用 Python 解决科学问题的实践指南

众多世界知名大学教授推荐阅读

异步图书
www.epubit.com.cn

Python 物理建模初学者指南

[美]Jesse M. Kinder　[美] Philip Nelson 著

盖磊 译

中国工信出版集团

人民邮电出版社
POSTS & TELECOM PRESS

Python 物理建模初学者指南

[美]Jesse M. Kinder [美] Philip Nelson 著

盖磊 译

人民邮电出版社

北京

图书在版编目（CIP）数据

Python物理建模初学者指南 /（美）金德（Jesse M. Kinder），（美）尼尔森（Philip Nelson）著；盖磊 译. -- 北京 : 人民邮电出版社，2017.11
ISBN 978-7-115-46541-2

Ⅰ. ①P… Ⅱ. ①金… ②尼… ③盖… Ⅲ. ①物理学－建模系统－程序设计 Ⅳ. ①O4-39

中国版本图书馆CIP数据核字(2017)第203248号

版权声明

◆ 著 [美] Jesse M. Kinder Philip Nelson
译 盖 磊
责任编辑 陈冀康
责任印制 焦志炜

◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路 11 号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
大厂聚鑫印刷有限责任公司印刷

◆ 开本：720×960 1/16
印张：12.75
字数：231 千字 2017 年 11 月第 1 版
印数：1－2 000 册 2017 年 11 月河北第 1 次印刷

著作权合同登记号 图字：01-2016-4645 号

定价：59.00 元

读者服务热线：(010)81055410 印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

内容提要

Python是一种得到广泛应用的编程语言。本书旨在帮助Python学习者掌握足够的Python编程技能以进行物理建模。

全书分为8章和5个附录，包括Python基础知识、数据结构与程序控制、数据输入和输出、Python高级知识和高级技术等，其中贯穿了三次不同方向和难度的物理建模上机实验。附录部分介绍了Python的安装、错误消息、版本差异以及可供深入学习的话题。

本书适合Python初学者阅读，尤其适合想要用Python进行科学计算和物理建模的读者学习参考。

前言

为什么要自学Python？为什么要用本书的方法自学？

学习计算机编程可改变你的思维。一开始你会感到步履维艰，三三两两地随处收集编程技巧，但是假以时日，你会发现自己能让计算机近乎无所不能。你可以添加常被物理学教授说成是可忽略不计的摩擦力和空气阻力的影响，你可以模拟出自己的捕食者——猎物生态去研究种群模型，你可以创建出自己的分形，你可以发现股票市场中的关联关系，不胜枚举。

要与计算机进行交流，你必须先学会一门计算机可以理解的语言。Python就是一个优秀的选择，它易于入门，结构非常自然，至少相比于其他的计算机语言是这样的。不久，你将发现大量时间不必再花费在如何向计算机解释你的计算方法，而是用于思考如何解决问题。无论你是出于什么动机学习Python，你都会想到这样的一个问题：是否确有必要费力读完本书中的所有内容。请相信我们！作为工作在一线的科学家，为了使读者能开始独自探索和学习，我们用自身经验准备好了尽量高效的方法。如果你能按本书推荐的顺序花费一些时间尝试书中推荐的所有内容，从长远看会节省学习Python的时间。在一开始写书时，我们就去除了所有读者不需要的内容，书中保留下来的是一组基本知识和技能。总有一天，你会发现这些内容的确有用。

如何使用本书

对于如何使用本书自学Python，下面给出一些建议：

- 可在http://epubit.com.cn获取本书中提供的一些代码例子、勘误、更新、数据集，还有更多的内容。
- 在读完前面的几页后，需要读者手边有一台能运行Python的计算机（不用担心，本书会教你如何设置该计算机）。可能书中会要求在该计算机上打开一个名为code_samples.txt的文本文档，该文档同样可从上面给出的Web地址获取。
- 最好将本书的纸质版放在计算机旁，其他可看电子书的设备，例如平板电脑也可。当然也可用运行Python的同一台电脑查看电子书。
- 本书将频繁地要求读者去尝试操作。其中的一些操作涉及文本形式的代码段。读者可以从code_samples.txt文件中复制并粘贴代码到自己的Python会话中，查看输出，并更改和操作代码。
- 读者也可交互地访问代码段。上面给出的Web站点还提供了访问每个代码例子的链接。可从Web页面复制和粘贴代码到Python中。
- 部分章节标注了①这样的标识。该标识用于指示更高级的内容，在首次阅读时可以跳过。

