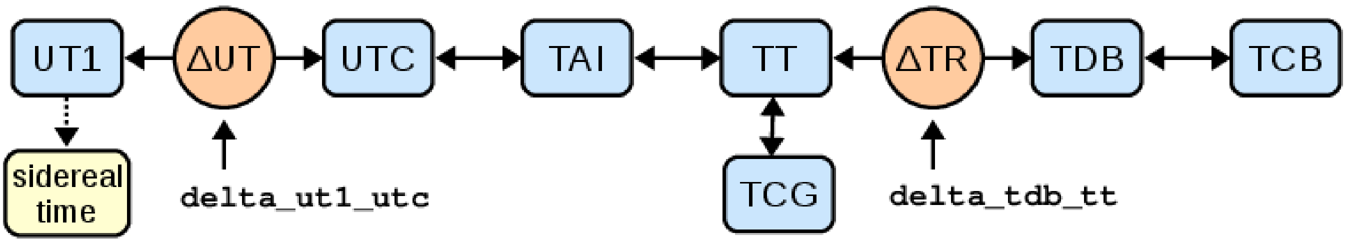
```
t.format = 'mjd'
t
```

Out[26]:

```
<Time object: scale='utc' format='mjd' value=57567.28828485375>
```

Time Scale (Time standard)

Scale	Description
tai	International Atomic Time (TAI)
tcb	Barycentric Coordinate Time (TCB)
tcg	Geocentric Coordinate Time (TCG)
tdb	Barycentric Dynamical Time (TDB)
tt	Terrestrial Time (TT)
ut1	Universal Time (UT1)
utc	Coordinated Universal Time (UTC)



文件读写

In [1]:

```
import pandas as pd
```

In [2]:

```
data = pd.read_csv('data/sample.txt', sep='|')
```

In [3]:

```
data
```

Out[3]:

	obsid	designation	obsdate	lmjd	planid	spid	fiberid	ra	dec	s
0	101001	J220848.54-020324.3	2011-10-24	55859	F5902	1	1	332.202274	-2.056767	2.
1	101002	J220953.17-020506.0	2011-10-24	55859	F5902	1	2	332.471576	-2.085015	2.
2	101008	J220928.49-015720.7	2011-10-24	55859	F5902	1	8	332.368745	-1.955771	1.
3	101009	J220849.59-015207.1	2011-10-24	55859	F5902	1	9	332.206665	-1.868653	1.
4	101016	J220923.69-020809.9	2011-10-24	55859	F5902	1	16	332.348725	-2.136096	2.
5	101017	J220946.66-015526.5	2011-10-24	55859	F5902	1	17	332.444417	-1.924046	2.
6	101020	J220853.37-015915.4	2011-10-24	55859	F5902	1	20	332.222379	-1.987626	2.
7	101021	J220924.33-014833.5	2011-10-24	55859	F5902	1	21	332.351381	-1.809333	6.
8	101023	J221001.52-020100.8	2011-10-24	55859	F5902	1	23	332.506374	-2.016900	2.

9 rows × 32 columns

In [ ]:

```
data['obsid']
```

In [ ]:

```
data['designation']
```

In [4]:

```
data['obsdate']
```

Out[4]:

```
0    2011-10-24
1    2011-10-24
2    2011-10-24
3    2011-10-24
4    2011-10-24
5    2011-10-24
6    2011-10-24
7    2011-10-24
8    2011-10-24
Name: obsdate, dtype: object
```

In [5]:

```
import numpy as np
from astropy.table import Table



a = [1, 4, 5]
b = [2.0, 5.0, 8.2]
c = ['x', 'y', 'z']
t = Table([a, b, c], names=('a', 'b', 'c'), meta={'name': 'first table'})

t
```

Out[5]:

<Table length=3>

a	b	c
int64	**float64**	**str1**
1	2.0	x
4	5.0	y
5	8.2	z

In [2]:

```
data_rows = [(1, 2.0, 'x'),
    (4, 5.0, 'y'),
    (5, 8.2, 'z')]

t = Table(rows=data_rows, names=('a', 'b', 'c'), meta={'name': 'first table'},
    dtype=('i4', 'f8', 'S1'))

t
```

Out[2]:

<Table length=3>

a	b	c
int32	**float64**	**bytes1**
1	2.0	x
4	5.0	y
5	8.2	z

In [3]:

```
t['b'].unit = 's'

t
```

Out[3]:

<Table length=3>

a	b	c
	s	
int32	**float64**	**bytes1**
1	2.0	x
4	5.0	y
5	8.2	z

In [5]:

```
t['b'].format = '7.3f'
```

In [6]:

```
t
```

Out[6]:

<Table length=3>

a	b	c
	s	
int32	float64	bytes1
1	2.000	x
4	5.000	y
5	8.200	z

In [7]:

```
t.colnames
```

Out[7]:

```
['a', 'b', 'c']
```

In [7]:

```
# 行索引

t[0:2]['b']
```

Out[7]:

<Column name='b' dtype='float64' length=2>

2.0
5.0

In [10]:

```
from astropy.time import Time
from astropy.coordinates import SkyCoord

tm = Time(['2000:002', '2002:345'])
sc = SkyCoord([10, 20], [-45, +40], unit='deg')
t = Table([tm, sc], names=['time', 'skycoord'])
t
```

Out[10]:

<Table length=2>

time	skycoord
	deg,deg
object	object
2000:002:00:00:00.000	10.0,-45.0
2002:345:00:00:00.000	20.0,40.0

In [13]:

```
from astropy.table import Table, Column

t = Table()
t['a'] = [1, 4]
t['b'] = Column([2.0, 5.0], unit='cm', description='Velocity')
t['c'] = ['x', 'y']

t = Table(names=('a', 'b', 'c'), dtype=('f4', 'i4', 'S2'))
t.add_row((1, 2.0, 'x'))
t.add_row((4, 5.0, 'y'))

t
```

Out[13]:

<Table length=2>

a	b	c
float32	**int32**	**bytes2**
1.0	2	x
4.0	5	y

In [17]:

```
# Column

col = Column([1, 2], name='a')  # shape=(2,)
col = Column([[1, 2], [3, 4]], name='a')  # shape=(2, 2)
col = Column([1, 2], name='a', dtype=float)
col = Column(np.array([1, 2]), name='a')
col = Column(['hello', 'world'], name='a')
```

In [19]:

```
t.columns   # Dict of table columns (access by column name, index, or slice)
t.colnames  # List of column names
t.meta      # Dict of meta-data
len(t)      # Number of table rows
```

Out[19]:

```
3
```

In [ ]:

```
# access

t['a']       # Column 'a'
t['a'][1]    # Row 1 of column 'a'
t[1]         # Row obj for with row 1 values
t[1]['a']    # Column 'a' of row 1
t[2:5]       # Table object with rows 2:5
t[[1, 3, 4]]  # Table object with rows 1, 3, 4 (copy)
t[np.array([1, 3, 4])]  # Table object with rows 1, 3, 4 (copy)
t[[]]        # Same table definition but with no rows of data
t['a', 'c']  # Table with cols 'a', 'c' (copy)
dat = np.array(t)  # Copy table data to numpy structured array object
t['a'].quantity  # an astropy.units.Quantity for Column 'a'
t['a'].to('km')  # an astropy.units.Quantity for Column 'a' in units of kilometers
t.columns[1]  # Column 1 (which is the 'b' column)
t.columns[0:2]  # New table with columns 0 and 1
```

In [ ]:

```
##

print(t)     # Print formatted version of table to the screen
t.pprint()   # Same as above
t.pprint(show_unit=True)  # Show column unit
t.pprint(show_name=False)  # Do not show column names
t.pprint(max_lines=-1, max_width=-1)  # Print full table no matter how long / wide it is

t.more()  # Interactively scroll through table like Unix "more"

print(t['a'])    # Formatted column values
t['a'].pprint()  # Same as above, with same options as Table.pprint()
t['a'].more()    # Interactively scroll through column

lines = t.pformat()  # Formatted table as a list of lines (same options as pprint)
lines = t['a'].pformat()  # Formatted column values as a list
```

