

1. Record Nr.	UNINA9910830429503321
Autore	Esmaili Rebekah Bradley
Titolo	Earth observation using Python : a practical programming guide / / Rebekah Bradley Esmaili
Pubbl/distr/stampa	Hoboken, New Jersey : , : AGU : , : Wiley, , [2021] ©2021
ISBN	1-119-60691-8 1-119-60689-6 1-119-60692-6
Descrizione fisica	1 online resource (300 pages)
Collana	Special publications series ; ; 75
Disciplina	550.2855133
Soggetti	Earth sciences - Data processing
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Nota di contenuto	Cover -- Title Page -- Copyright Page -- Contents -- Foreword -- Acknowledgments -- Introduction -- Part I Overview of Satellite Datasets -- Chapter 1 A Tour of Current Satellite Missions and Products -- 1.1 History of Computational Scientific Visualization -- 1.2 Brief Catalog of Current Satellite Products -- 1.2.1 Meteorological and Atmospheric Science -- 1.2.2 Hydrology -- 1.2.3 Oceanography and Biogeosciences -- 1.2.4 Cryosphere -- 1.3 The Flow of Data from Satellites to Computer -- 1.4 Learning Using Real Data and Case Studies -- 1.5 Summary -- References -- Chapter 2 Overview of Python -- 2.1 Why Python? -- 2.2 Useful Packages for Remote Sensing Visualization -- 2.2.1 NumPy -- 2.2.2 Pandas -- 2.2.3 Matplotlib -- 2.2.4 netCDF4 and h5py -- 2.2.5 Cartopy -- 2.3 Maturing Packages -- 2.3.1 xarray -- 2.3.2 Dask -- 2.3.3 Iris -- 2.3.4 MetPy -- 2.3.5 cfrib and eccodes -- 2.4 Summary -- References -- Chapter 3 A Deep Dive into Scientific Data Sets -- 3.1 Storage -- 3.1.1 Single Values -- 3.1.2 Arrays -- 3.2 Data Formats -- 3.2.1 Binary -- 3.2.2 Text -- 3.2.3 Self- Describing Data Formats -- 3.2.4 Table-Driven Formats -- 3.2.5 geoTIFF -- 3.3 Data Usage -- 3.3.1 Processing Levels -- 3.3.2 Product Maturity -- 3.3.3 Quality Control -- 3.3.4 Data Latency -- 3.3.5 Reprocessing -- 3.4 Summary -- References -- Part II Practical Python Tutorials for Remote Sensing -- Chapter 4 Practical Python Syntax --