那好，我们现在开始吧！

目录

第1章 Python入门

分析机编织代数模式，就像提花织机在编织花和叶。

——埃达·洛夫莱斯伯爵夫人，1815－1853

1.1 算法与算法思想

本书的目的在于让读者运用计算机语言Python开始计算科学之旅。Python是一种开源软件，读者可自行下载、安装并使用它。现在好的Python入门教程不胜枚举，并且每年还在推陈出新。与这些内容相比，本书的独到之处在于侧重能解决物理建模问题的有用技能。

对物理系统建模或许会是一项十分复杂的任务。下面让我们了解一下功能强大的计算机处理器是如何为此提供帮助的。

1.1.1 算法思想

假设你需要指导一位朋友完成倒车入位操作。当时是一个紧急情况，必须由你这位从未开过车的朋友完成操作，开始操作前，你们之间只能通过电话联系。

你需要将操作分解为可被你的朋友能理解的小步骤，这些小步骤应是明确的，依次执行即可完成任务。例如，你可给出如下一系列指令：

```
1  将车钥匙插入点火器。
2  转动钥匙直至启动，然后松开钥匙。
3  按下变速杆上的按钮，将变速挂入“倒车”标识的档位。
4  ……
```

遗憾的是，即使你的朋友理解了每条指令，该“代码”对一些车辆并不起作用。这个过程有“程序故障”。在做第3步指令前，不少车辆需要驾驶者：

```
踩下左脚踏板。
```

此外，变速器上的倒车档可能是用“R”标记的，而非“倒车”。创建这样的操作指令时，难以做到一开始就习惯操作所需的高精确度。

因为指令是预先给出的（假定你的朋友没有手机），所以好的做法是允许存在意外情况：

```
在听到了吱嘎声后，踩下左踏板……
```

这就是算法思想的开端，将长的操作步骤分解为小的、清晰的子步骤，以及预期中的意外情况。

如果你的朋友见过别人驾车，并已有了大量的经验，那么上述指令足以适用。但是对于另外一些没有任何经验的朋友，甚至是机器人，需要提供更多的细节。例如，最初两步指令可能需要进行如下扩展：

```
握住钥匙的大头端。
将钥匙的另一端插入位于驾驶杆右下方的钥匙孔内。
按顺时针方向扭动钥匙（从钥匙的大头端向对端的角度看）。
……
```

低层计算机程序使用计算机可理解的语言形式提供类似的指令[①]。而高层系统可以理解很多通用任务，因而编程方式更为简明扼要，就像在上面例子中第一次所给出的指令。Python就是一种高层语言，它具有进行数学计算、文本处理、文件操作的通用操作命令。此外，Python还可以访问很多标准程序库。作为程序的集合，这些程序库可以执行数据可视化、图像处理等高级功能。

Python也提供命令行接口，即一个执行在其中输入的Python命令的程序。这样在Python中，你可以输入命令并立刻执行。与之形成对比的是，用C、C++、Fortran等很多科学计算中使用的其他语言所编写的程序，在执行前需要进行编译。这时要先由一个称为编译器的独立程序将你的代码转译为低级语言，然后你才能运行生成的编译程序去执行（实现）你的算法。使用Python，相对容易实现程序的快速编写、运行和调试（虽然依然需要耐心和实践）。

命令行解释器，连同标准函数库和你自己编写的程序，一起提供了便利的、强大的科学计算平台。

1.1.2 状态

也许以前你在上几何课时，就已经学过了如何做多步数学证明。这种证明方法的目标在于通过依次诉诸已知的信息和形式化系统去证实一个期望结论的真实性。这样，虽然孤立看来每个语句的真实性不明显，综合考虑前面的语句，就可以明确地判定期望的真实性。在通读证明过程中，读者的“状态”（已知为真的命题列表）会发生改变，最终给出了一个如何从公理和假设引出结果的完整逻辑推导链条。