In [23]:

```
t['b'].format = "%6.3f"
t['b'].unit = 'm sec^-1'
t
```

Out[23]:

<Table length=3>

a	b	c
	m sec^-1	
int64	float64	str1
1	2.000	x
4	5.000	y
5	8.200	z

In [26]:

```
t.info('stats')
```

```
<Table length=3>
name      mean           std      min max
---- ------------- ------------- --- ---
   a 3.33333333333  1.6996731712   1   5
   b 5.06666666667 2.53157833947 2.0 8.2
   c            --            --  --  --
```

```
/Users/hebl/anaconda/lib/python3.5/site-packages/astropy/table/column.py:26
8: FutureWarning: elementwise comparison failed; returning scalar instead, b
ut in the future will perform elementwise comparison
  return self.data.__eq__(other)
```

In [27]:

```
t['b'].info
```

Out[27]:

```
name = b
dtype = float64
unit = m sec^-1
format = %6.3f
class = Column
n_bad = 0
length = 3
```

In [8]:

```
!head data/catalog
```

```
   1          BD+44 4550      3 36042          46           000001.1+4440220
00509.9+451345114.44-16.88 6.70  +0.07 +0.08         A1Vn               -0.0
12-0.018      -018      195  4.2  21.6AC   3
   2          BD-01 4525      6128569                       235956.2-0103300
00503.8-003011 98.33-61.14 6.29  +1.10 +1.02        gG9                 +0.0
45-0.060      +014V
   3 33    PscBD-06 6357     281285721002I         Var?     000013.0-0616010
00520.1-054227 93.75-65.93 4.61  +1.04 +0.89 +0.54   K0IIIbCN-0.5       -0.0
09+0.089 +.014-006SB1O < 17  2.5   0.0     3*
   4 86    PegBD+12 5063     87 917012004                   000033.8+1250230
00542.0+132346106.19-47.98 5.51  +0.90               G5III              +0.0
45-0.012      -002V?
   5          BD+57 2865    123 21085          61  V640 Cas 000101.8+5752450
00616.0+582612117.03-03.92 5.96  +0.67 +0.20         G5V                +0.2
63+0.030 +.047-012V          0.8   1.4      *
   6          CD-4914337    142214963      W                000108.4-4937510
00619.0-490430321.61-66.38 5.70  +0.52 +0.05         G1IV               +0.5
65-0.038 +.050+003SB         5.7   5.4      *
   7 10    CasBD+63 2107    144 109782005                   000114.4+6338220
00626.5+641146118.06  1.75 5.59  -0.03 -0.19         B9III             e+0.0
08 0.000      -000V     153                 *
   8          BD+28 4704    166 73743          69     33    000125.2+2828110
00636.8+290117111.26-32.83 6.13  +0.75 +0.33         K0V                +0.3
80-0.182 +.067-008V          2.6 158.6AB   4*
   9          CD-23    4    2031660531003                   000143.0-2339470
00650.1-230627 52.21-79.14 6.18  +0.38 +0.05         A7V                +0.1
00-0.045      +003V
  10          BD-18 6428    256147090                       000211.8-1756390
00718.2-172311 74.36-75.90 6.19  +0.14 +0.10         A6Vn               -0.0
18+0.036      -009V?    195                 *
```

In [31]:

```
##IO

bsc = Table.read("data/catalog",
                readme="http://vizier.china-vo.org/ftp/cats/V/50/ReadMe",
               format="ascii.cds")
bsc
```

Downloading http://vizier.china-vo.org/ftp/cats/V/50/ReadMe [Done]

Out[31]:

<Table masked=True length=9110>

HR	Name	DM	HD	SAO	FK5	IRflag	r_IRflag	Multiple	ADS	ADScom
int64	**str10**	**str11**	**int64**	**int64**	**int64**	**str1**	**str1**	**str1**	**str5**	**str2**
1	--	BD+44 4550	3	36042	--	--	--	--	46	--
2	--	BD-01 4525	6	128569	--	--	--	--	--	--
3	33 Psc	BD-06 6357	28	128572	1002	I	--	--	--	--
4	86 Peg	BD+12 5063	87	91701	2004	--	--	--	--	--
5	--	BD+57 2865	123	21085	--	--	--	--	61	--
6	--	CD-4914337	142	214963	--	--	--	W	--	--
7	10 Cas	BD+63 2107	144	10978	2005	--	--	--	--	--
8	--	BD+28 4704	166	73743	--	--	--	--	69	--
9	--	CD-23 4	203	166053	1003	--	--	--	--	--
...	...	...	...	...	...	...	...	...	...	...
9101	--	BD-17 6868	225197	147064	--	--	--	--	--	--
9102	--	CD-2918950	225200	166031	--	--	--	--	--	--
9103	3 Cet	BD-11 6194	225212	147066	2001	I	--	--	--	--
9104	--	BD+66 1679	225216	10956	--	--	--	--	--	--
9105	--	BD+41 4933	225218	36037	--	I	--	--	30	--
9106	--	CP-73 2346	225233	255629	--	--	--	--	--	--
9107	--	BD+33 4828	225239	53622	2002	--	--	--	--	--
9108	--	CP-72 2800	225253	255631	1001	--	--	--	--	--
		BD+25								

9109	--	5068	225276	73731	--	I	--	--	42	--
9110	--	BD+60 2667	225289	10962	--	--	--	--	--	--

Format	Read	Write	Auto-identify	Deprecated
aastex	Yes	Yes	No	Yes
ascii	Yes	Yes	No	
ascii.aastex	Yes	Yes	No	
ascii.basic	Yes	Yes	No	
ascii.cds	Yes	No	No	
ascii.commented_header	Yes	Yes	No	
ascii.daophot	Yes	No	No	
ascii.ecsv	Yes	Yes	No	
ascii.fixed_width	Yes	Yes	No	
ascii.fixed_width_no_header	Yes	Yes	No	
ascii.fixed_width_two_line	Yes	Yes	No	
ascii.html	Yes	Yes	Yes	
ascii.ipac	Yes	Yes	No	
ascii.latex	Yes	Yes	Yes	
ascii.no_header	Yes	Yes	No	
ascii.rdb	Yes	Yes	Yes	
ascii.sextractor	Yes	No	No	
ascii.tab	Yes	Yes	No	
ascii.csv	Yes	Yes	Yes	
cds	Yes	No	No	Yes
daophot	Yes	No	No	Yes
fits	Yes	Yes	Yes	
hdf5	Yes	Yes	Yes	
html	Yes	Yes	No	Yes
ipac	Yes	Yes	No	Yes
latex	Yes	Yes	No	Yes
rdb	Yes	Yes	No	Yes
votable	Yes	Yes	Yes	

In [ ]:

```
t = Table.read('aj285677t3_votable.xml')
```

In [ ]:

```

```

In [1]:

```
%matplotlib inline
import numpy as np
```

In [2]:

```
from astropy.coordinates import SkyCoord  # High-level coordinates
from astropy.coordinates import ICRS, Galactic, FK4, FK5  # Low-level frames
from astropy.coordinates import Angle, Latitude, Longitude  # Angles
import astropy.units as u
```

语法

```
SkyCoord(COORD, [FRAME | frame=FRAME], [unit=UNIT], keyword_args ...)
SkyCoord(LON, LAT, [DISTANCE], [FRAME | frame=FRAME], [unit=UNIT], keyword_args ...)
SkyCoord([FRAME | frame=FRAME], <lon_name>=LON, <lat_name>=LAT, [unit=UNIT],
        keyword_args ...)
```