4.1 "Hello Earth" in Python -- 4.2 Variable Assignment and Arithmetic
-- 4.3 Lists -- 4.4 Importing Packages -- 4.5 Array and Matrix
Operations -- 4.6 Time Series Data -- 4.7 Loops -- 4.8 List
Comprehensions -- 4.9 Functions -- 4.10 Dictionaries -- 4.11
Summary -- References -- Chapter 5 Importing Standard Earth Science
Datasets -- 5.1 Text -- 5.2 NetCDF -- 5.2.1 Manually Creating a Mask
Variable Using True and False Values.
5.2.2 Using NumPy Masked Arrays to Filter Automatically -- 5.3 HDF --
5.4 GRIB2 -- 5.5 Importing Data Using Xarray -- 5.5.1 netCDF -- 5.5.2
Examining Vertical Cross Sections -- 5.5.3 Examining Horizontal Cross
Sections -- 5.5.4 GRIB2 using Cfgrib -- 5.5.5 Accessing Datasets Using
OpenDAP -- 5.6 Summary -- References -- Chapter 6 Plotting and
Graphs for All -- 6.1 Univariate Plots -- 6.1.1 Histograms -- 6.1.2
Barplots -- 6.2 Two Variable Plots -- 6.2.1 Converting Data to a Time
Series -- 6.2.2 Useful Plot Customizations -- 6.2.3 Scatter Plots --
6.2.4 Line Plots -- 6.2.5 Adding Data to an Existing Plot -- 6.2.6
Plotting Two Side-by-Side Plots -- 6.2.7 Skew-T Log-P -- 6.3 Three
Variable Plots -- 6.3.1 Filled Contour Plots -- 6.3.2 Mesh Plots -- 6.4
Summary -- References -- Chapter 7 Creating Effective and Functional
Maps -- 7.1 Cartographic Projections -- 7.1.1 Geographic Coordinate
Systems -- 7.1.2 Choosing a Projection -- 7.1.3 Some Common
Projections -- 7.2 Cylindrical Maps -- 7.2.1 Global Plots -- 7.2.2
Changing Projections -- 7.2.3 Regional Plots -- 7.2.4 Swath Data --
7.2.5 Quality Flag Filtering -- 7.3 Polar Stereographic Maps -- 7.4
Geostationary Maps -- 7.5 Creating Maps from Datasets Using
OpenDAP -- 7.6 Summary -- References -- Chapter 8 Gridding
Operations -- 8.1 Regular One-Dimensional Grids -- 8.2 Regular Two-
Dimensional Grids -- 8.3 Irregular Two-Dimensional Grids -- 8.3.1
Resizing -- 8.3.2 Regridding -- 8.3.3 Resampling -- 8.4 Summary --
References -- Chapter 9 Meaningful Visuals through Data Combination
-- 9.1 Spectral and Spatial Characteristics of Different Sensors -- 9.2
Normalized Difference Vegetation Index (NDVI) -- 9.3 Window
Channels -- 9.4 RGB -- 9.4.1 True Color -- 9.4.2 Dust RGB -- 9.4.3.
Fire/Natural RGB -- 9.5 Matching with Surface Observations -- 9.5.1
With User-Defined Functions -- 9.5.2 With Machine Learning.
9.6 Summary -- References -- Chapter 10 Exporting with Ease -- 10.1
Figures -- 10.2 Text Files -- 10.3 Pickling -- 10.4 NumPy Binary Files
-- 10.5 NetCDF -- 10.5.1 Using netCDF4 to Create netCDF Files --
10.5.2 Using Xarray to Create netCDF Files -- 10.5.3 Following Climate
and Forecast (CF) Metadata Conventions -- 10.6 Summary -- Part III
Effective Coding Practices -- Chapter 11 Developing a Workflow --
11.1 Scripting with Python -- 11.1.1 Creating Scripts Using Text
Editors -- 11.1.2 Creating Scripts from Jupyter Notebook -- 11.1.3
Running Python Scripts from the Command Line -- 11.1.4 Handling
Output When Scripting -- 11.2 Version Control -- 11.2.1 Code Sharing
through Online Repositories -- 11.2.2 Setting up on GitHub -- 11.3
Virtual Environments -- 11.3.1 Creating an Environment -- 11.3.2
Changing Environments from the Command Line -- 11.3.3 Changing
Environments in Jupyter Notebook -- 11.4 Methods for Code
Development -- 11.5 Summary -- References -- Chapter 12
Reproducible and Shareable Science -- 12.1 Clean Coding Techniques
-- 12.1.1 Stylistic Conventions -- 12.1.2 Tools for Clean Code -- 12.2
Documentation -- 12.2.1 Comments and Docstrings -- 12.2.2 README
File -- 12.2.3 Creating Useful Commit Messages -- 12.3 Licensing --
12.4 Effective Visuals -- 12.4.1 Make a Statement -- 12.4.2 Undergo
Revision -- 12.4.3 Are Accessible and Ethical -- 12.5 Summary --
References -- Conclusion -- Appendix A Installing Python -- A.1.
Download Tutorials for This Book -- A.2. Download and Install

Anaconda -- A.3. Package Management in Anaconda -- Appendix B Jupyter Notebook -- B.1. Running on a Local Machine (New Coders) -- B.2. Running on a Remote Server (Advanced) -- B.3. Tips for Advanced Users -- B.3.1. Customizing Notebooks with Configuration Files -- B.3.2. Starting and Ending Python Scripts -- B.3.3. Creating Git Commit Templates.
Appendix C Additional Learning Resources -- Appendix D Tools -- D.1. Text Editors and IDEs -- D.2. Terminals -- Appendix E Finding, Accessing, and Downloading Satellite Datasets -- E.1. Ordering Data from NASA EarthData -- E.2. Ordering Data from NOAA/CLASS -- Appendix F Acronyms -- Index -- EULA.

Sommario/riassunto

"Python is a modern programming language that has exploded in popularity both inside and outside of the Earth science community. Part of its appeal is its easy-to-learn syntax and the thousands of available libraries which can be synthesized with core Python to do nearly any computing task imaginable. In particular, Python is useful for reading Earth-observing satellite datasets, which can be notoriously difficult to use due to the volume of information that results from the multitude of sensors, platforms, and spatio-temporal spacing. Python facilitates reading a variety of self-describing binary datasets that these observations are often encoded in. Using the same software, one can complete the entirety of a research project and even produce plots. Within a notebook environment, the scientist can document and distribute the code which can improve efficiency and transparency within the Earth sciences community. Even with the right tools data are seldom ready off-the-shelf for analysis and research and requires a number of pre-processing steps to make the data useable. What steps to take and why are often except perhaps for data developers themselves. Data users often misunderstand concepts such as data quality, how to perform an atmospheric correction, or the complex regridding schemes necessary to compare data with different resolutions. Even to a technical user, the nuances can be frustrating and difficult to overcome. The consequence of this is that data remains unused, or worse, potentially misused"--