每个算法都具有各自的目标。可以将算法看成是一个指令链，其中的每条指令都描述了一个基本操作，这些操作共同实现了一个复杂的任务。指令链中可能包括不少重复的操作，因此你不会想要去对执行中的每一步进行管理。替代的做法是，你预先制订了所有步骤，然后观看你的电子助理是如何快速执行的。其中可能存在一些预料之外的事件（例如听到汽车吱嘎作响……）。

在算法中，计算机的状态时常会发生改变。例如，一个计算机有多个内存单元，其中的内容会在操作过程中发生变化。你的目标可能是安排一到多

① 机器码和汇编语言都是低层编程语言。

个这样的单元，用于在完成复杂计算后保存一些运行的结果。你可能还希望能绘制出特定的图形图像。

1.1.3 “a=a+1”是什么意思

为使计算机执行你的算法，必须用计算机能理解的语言与计算机进行通信。一开始接触计算机编程时，你可能会对所使用的命令感到困惑，尤其是这些命令与标准数学用法有冲突的时候。例如，很多编程语言（其中包括Python）接受下面这样的声明：

```
1  a = 100
2  a = a + 1
```

这在数学中是不合理的。第二行是一个永假断言，相应地也是一个无解的等式。但是对于Python而言，“=”并非是测试相等性，而是一个需要执行的指令。上面两行命令的大概意思为[①]：

1. 命名整数对象a，并赋值100。

2. 提取命名对象a的值，并与1做求和运算，然后将运算结果赋于a，并抛弃a所指代对象的原始值。

换句话说，等号“=”通知Python去更改变量自身的状态。相反，等号在数学概念中是用于创建一个结果为真或为假的命题。还应注意的是，Python对命令“x=y”中等号两侧内容的处理不同，而在数学中等号两侧是对称的。例如，Python将对“b+1=a”这样的命令报错，因为赋值语句的左侧必须是一个变量名称，该变量可被赋值为右侧表达式的求值。

我们往往希望能确定一个变量是否具有特定的值。为避免赋值和等价测试间的模糊性，Python以及其他很多语言都对后者使用双等号“==”。例如：

```
1  a = 1
2  a == 0
3  b = (a == 1)
```

① 关于赋值语句的处理，在附录D中给出了更为详细的信息。

上面的代码再次设置了变量a，并对a赋了一个数值。然后将该数值与0进行比较。最后，创建了第二个变量b。在完成另一次比较运算后，将变量b赋予一个逻辑值（**True**或**False**）。该值可在或然代码中使用，我们将在本书的随后内容中介绍。

注意：不要在应该使用“==”（等价测试）的地方使用“=”（赋值）。

这是编程新手常犯的一个错误。因为“=”和“==”都是合法的Python语法，这个错误会产生无法预料的结果。但是无论在何种情况下，“=”和“==”两者中只会有一个是你所需要的。

1.1.4 符号和数字的对比

在不知道a值的情况下使用“假定$b=a^2-a$”进行减法运算，这在数学上是完全合理的。该语句将会根据a定义b，无论a的值是什么。

如果我们启动Python并直接给出上面运算的一个等价语句“`b=a**2-a`”，这会报错[①]。每次按下<Return/Enter>键之后，Python会尝试对每个赋值语句求值。如果变量尚未赋值，那么求值就会失败，Python就会报错。这样的输入可能会被其他的数学计算软件包接受，因为它们可以追踪符号的关系并随后求值，但是基本Python并不会这样做[②]。

在数学中可理解的是，类似于“假定$b=a^2-a$”这样的定义在整个问题讨论期间都不会发生改变。如果我们说“当a=1时”，那么读者就知道b等于0；如果随后我们说“当a=2时，……”，我们就不必再次定义b，读者知道b现在代表的值是$2^2-2=2$。

与此相对比，Python这样的数学系统在执行赋值语句“`b=a**2-a`”后，并不会记住b和a之间的关系，它所记住的只是赋给b的值。如果我们随后更改了a的值，b的值并不会发生变化[③]。