In [4]:

```
c1 = SkyCoord(10, 20, unit='deg')
c1
```

Out[4]:

```
<SkyCoord (ICRS): (ra, dec) in deg
    (10.0, 20.0)>
```

In [5]:

```
coords = ["1:12:43.2 +1:12:43", "1 12 43.2 +1 12 43"]

c2 = SkyCoord(coords, frame=FK4, unit=(u.hourangle, u.deg), obstime="J1992.21")
c2
```

Out[5]:

```
<SkyCoord (FK4: equinox=B1950.000, obstime=J1992.210): (ra, dec) in deg
    [(18.18, 1.21194444), (18.18, 1.21194444)]>
```

In [6]:

```
c3 = SkyCoord("1h12m43.2s", "+1d12m43s", frame=Galactic)
c3
```

Out[6]:

```
<SkyCoord (Galactic): (l, b) in deg
    (18.18, 1.21194444)>
```

In [8]:

```
c3 = SkyCoord("1h12m43.2s +1d12m43s", frame=Galactic)
c3
```

Out[8]:

```
<SkyCoord (Galactic): (l, b) in deg
    (18.18, 1.21194444)>
```

In [9]:

```
c4 = SkyCoord("15h17+89d15")
c4
```

Out[9]:

```
<SkyCoord (ICRS): (ra, dec) in deg
    (229.25, 89.25)>
```

In [16]:

```
c5 = SkyCoord("J080000.00-050036.00", unit=(u.hour, u.deg))
```

In [17]:

```
c1,c2,c3,c4,c5
```

Out[17]:

```
(<SkyCoord (ICRS): (ra, dec) in deg
     (10.0, 20.0)>,
 <SkyCoord (FK4: equinox=B1950.000, obstime=J1992.210): (ra, dec) in deg
     [(18.18, 1.21194444), (18.18, 1.21194444)]>,
 <SkyCoord (Galactic): (l, b) in deg
     (18.18, 1.21194444)>,
 <SkyCoord (ICRS): (ra, dec) in deg
     (229.25, 89.25)>,
 <SkyCoord (ICRS): (ra, dec) in deg
     (120.0, -5.01)>)
```

In [18]:

```
c1.data
```

Out[18]:

```
<UnitSphericalRepresentation (lon, lat) in deg
    (10.0, 20.0)>
```

In [20]:

```
c1.ra, c1.dec
```

Out[20]:

```
(<Longitude 10.0 deg>, <Latitude 20.0 deg>)
```

In [21]:

```
c1.distance(c2)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-21-1903a29c48c4> in <module>()
----> 1 c1.distance(c2)

TypeError: 'Quantity' object is not callable
```

In [24]:

```
c2.obstime
```

Out[24]:

```
<Time object: scale='utc' format='jyear_str' value=J1992.210>
```

In [25]:

```
c1.distance
```

Out[25]:

1

In [26]:

```
c4.spherical
```

Out[26]:

```
<SphericalRepresentation (lon, lat, distance) in (deg, deg, )
    (229.25, 89.25, 1.0)>
```

In [27]:

```
c2.to_string
```

Out[27]:

```
<bound method SkyCoord.to_string of <SkyCoord (FK4: equinox=B1950.000, obsti
me=J1992.210): (ra, dec) in deg
    [(18.18, 1.21194444), (18.18, 1.21194444)]>>
```

In [28]:

```
c2.frame
```

Out[28]:

```
<FK4 Coordinate (equinox=B1950.000, obstime=J1992.210): (ra, dec) in deg
    [(18.18, 1.21194444), (18.18, 1.21194444)]>
```

In [29]:

```
c2.frame.data
```

Out[29]:

```
<UnitSphericalRepresentation (lon, lat) in (hourangle, deg)
    [(1.212, 1.21194444), (1.212, 1.21194444)]>
```

In [30]:

```
# Transform

from astropy.coordinates import FK5
```

In [31]:

```
c1.galactic
```

Out[31]:

```
<SkyCoord (Galactic): (l, b) in deg
    (119.26936774, -42.79039286)>
```

In [32]:

```
c1
```

Out[32]:

```
<SkyCoord (ICRS): (ra, dec) in deg
    (10.0, 20.0)>
```

In [33]:

```
c1.transform_to('fk5')
```

Out[33]:

```
<SkyCoord (FK5: equinox=J2000.000): (ra, dec) in deg
    (10.0000085, 20.00000153)>
```

In [34]:

```
c1.transform_to(FK5(equinox='J1975'))
```

Out[34]:

```
<SkyCoord (FK5: equinox=J1975.000): (ra, dec) in deg
    (9.67112136, 19.86286033)>
```

In [35]:

```
# 坐标系统转换

c = SkyCoord(x=1, y=2, z=3, unit='kpc', representation='cartesian')
```

In [36]:

```
c
```

Out[36]:

```
<SkyCoord (ICRS): (x, y, z) in kpc
    (1.0, 2.0, 3.0)>
```

In [37]:

```
c.representation = 'cylindrical'
```

In [38]:

```
c
```

Out[38]:

```
<SkyCoord (ICRS): (rho, phi, z) in (kpc, deg, kpc)
    (2.23606798, 63.43494882, 3.0)>
```

In [40]:

```
c.info
```

Out[40]:

```
dtype = object
unit = kpc,deg,kpc
class = SkyCoord
n_bad = 0
```

In [41]:

```
c.representation = 'spherical'
```

In [42]:

```
c
```

Out[42]:

```
<SkyCoord (ICRS): (ra, dec, distance) in (deg, deg, kpc)
    (63.43494882, 53.3007748, 3.74165739)>
```

In [43]:

```
c.representation = 'unitspherical'
```

In [44]:

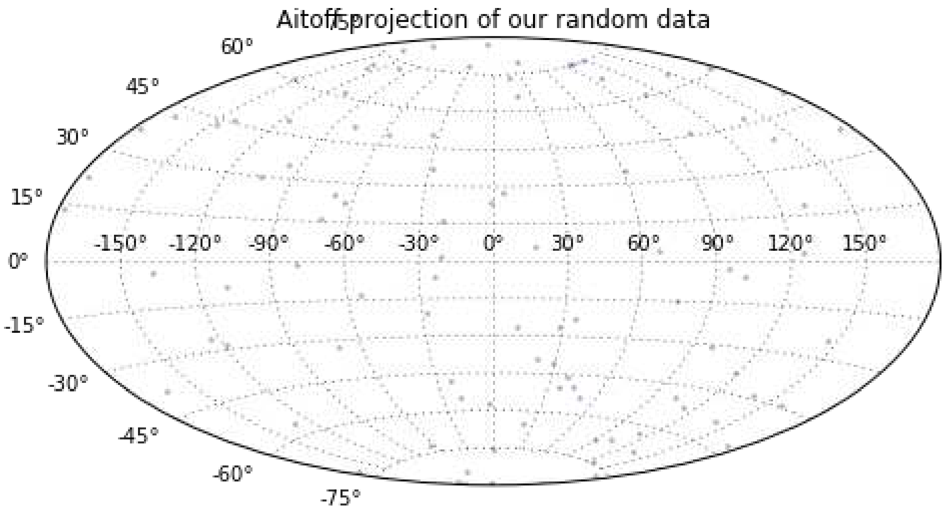
```
c
```

Out[44]:

```
<SkyCoord (ICRS): (ra, dec) in deg
    (63.43494882, 53.3007748)>
```

In [48]:

```
import matplotlib.pyplot as plt

ra = np.random.rand(100)*360.0 * u.degree
dec = (np.random.rand(100)*180.0-90.0) * u.degree

c = SkyCoord(ra=ra, dec=dec, frame='icrs')

ra_rad = c.ra.wrap_at(180 * u.deg).radian
dec_rad = c.dec.radian

fig = plt.figure(figsize=(8,4.2))

ax = plt.subplot(111, projection="aitoff")
ax.set_title("Aitoff projection of our random data")
ax.grid(True)
ax.plot(ra_rad, dec_rad, 'o', markersize=2, alpha=0.3)
fig.subplots_adjust(top=0.95,bottom=0.0)
```



In [ ]:

```
```

In [1]:

```
import numpy as np
from astropy.vo.client import conesearch
conesearch.list_catalogs()
```

Out[1]:

```
['2MASS All-Sky Point Source Catalog 1',
 'Guide Star Catalog v2 1',
 'SDSS DR7 - Sloan Digital Sky Survey Data Release 7 1',
 'SDSS DR7 - Sloan Digital Sky Survey Data Release 7 2',
 'SDSS DR7 - Sloan Digital Sky Survey Data Release 7 3',
 'SDSS DR7 - Sloan Digital Sky Survey Data Release 7 4',
 'SDSS DR8 - Sloan Digital Sky Survey Data Release 8 1',
 'SDSS DR8 - Sloan Digital Sky Survey Data Release 8 2',
 'The HST Guide Star Catalog, Version 1.1 (Lasker+ 1992) 1',
 'The HST Guide Star Catalog, Version 1.2 (Lasker+ 1996) 1',
 'The HST Guide Star Catalog, Version GSC-ACT (Lasker+ 1996-99) 1',
 'The PMM USNO-A1.0 Catalogue (Monet 1997) 1',
 'The USNO-A2.0 Catalogue (Monet+ 1998) 1',
 'Two Micron All Sky Survey (2MASS) 1',
 'Two Micron All Sky Survey (2MASS) 2',
 'USNO-A2 Catalogue 1']
```

In [2]:

```
from astropy.coordinates import SkyCoord
from astropy import units as u

c = SkyCoord.from_name('M31')
c
```

Out[2]:

```
<SkyCoord (ICRS): (ra, dec) in deg
    (10.6847083, 41.26875)>
```

In [3]:

```
result = conesearch.conesearch(c, 0.1 * u.degree, catalog_db='2MASS All-Sky Point Source
 Catalog 1')
```

```
Trying http://irsa.ipac.caltech.edu/cgi-bin/Oasis/CatSearch/nph-catsearch?CA
T=fp_psc&
Downloading http://irsa.ipac.caltech.edu/cgi-bin/Oasis/CatSearch/nph-catsear
ch?CAT=fp_psc&SR=0.1&VERB=1&RA=10.6847083&DEC=41.26875 [Done]
```

```
WARNING: W22: http://irsa.ipac.caltech.edu/cgi-bin/Oasis/CatSearch/nph-catse
arch?CAT=fp_psc&SR=0.1&VERB=1&RA=10.6847083&DEC=41.26875:5:0: W22: The DEFIN
ITIONS element is deprecated in VOTable 1.1.  Ignoring [astropy.io.votable.t
ree]
WARNING: W27: http://irsa.ipac.caltech.edu/cgi-bin/Oasis/CatSearch/nph-catse
arch?CAT=fp_psc&SR=0.1&VERB=1&RA=10.6847083&DEC=41.26875:6:0: W27: COOSYS de
precated in VOTable 1.2 [astropy.io.votable.tree]
WARNING: W06: http://irsa.ipac.caltech.edu/cgi-bin/Oasis/CatSearch/nph-catse
arch?CAT=fp_psc&SR=0.1&VERB=1&RA=10.6847083&DEC=41.26875:12:0: W06: Invalid
 UCD 'POS_EQ_RA_MAIN': Unknown word 'POS_EQ_RA_MAIN' [astropy.io.votable.tre
e]
WARNING: W06: http://irsa.ipac.caltech.edu/cgi-bin/Oasis/CatSearch/nph-catse
arch?CAT=fp_psc&SR=0.1&VERB=1&RA=10.6847083&DEC=41.26875:13:0: W06: Invalid
 UCD 'POS_EQ_DEC_MAIN': Unknown word 'POS_EQ_DEC_MAIN' [astropy.io.votable.t
ree]
WARNING: W06: http://irsa.ipac.caltech.edu/cgi-bin/Oasis/CatSearch/nph-catse
arch?CAT=fp_psc&SR=0.1&VERB=1&RA=10.6847083&DEC=41.26875:40:0: W06: Invalid
 UCD 'ID_MAIN': Unknown word 'ID_MAIN' [astropy.io.votable.tree]
```

In [4]:

```
result
```

Out[4]:

```
<Table masked=True length=2008>
   ra     dec         err_maj        ... scan_key coadd_key coadd
  deg     deg                        ...
float64 float64       float64        ...  int32     int32   int32
------- ------- -------------------- ... -------- --------- -----
 10.733  41.215                 0.22 ...    69157   1590591    33
 10.738  41.216                 0.25 ...    69157   1590591    33
 10.726  41.211  0.23000000000000001 ...    69157   1590591    33
 10.740  41.193  0.20000000000000001 ...    69157   1590592    44
 10.744  41.198                 0.13 ...    69157   1590592    44
 10.684  41.194  0.27000000000000002 ...    69157   1590592    44
 10.683  41.193  0.28999999999999998 ...    69157   1590592    44
 10.689  41.190  0.34999999999999998 ...    69157   1590592    44
 10.692  41.195  0.28000000000000003 ...    69157   1590592    44
    ...     ...                  ... ...      ...       ...   ...
 10.779  41.219  0.29999999999999999 ...    69158   1590631   232
 10.768  41.226  0.29999999999999999 ...    69157   1590591    33
 10.810  41.238  0.35999999999999999 ...    69158   1590631   232
 10.787  41.233                 0.31 ...    69158   1590631   232
 10.794  41.238                 0.13 ...    69158   1590631   232
 10.798  41.238  0.26000000000000001 ...    69158   1590631   232
 10.801  41.238  0.27000000000000002 ...    69158   1590631   232
 10.806  41.237 0.080000000000000002 ...    69158   1590631   232
 10.780  41.203  0.23999999999999999 ...    69158   1590631   232
 10.793  41.222                 0.25 ...    69158   1590631   232
```