一般来说，在证明的过程中改变符号关系并非一个好做法。但是在Python

① “**”符号表示幂运算，参见第1.4.2节。
② SymPy库为Python提供了符号计算功能，参见第7.3.1节。
③ 在数学中，表达式$b=a^2-a$本身就定义了b是a的一个函数。在Python中，我们当然也可以这样做，定义一个返回a^2-a值的函数，并将该函数赋值给b。关键在于这并非是“=”的用法。

中，如果我们说“b=a**2-a”，我们稍后也可以说“b=2**a”。第二次赋值会丢弃第一次赋值时计算的值，并替换为新计算的值，这样更新了Python的状态。

1.2 启动Python

与其在书中看输入命令后会发生什么，不如亲自动手尝试命令。从现在开始，你应该在阅读本书的同时运行Python，尝试每个代码段，观察Python的响应情况。例如，本书不会显示任何图形或输出，你必须通过运行例子自己生成。

> 阅读本书并不能教会你如何使用Python编程。要通过操作书中的所有例子和练习来自学Python，进而灵活使用所学的内容。

为自己设置一些小的挑战，并尝试着去解决他们（例如，“如果……，那么会发生什么？”“我如何能做到这样？”）。Python并非一堆昂贵的实验仪器，输入错误就会摔坏甚至爆炸！你应该勇于尝试。相比起被动的知识堆砌，该学习策略不仅能让你获得更多的乐趣，而且更为有效。

一个完整的Python编程环境中会包括很多组件，表1.1中列出了本书随后将会介绍到的组件。注意在本书中，“Python”一词的用法十分宽泛。除了指代语言本身之外，还可指代Python解释器。Python解释器是一个计算机应用程序，它接收命令并执行程序中所描述的步骤。此外，Python也可用于指代包括通用软件库在内的一种语言。

表1.1 本书中所用到的Python环境组件

组件名称	描　述
Python	一种计算机编程语言，用于向计算机描述算法
IPython	一种Python解释器：它是一个计算机应用，提供执行Python命令和程序的便利交互模式
Spyder	一个集成开发环境（IDE，Integrated Development Environment）：Spyder是一个包含IPython的计算机应用，可以进行变量检查，也是一个编写和调试程序的文本编辑器，还具有更多的功能

续表

组件名称	描　述
NumPy	一个提供数值数组和算术函数的标准函数库
PyPlot	一个提供了可视化工具的标准函数库
SciPy	一个提供了科学计算工具的标准函数库
Anaconda	Python的一个发布版本。它是一个包括上述所有组件的单一下载文件，提供对更多其他用于特殊目的函数库的访问，其中也包括维护所有组件版本为最新版本的软件包管理器

附录A中介绍了如何安装和启动Python。所提供的大部分代码都可运行于任何Python发布版。但是考虑到我们无法对每个可用Python版本和集成开发环境（IDE，Integrated Development Environment）给出建议，我们选定了下面的特定配置：

- 使用Python 3.4的Anaconda版，下载地址为https://store.continuum.io/cshop/anaconda/[①]。也有一些科学家使用了Python的早期版本（例如2.7版）。附录C中讨论了对于早期版本，本书中代码需要调整的两处小变化。
- Spyder IDE，它是随Anaconda一起发布的，也可在https://code.google.com/archive/p/spyderlib/处下载[②]。

任何编程任务都可使用不同的IDE实现，也可以完全不用IDE，本书假定读者使用的是Spyder，当然使用其他的IDE也可以。例如，随所有Python发布版一并提供的IDLE，以及基于Web的IPython Notebooks等。

版本选取是个人喜好问题。我们选定了Anaconda是因为它易于安装、更新和维护，并且是免费的。读者可以找到适合自身需要的不同发布版。Enthought Canopy就是另一种免费并广为使用的Python发布版。

1.2.1 IPython控制台

启动Spyder时，会打开一个包含多个窗格（pane）的窗口，如图1.1所示。

① 地址已跳转到https://www.continuum.io/downloads/。——译者注
② 由于Google Code策略的改变，该项目已迁移到Github，地址为https://github.com/spyder-ide/spyder/。——译者注