In [5]:

```
result.array
```

Out[5]:

```
masked_array(data = [ (10.733387, 41.214523, 0.22, 0.2, 178, b'00425601+4112
522', 15.968, --, --, --, 14.864, --, --, --, 15.066, 0.1738, 0.174, 8.51,
 b'UUC', b'002', b'001', b'000', b'000004', 12.1, 69, 1287660706, 2, 0, b'12
87660709', b'n', b'1997-10-24', 8, 121.211558, -21.628691, -42.4, 2450745.85
89, --, --, 1.01, --, --, --, --, 15.685, 0.431, 2873.2, 207.5, b'ne', 0, 1,
 b'0', --, --, --, --, 0, --, 69157, 1590591, 33)
 (10.73756, 41.215744, 0.25, 0.21, 81, b'00425701+4112566', 16.313, 0.1649,
 0.1653, 9.71, 14.736, --, --, --, 14.256, --, --, --, b'CUU', b'200', b'10
0', b'000', b'050000', 12.1, 249, 1287660709, 2, 0, b'1287660706', b'n', b'1
997-10-24', 8, 121.214979, -21.627583, -53.7, 2450745.8589, 1.47, --, --, 1
5.749, 0.143, --, --, --, --, 2868.7, 196.2, b'ne', 0, 1, b'0', --, --, --,
 --, 0, --, 69157, 1590591, 33)
 (10.726409, 41.210964, 0.23, 0.21, 89, b'00425433+4112394', 16.3, 0.1528,
 0.1533, 9.83, 15.077, --, --, --, 14.548, --, --, --, b'BUU', b'200', b'10
0', b'000', b'050000', 22.8, 56, 1287660709, 2, 0, b'1287660724', b'n', b'19
97-10-24', 8, 121.205778, -21.632061, -23.5, 2450745.8589, 1.48, --, --, 16.
611, 0.367, --, --, --, --, 2886.2, 226.4, b'ne', 0, 1, b'0', --, --, --, -
-, 0, --, 69157, 1590591, 33)
 ...,
 (10.805706, 41.237198, 0.08, 0.06, 90, b'00431336+4114139', 12.351, 0.0194,
 0.0229, 360.6, 12.026, 0.0202, 0.022, 223.11, 11.945, 0.02, 0.0218, 145.69,
 b'AAA', b'222', b'111', b'000', b'666666', 11.3, 73, 1287682716, 2, 0, b'12
87682701', b'n', b'1997-10-24', 9, 121.270882, -21.607929, 162.63, 2450745.8
671, 1.35, 0.99, 1.15, 12.355, 0.011, 12.049, 0.007, 11.944, 0.027, 2779.7,
 87.5, b'nw', 1, 1, b'0', --, --, --, --, 0, --, 69158, 1590631, 232)
 (10.780244, 41.203281, 0.24, 0.23, 83, b'00430725+4112118', 16.113, --, --,
 --, 15.767, 0.1867, 0.1869, 7.11, 14.682, --, --, --, b'UCU', b'020', b'01
0', b'000', b'001500', 21.2, 272, 1287660744, 2, 0, b'1287682447', b'n', b'1
997-10-24', 9, 121.249032, -21.641165, 231.7, 2450745.8671, --, 0.83, --, -
-, --, 15.572, 0.428, --, --, 2902.3, 18.5, b'nw', 0, 0, b'0', --, --, --, -
-, 0, --, 69158, 1590631, 232)
 (10.792816, 41.222042, 0.25, 0.23, 84, b'00431027+4113193', 16.481, 0.1669,
 0.1673, 8.04, 15.627, 0.1766, 0.1768, 8.09, 14.912, --, --, --, b'CCU', b'2
20', b'110', b'000', b'050500', 37.2, 301, 1287682628, 2, 0, b'1287682587',
 b'n', b'1997-10-24', 9, 121.259899, -21.622743, 197.58, 2450745.8671, 1.42,
 1.18, --, 16.057, 0.194, 15.071, 0.124, --, --, 2834.5, 52.6, b'nw', 0, 1,
 b'0', --, --, --, --, 0, --, 69158, 1590631, 232)],
             mask = [ (False, False, False, False, False, False, False, Tru
e, True, True, False, True, True, True, False, False, False, False, False, F
alse, False, False, False, False, False, False, False, False, False, False,
 False, False, False, False, False, False, True, True, False, True, True, Tr
ue, True, False, False, False, False, False, False, False, False, True, Tru
e, True, True, False, True, False, False, False)
 (False, False, False, False, False, False, False, False, False, False, Fals
e, True, True, True, False, True, True, True, False, False, False, False, Fa
lse, False, False, False, False, False, False, False, False, False, False, F
alse, False, False, False, True, True, False, False, True, True, True, True,
 False, False, False, False, False, False, True, True, True, True, False, Tr
ue, False, False, False)
 (False, False, False, False, False, False, False, False, False, False, Fals
e, True, True, True, False, True, True, True, False, False, False, False, Fa
lse, False, False, False, False, False, False, False, False, False, False, F
alse, False, False, False, True, True, False, False, True, True, True, True,
 False, False, False, False, False, False, True, True, True, True, False, Tr
ue, False, False, False)
 ...,
 (False, False, False, False, False, False, False, False, False, False, Fals
e, False, False, False, False, False, False, False, False, False, False, Fal
se, False, False, False, False, False, False, False, False, False, False, Fa
lse, False, False, False, False, False, False, False, False, False, False, F
alse, False, False, False, False, False, False, False, False, True, True, True, Tru
```

```
e, False, True, False, False, False)
 (False, False, False, False, False, False, False, True, True, True, False,
 False, False, False, False, True, True, True, False, False, False, False, F
alse, False, False, False, False, False, False, False, False, False, False,
 False, False, False, True, False, True, True, True, False, False, True, Tru
e, False, False, False, False, False, False, True, True, True, True, False,
 True, False, False, False)
 (False, False, False, False, False, False, False, False, False, False, Fals
e, False, False, False, False, True, True, True, False, False, False, False,
 False, False, False, False, False, False, False, False, False, False, Fals
e, False, False, False, False, False, True, False, False, False, False, Tru
e, True, False, False, False, False, False, False, True, True, True, True, F
alse, True, False, False, False)],
       fill_value = (1e+20, 1e+20, 1e+20, 1e+20, 999999, '?', 1e+20, 1e+20,
 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, '?',
 '?', '?', '?', '?', 1e+20, 999999, 999999, 999999, 999999, '?', '?', '?', 9
99999, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, 1e+20,
 1e+20, 1e+20, 1e+20, 1e+20, 1e+20, '?', 999999, 999999, '?', 1e+20, 999999,
 1e+20, 1e+20, 999999, 999999, 999999, 999999, 999999),
            dtype = (numpy.record, [('ra', '<f8'), ('dec', '<f8'), ('err_ma
j', '<f8'), ('err_min', '<f8'), ('err_ang', '<i4'), ('designation', 'O'),
 ('j_m', '<f8'), ('j_cmsig', '<f8'), ('j_msigcom', '<f8'), ('j_snr', '<f8'),
 ('h_m', '<f8'), ('h_cmsig', '<f8'), ('h_msigcom', '<f8'), ('h_snr', '<f8'),
 ('k_m', '<f8'), ('k_cmsig', '<f8'), ('k_msigcom', '<f8'), ('k_snr', '<f8'),
 ('ph_qual', 'O'), ('rd_flg', 'O'), ('bl_flg', 'O'), ('cc_flg', 'O'), ('nde
t', 'O'), ('prox', '<f8'), ('pxpa', '<i4'), ('pxcntr', '<i4'), ('gal_conta
m', '<i4'), ('mp_flg', '<i4'), ('cntr', 'O'), ('hemis', 'O'), ('xdate',
 'O'), ('scan', '<i4'), ('glon', '<f8'), ('glat', '<f8'), ('x_scan', '<f8'),
 ('jdate', '<f8'), ('j_psfchi', '<f8'), ('h_psfchi', '<f8'), ('k_psfchi', '<
f8'), ('j_m_stdap', '<f8'), ('j_msig_stdap', '<f8'), ('h_m_stdap', '<f8'),
 ('h_msig_stdap', '<f8'), ('k_m_stdap', '<f8'), ('k_msig_stdap', '<f8'), ('d
ist_edge_ns', '<f8'), ('dist_edge_ew', '<f8'), ('dist_edge_flg', 'O'), ('dup
_src', '<i4'), ('use_src', '<i4'), ('a', 'O'), ('dist_opt', '<f8'), ('phi_op
t', '<i4'), ('b_m_opt', '<f8'), ('vr_m_opt', '<f8'), ('nopt_mchs', '<i4'),
 ('ext_key', '<i4'), ('scan_key', '<i4'), ('coadd_key', '<i4'), ('coadd', '<
i4')]))
```

In [6]:

```
result.array['ra']
```

Out[6]:

```
masked_array(data = [10.733387 10.73756 10.726409 ..., 10.805706 10.780244 1
0.792816],
             mask = [False False False ..., False False False],
       fill_value = 1e+20)
```

AstroQuery

- Simbad: Basic data, cross-identifications, bibliography and measurements for astronomical objects outside the solar system.
- Vizier: Set of 11,000+ published, multiwavelength catalogues hosted by the CDS.
- IRSA dust: Galactic dust reddening and extinction maps from IRAS 100 um data.
- NED: NASA/IPAC Extragalactic Database. Multiwavelength data from both surveys and publications.
- IRSA: NASA/IPAC Infrared Science Archive. Science products for all of NASA's infrared and sub-mm missions.
- UKIDSS: UKIRT Infrared Deep Sky Survey. JHK images of 7500 sq deg. in the northern sky.
- MAGPIS: Multi-Array Galactic Plane Imaging Survey. 6 and 20-cm radio images of the Galactic plane from the VLA.
- NRAO: Science data archive of the National Radio Astronomy Observatory. VLA, JVLA, VLBA and GBT data products.
- Besancon: Model of stellar population synthesis in the Galaxy.
- NIST: National Institute of Standards and Technology (NIST) atomic lines database.
- Fermi: Fermi gamma-ray telescope archive.
- SDSS: Sloan Digital Sky Survey data, including optical images, spectra, and spectral templates.
- Alfalfa: Arecibo Legacy Fast ALFA survey; extragalactic HI radio data.
- SHA: Spitzer Heritage Archive; infrared data products from the Spitzer Space Telescope
- Lamda: Leiden Atomic and Molecular Database; energy levels, radiative transitions, and collisional rates for - astrophysically relevant atoms and molecules.
- Ogle: Optical Gravitational Lensing Experiment III; information on interstellar extinction towards the Galactic bulge.
- Splatalogue: National Radio Astronomy Observatory (NRAO)-maintained (mostly) molecular radio and millimeter line list service.
- CosmoSim: The CosmoSim database provides results from cosmological simulations performed within different projects: the MultiDark project, the BolshoiP project, and the CLUES project.
- ESO Archive
- ALMA Archive
- GAMA database
- NVAS archive
- Open Expolanet Catalog (OEC)

In [7]:

```
from astroquery.simbad import Simbad

result_table = Simbad.query_object("m31")

result_table.pprint(show_unit=True)
```

```
MAIN_ID      RA          DEC      ... COO_WAVELENGTH     COO_BIBCODE
          "h:m:s"      "d:m:s"    ...
------- ------------ ------------ ... -------------- -------------------
  M  31 00 42 44.330 +41 16 07.50 ...              I 2006AJ....131.1163S
```

In [8]:

```
from astroquery.vizier import Vizier

catalog_list = Vizier.find_catalogs('LAMOST')
```

In [9]:

```
print({k:v.description for k,v in catalog_list.items()})
```

```
{'J/ApJ/798/110': 'Equivalent widths of LAMOST metal-poor stars (Li+, 201
5)', 'J/other/RAA/15.1424': 'LAMOST luminous infrared galaxies (Lam+, 201
5)', 'J/AJ/145/159': 'LAMOST. II. ugriz photometry of 526 new quasars (Huo+,
 2013)', 'J/other/RAA/15.1154': 'M-giant star candidates in LAMOST DR 1 (Zho
ng+, 2015)', 'J/A+A/570/A107': 'WDMS from LAMOST DR1 (Ren+, 2014)', 'J/othe
r/RAA/15.1197': 'LAMOST DR2 star clusters candidate members (Zhang+, 2015)',
 'J/MNRAS/448/822': 'LAMOST candidate members of star clusters (Xiang+, 201
5)', 'J/other/RAA/15.1438': 'LAMOST new QSOs in M31 and M33 vicinity (Huo+,
 2015)', 'J/other/RAA/11.924': 'Atmospheric parameters for 771 stars (Wu+, 2
011)', 'J/other/RAA/15.1325': 'Be stars in LAMOST DR1 (Lin+, 2015)', 'J/ApJ
S/220/19': 'LAMOST obs. in the Kepler field. I. (De Cat+, 2015)', 'J/MNRAS/4
49/1401': 'Am stars candidates from LAMOST DR1 (Hou+, 2015)', 'V/146': 'LAMO
ST DR1 catalogs (Luo+, 2015)', 'J/MNRAS/452/765': 'White dwarf candidates us
ing LAMOST DR3 (Gentile Fusillo+, 2015)', 'J/other/RAA/15.1392': 'LAMOST glo
bular clusters in M 31 and M 33 (Chen+, 2015)', 'J/AJ/152/6': 'Parameters of
 Kepler stars using LAMOST & seismic data (Wang+, 2016)', 'J/AJ/150/42': 'Ca
talog of 2612 M dwarfs from LAMOST (Zhong+, 2015)', 'J/AJ/150/187': 'Abundan
ces and stellar parameters of LAMOST stars (Lee+, 2015)', 'J/other/RAA/15.11
82': 'M Dwarf catalog of LAMOST DR1 (Guo+, 2015)', 'J/other/RAA/15.1414': 'E
+A galaxy candidates in LAMOST DR2 (Yang+, 2015)', 'J/AJ/146/34': 'Infrared
 photometry of DA white dwarfs from LAMOST (Zhang+, 2013)'}
```

In [10]:

```
catalog_list.keys()
```

Out[10]:

```
dict_keys(['J/ApJS/220/19', 'J/AJ/145/159', 'J/other/RAA/15.1154', 'J/A+A/57
0/A107', 'J/other/RAA/15.1197', 'J/MNRAS/449/1401', 'J/other/RAA/15.1424',
 'J/other/RAA/15.1438', 'J/MNRAS/448/822', 'J/other/RAA/11.924', 'J/other/RA
A/15.1325', 'J/ApJ/798/110', 'J/AJ/146/34', 'J/MNRAS/452/765', 'J/other/RAA/
15.1392', 'V/146', 'J/AJ/152/6', 'J/AJ/150/42', 'J/AJ/150/187', 'J/other/RA
A/15.1182', 'J/other/RAA/15.1414'])
```

In [11]:

```
from astropy.coordinates import Angle

result = Vizier.query_region("3C 273", radius=Angle(0.1, "deg"), catalog='GSC')
```

In [12]:

```
print(result)
```

```
TableList with 3 tables:
	'0:I/254/out' with 15 column(s) and 17 row(s)
	'1:I/271/out' with 23 column(s) and 50 row(s)
	'2:I/305/out' with 35 column(s) and 50 row(s)
```

In [13]:

```
print(result['I/271/out'])
```

```
 _RAJ2000   _DEJ2000     _r       GSC2.2    ...   e    aPA  Status  Np
   deg        deg       deg                 ...        deg              
---------- ---------- -------- ------------ ... ----- ----- ------ ---
187.232068   2.126962 0.087503 N12001333545 ...  0.27 130.3  10102   4
187.267540   2.147471 0.095629 N12001333616 ...  0.15 139.2  10112   4
187.292515   2.149536 0.098223 N12001333619 ...  0.17 118.0    102   4
187.183862   2.031656 0.096244 N12001333250 ...  0.41  99.5  10012   4
187.210711   2.012360 0.078183 N12001333173 ...  0.19  47.2  10102   4
187.216187   2.100738 0.078358 N12001333477 ...  0.48  22.1  10112   4
187.189455   2.086431 0.094713 N12001333440 ...  0.10 172.7  10112   4
187.206795   2.075025 0.074575   N120013337 ...  0.02 122.9  10112   4
187.193045   2.095777 0.095250   N120013334 ...  0.04  54.1  10112   4
187.212103   2.108647 0.086528   N120013333 ...  0.04  30.5  10112   4
       ...        ...      ...          ... ...   ...   ...    ... ...
187.241476   1.960722 0.098646 N12001332978 ...  0.07  11.0  10112   4
187.247385   1.958396 0.098832 N12001332971 ...  0.14 169.4    112   4
187.289263   1.976185 0.077060 N12001335948 ...  0.58 100.6 100102   1
187.304849   2.002505 0.056702   N120013351 ...  0.11 163.6  10112   4
187.284938   2.005146 0.047779   N120013348 ...  0.08 131.7  10112   4
187.249944   2.031602 0.034834 N12001333248 ...  0.10  85.1    112   4
187.254497   2.013624 0.045289 N12001333178 ...  0.07 147.8  10102   4
187.273495   2.009878 0.042754 N12001333164 ...  0.08  14.7  10112   4
187.302838   2.008650 0.050353 N12001333162 ...  0.11 123.1  10112   4
187.306472   2.007573 0.053151 N12001333156 ...  0.34  42.2  10102   4
Length = 50 rows
```

In [14]:

```
result = Vizier.query_object("HD 226868", catalog=["NOMAD", "UCAC", 'LAMOST'])

print(result)
```

```
TableList with 3 tables:
	'0:I/297/out' with 29 column(s) and 50 row(s) 
	'1:I/289/out' with 27 column(s) and 18 row(s) 
	'2:I/322A/out' with 64 column(s) and 28 row(s) 
```

In [15]:

```
print(result['I/297/out'])
```

```
 _RAJ2000   _DEJ2000    _r      NOMAD1    ...  Hmag   Kmag  Xflags  R
   deg        deg      arcm               ...  mag    mag
---------- ---------- ------ ------------ ... ------ ------ ------ ---
299.553972  35.206047 1.8017 1252-0387738 ... 14.092 13.704  00001
299.557761  35.208218 1.6446 1252-0387753 ... 16.016 15.262  00800
299.557850  35.213869 1.7535 1252-0387754 ... 14.425 14.135  00001
299.560278  35.208417 1.5283 1252-0387769 ...     --     --  00001
299.560997  35.211502 1.5552 1252-0387771 ... 15.935 15.471  00800
299.561232  35.208408 1.4832 1252-0387773 ... 15.351 15.047  00802
299.561497  35.201692 1.4130 1252-0387776 ... 13.640 13.465  00001
299.562633  35.207706 1.4057 1252-0387784 ... 15.294 14.896  00802
299.562731  35.218836 1.7023 1252-0387786 ... 15.093 14.954  00003
299.564331  35.220031 1.6867 1252-0387794 ... 15.266 15.041  00003
       ...        ...    ...          ... ...    ...    ...    ... ...
299.581908  35.201578 0.4122 1252-0387887 ...     --     --  00201
299.582021  35.208511 0.5806 1252-0387888 ... 15.577 15.384  00802
299.582069  35.223708 1.3864 1252-0387890 ... 15.115 14.910  00401
299.582094  35.230200 1.7624 1252-0387891 ...     --     --  00001
299.584617  35.234339 1.9838 1252-0387898 ...     --     --  00001
299.584681  35.220561 1.1704 1252-0387899 ... 15.060 14.686  00403
299.584958  35.201653 0.2627 1252-0387900 ...     --     --  00201
299.585102  35.209595 0.5433 1252-0387902 ... 15.539 15.384  00802
299.586356  35.200542 0.2044 1252-0387908 ... 14.480 14.118  00802
299.586801  35.216694 0.9216 1252-0387910 ... 15.394 14.444  00802
Length = 50 rows
```

In [16]:

```
agn = Vizier(catalog="VII/258/vv10").query_constraints(Vmag="10.0..11.0")[0]

agn.pprint()
```

```
_RAJ2000 _DEJ2000  Cl  nR      Name     ...  U-B   Mabs  FC  r_Vmag r_z
  deg      deg                          ...  mag   mag
-------- -------- --- --- ------------- ... ----- ----- ---- ------ ----
 10.6846  41.2694   Q               M 31 ...  0.73    -- 1959   1978 1936
 60.2779 -16.1108   Q     NPM1G-16.0168 ...    -- -24.8   --     --  988
 27.2387   5.9067   A   *      NGC  676 ...    -- -20.6   --     -- 1034
 40.6696  -0.0131   A          NGC 1068 ...  0.13 -18.8 1973    592   58
139.7596  26.2697   A          NGC 2824 ...    -- -21.8   --     -- 2528
147.5921  72.2792   A          NGC 2985 ...    -- -19.8 1973   1377 1033
173.1442  53.0678   A          NGC 3718 ...    -- -19.2 2619   1377 1033
184.9608  29.6139   A         UGC  7377 ...    -- -19.1   --     -- 2500
185.0287  29.2808   A          NGC 4278 ...  0.51 -17.8 1973    590 1033
186.4537  33.5467   A          NGC 4395 ...    -- -17.3 1973   1377 1033
192.7196  41.1194   A          NGC 4736 ...  0.52 -16.4 1973   1377 1032
208.3612  40.2831   A          NGC 5353 ...    -- -21.2   --     --  368
```

In [17]:

```
guide = Vizier(catalog="II/246", column_filters={"Kmag":"<9.0"}).query_region(agn,
radius="30s", inner_radius="2s")[0]
guide.pprint()
```

```
 _q  _RAJ2000   _DEJ2000    _r    RAJ2000   ... scanKey coaddKey coadd Opt
       deg        deg      arcs     deg     ...                          
--- ---------- ---------- ------ ---------- ... ------- -------- ----- ---
  1  10.686015  41.269630  3.917  10.686015 ...   69157  1590591    33 Opt
  1  10.685657  41.269550  2.911  10.685657 ...   69157  1590591    33 Opt
  1  10.685837  41.270599  5.462  10.685837 ...   69157  1590591    33 Opt
  1  10.683263  41.267456  7.878  10.683263 ...   69157  1590591    33 Opt
  1  10.683465  41.269676  3.228  10.683465 ...   69157  1590591    33 Opt
  3  27.238636   5.906066  2.294  27.238636 ...   59940  1378619   256 Opt
  4  40.669277  -0.014225  4.214  40.669277 ...    7053   162197     9 Opt
  4  40.668802  -0.013064  2.876  40.668802 ...    7053   162197     9 Opt
  4  40.669219  -0.012236  3.399  40.669219 ...    7053   162197     9 Opt
  4  40.670761  -0.012208  5.271  40.670761 ...    7053   162197     9 Opt
  4  40.670177  -0.012830  2.293  40.670177 ...    7053   162197     9 Opt
 11 192.721982  41.121040  8.751 192.721982 ...   13349   307008    44 Opt
 11 192.721179  41.120201  5.163 192.721179 ...   13349   307008    44 Opt
```

SDSS

In [19]:

```
from astroquery.sdss import SDSS

from astropy import coordinates as coords
```

```
/usr/local/lib/python3.5/site-packages/astroquery/sdss/__init__.py:28: UserW
arning: Experimental: SDSS has not yet been refactored to have its API match
 the rest of astroquery (but it's nearly there).
  warnings.warn("Experimental: SDSS has not yet been refactored to have its
 API "
```

In [20]:

```
pos = coords.SkyCoord('0h8m05.63s +14d50m23.3s', frame='icrs')
xid = SDSS.query_region(pos, spectro=True)
print(xid)
```

```
      ra           dec             objid        ... run2d instrument
------------- ------------- ------------------- ... ----- ----------
2.02344596303 14.8398237521 1237652943176138868 ...    26       SDSS
```

In [21]:

```
sp = SDSS.get_spectra(matches=xid)
im = SDSS.get_images(matches=xid, band='g')
```

In [35]:

```
print(sp[0][1].data)
```

```
[(30.596626, 3.5797, 0.064408027, 0, 0, 1.2189666, 8.154254, 36.077015)
 (33.245728, 3.5797999, 0.0, 0, 0, 1.2187515, 7.656426, 34.997238)
 (35.895119, 3.5799, 0.062928334, 0, 0, 1.2185355, 7.2311668, 35.379208)
 ..., (53.27969, 3.9635, 0.2728394, 0, 0, 0.64196426, 4.1553526, 50.136108)
 (50.236168, 3.9635999, 0.28062949, 0, 0, 0.64184296, 4.2312737, 50.033169)
 (51.702717, 3.9637001, 0.18243204, 0, 33554432, 0.64171964, 4.3413963, 50.2
08874)]
```

In [29]:

```
im
```

Out[29]:

```
[[<astropy.io.fits.hdu.image.PrimaryHDU at 0x1054baeb8>,
  <astropy.io.fits.hdu.image.ImageHDU at 0x1054c1358>,
  <astropy.io.fits.hdu.table.BinTableHDU at 0x1054c1a20>,
  <astropy.io.fits.hdu.table.BinTableHDU at 0x1054c8358>]]
```

In [32]:

```
im[0].info()
```

```
Filename: (No file associated with this HDUList)
No.    Name         Type      Cards   Dimensions   Format
0    PRIMARY     PrimaryHDU      85   (2048, 1489)   float32
1                ImageHDU         6   (2048,)      float32
2                BinTableHDU     27   1R x 3C      [49152E, 2048E, 1489E]
3                BinTableHDU     79   1R x 31C     [J, 3A, J, A, D, D, 2J,
 J, D, D, D, D, D, D, D, D, D, D, D, D, D, D, D, D, D, D, D, D, D, E, E]
```

In [18]:

```
help(Vizier.query_object)
```

```
Help on function query_object in module astroquery.utils.process_asyncs:

query_object(self, *args, **kwargs)
    Queries the service and returns a table object.

    Serves the same purpose as `query_object` but only
    returns the HTTP response rather than the parsed result.

    Parameters
    ----------
    object_name : str
        The name of the identifier.
    catalog : str or list, optional
        The catalog(s) which must be searched for this identifier.
        If not specified, all matching catalogs will be searched.
    radius : `astropy.unit.Unit` or None
        A degree-equivalent unit (optional)
    coordinate_system : str or None
        If the object name is given as a coordinate, you *should* use
        `query_region`, but you can specify a coordinate frame here
        instead (today, J2000, B1975, B1950, B1900, B1875, B1855,
        Galactic, Supergal., Ecl.J2000, )


    Returns
    -------
    table : A `~astropy.table.Table` object.
```

In [37]:

```
from astropy import units as u
from astroquery.xmatch import XMatch

table = XMatch.query(cat1=open('data/list.txt'),
                     cat2='vizier:II/246/out',
                     max_distance=5 * u.arcsec, colRA1='ra',
                     colDec1='dec')
```

In [38]:

```
print(table)
```

```
angDist      ra       dec          2MASS       ... Qfl Rfl  X   MeasureJD
-------- --------- --------- ---------------- ... --- --- --- ------------
1.352044 267.22029 -20.35869 17485281-2021323 ... EEU 226   2 2450950.8609
1.578188 267.22029 -20.35869 17485288-2021328 ... UUB 662   2 2450950.8609
3.699368 267.22029 -20.35869 17485264-2021294 ... UUB 662   2 2450950.8609
3.822922 267.22029 -20.35869 17485299-2021279 ... EBA 222   2 2450950.8609
4.576677 267.22029 -20.35869 17485255-2021326 ... CEU 226   2 2450950.8609
0.219609 274.83971 -25.42714 18192154-2525377 ... AAA 211   0 2451407.5033
1.633225 275.92229 -30.36572 18234133-3021582 ... EEE 222   2 2451021.7212
0.536998 283.26621  -8.70756 18530390-0842276 ... AAA 222   0 2451301.7945
1.178542 306.01575  33.86756 20240382+3352021 ... AAA 222   0 2450948.9708
0.853178   322.493  12.16703 21295836+1210007 ... EEA 222   0 2451080.6935
 4.50395   322.493  12.16703 21295861+1210023 ... EEE 222   0 2451080.6935
```

In [39]:

```
!cat data/list.txt
```

```
ra,dec
267.22029,-20.35869
274.83971,-25.42714
275.92229,-30.36572
283.26621,-8.70756
306.01575,33.86756
322.493,12.16703
```

In [41]:

```
table.columns
```

Out[41]:

```
TableColumns([('angDist',
               <MaskedColumn name='angDist' dtype='float64' length=11>
               1.352044
               1.578188
               3.699368
               3.822922
               4.576677
               0.219609
               1.633225
               0.536998
               1.178542
               0.853178
                4.50395),
              ('ra', <MaskedColumn name='ra' dtype='float64' length=11>
               267.22029
               267.22029
               267.22029
               267.22029
               267.22029
               274.83971
               275.92229
               283.26621
               306.01575
                 322.493
                 322.493),
              ('dec', <MaskedColumn name='dec' dtype='float64' length=11>
               -20.35869
               -20.35869
               -20.35869
               -20.35869
               -20.35869
               -25.42714
               -30.36572
                -8.70756
                33.86756
                12.16703
                12.16703),
              ('2MASS', <MaskedColumn name='2MASS' dtype='str16' length=11>
               17485281-2021323
               17485288-2021328
               17485264-2021294
               17485299-2021279
               17485255-2021326
               18192154-2525377
               18234133-3021582
               18530390-0842276
               20240382+3352021
               21295836+1210007
               21295861+1210023),
              ('RAJ2000',
               <MaskedColumn name='RAJ2000' dtype='float64' length=11>
               267.220049
               267.220348
               267.219344
               267.220825
               267.218994
               274.839773
               275.922233
               283.266284
               306.015944
               322.493171
```

```
 322.494242),
('DEJ2000',
 <MaskedColumn name='DEJ2000' dtype='float64' length=11>
  -20.35899
 -20.359125
 -20.358171
 -20.357754
 -20.359064
 -25.427162
 -30.366171
   -8.70769
  33.867275
  12.166862
  12.167332),
('errHalfMaj',
 <MaskedColumn name='errHalfMaj' dtype='float64' length=11>
 0.15
 0.14
 0.13
 0.08
 0.14
 0.06
 0.08
 0.06
 0.06
  0.1
  0.1),
('errHalfMin',
 <MaskedColumn name='errHalfMin' dtype='float64' length=11>
 0.11
 0.12
 0.12
 0.07
 0.13
 0.06
 0.08
 0.06
 0.06
 0.08
 0.08),
('errPosAng',
 <MaskedColumn name='errPosAng' dtype='int64' length=11>
  16
 158
  33
   7
  87
  90
  55
  45
  90
 179
   1),
('Jmag', <MaskedColumn name='Jmag' dtype='float64' length=11>
  9.931
  8.868
  9.562
 10.756
 10.431
  9.368
 12.947
```

```
              12.182
              13.575
               9.798
              10.057),
             ('Hmag', <MaskedColumn name='Hmag' dtype='float64' length=11>
               8.822
               7.784
               8.525
               9.725
               9.348
               8.431
              12.334
              11.534
              12.684
               9.339
                9.72),
             ('Kmag', <MaskedColumn name='Kmag' dtype='float64' length=11>
                7.55
                8.53
               9.445
               9.287
               7.926
               7.919
              12.145
               11.38
              12.321
               9.176
               9.483),
             ('e_Jmag', <MaskedColumn name='e_Jmag' dtype='float64' length=
11>
              0.239
                 --
                 --
              0.139
              0.159
              0.024
              0.156
              0.057
              0.025
              0.109
              0.077),
             ('e_Hmag', <MaskedColumn name='e_Hmag' dtype='float64' length=
11>
              0.241
                 --
                 --
              0.127
              0.316
              0.044
              0.221
              0.071
              0.027
               0.15
              0.136),
             ('e_Kmag', <MaskedColumn name='e_Kmag' dtype='float64' length=
11>
                 --
              0.128
              0.119
              0.103
                 --
```

```
 0.036
 0.127
 0.063
 0.026
   0.1
 0.088),
('Qfl', <MaskedColumn name='Qfl' dtype='str3' length=11>
 EEU
 UUB
 UUB
 EBA
 CEU
 AAA
 EEE
 AAA
 AAA
 EEA
 EEE),
('Rfl', <MaskedColumn name='Rfl' dtype='int64' length=11>
 226
 662
 662
 222
 226
 211
 222
 222
 222
 222
 222),
('X', <MaskedColumn name='X' dtype='int64' length=11>
 2
 2
 2
 2
 2
 0
 2
 0
 0
 0
 0),
('MeasureJD',
 <MaskedColumn name='MeasureJD' dtype='float64' length=11>
 2450950.8609
 2450950.8609
 2450950.8609
 2450950.8609
 2450950.8609
 2451407.5033
 2451021.7212
 2451301.7945
 2450948.9708
 2451080.6935
 2451080.6935)])
```

In [ ]:

```

```