Hands-on Large Scale Optimization in Python

From Beginning to Giving Up

Kunlei Lian

2023-02-18

Table of contents

Preface

This is a Quarto book.

To learn more about Quarto books visit https://quarto.org/docs/books.

1 Environment Setup

In this chapter, we explain the steps needed to set up Python and Google OR-Tools. All the steps below are based on MacBook Air with M1 chip and macOS Ventura 13.1.

1.1 Install Homebrew

The first tool we need is Homebrew, 'the Missing Package Manager for macOS (or Linux)', and it can be accessed at https://brew.sh/. To install Homebrew, just copy the command below and run it in the Terminal.

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install
```

We can then use the `brew --version` command to check the installed version. On my system, it shows the info below.

```
  ~/ brew --version
Homebrew 3.6.20
Homebrew/homebrew-core (git revision 5f1582e4d55; last commit 2023-02-05)
Homebrew/homebrew-cask (git revision fa3b8a669d; last commit 2023-02-05)
```

1.2 Install Anaconda

Since there are several Python versions available for our use and we may end up having multiple Python versions installed on our machine, it is important to use a consistent environment to work on our project in. Anaconda is a package and environment manager for Python and it provides easy-to-use tools to facilitate our data science needs. To install Anaconda, run the below command in the Terminal.

```
  ~/ brew install anaconda
```

After the installation is done, we can use `conda --version` to verify whether it is available on our machine or not.

```
  ~/ conda --version
conda 23.1.0
```

1.3 Create a Conda Environment

Now we will create a Conda environment named 'ortools'. Execute the below command in the Terminal, which effectively creates the required environment with Python version 3.10.

```
  ~/ conda create -n ortools python=3.10
Retrieving notices: ...working... done
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /opt/homebrew/anaconda3/envs/test

  added / updated specs:
    - python=3.10


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    setuptools-67.4.0          |     pyhd8ed1ab_0         567 KB  conda-forge
    ------------------------------------------------------------
                                           Total:         567 KB

The following NEW packages will be INSTALLED:

  bzip2              conda-forge/osx-arm64::bzip2-1.0.8-h3422bc3_4
  ca-certificates    conda-forge/osx-arm64::ca-certificates-2022.12.7-h4653dfc_0
  libffi             conda-forge/osx-arm64::libffi-3.4.2-h3422bc3_5
  libsqlite          conda-forge/osx-arm64::libsqlite-3.40.0-h76d750c_0
  libzlib            conda-forge/osx-arm64::libzlib-1.2.13-h03a7124_4
  ncurses            conda-forge/osx-arm64::ncurses-6.3-h07bb92c_1
  openssl            conda-forge/osx-arm64::openssl-3.0.8-h03a7124_0
  pip                conda-forge/noarch::pip-23.0.1-pyhd8ed1ab_0
  python             conda-forge/osx-arm64::python-3.10.9-h3ba56d0_0_cpython
```

```
  readline           conda-forge/osx-arm64::readline-8.1.2-h46ed386_0
  setuptools         conda-forge/noarch::setuptools-67.4.0-pyhd8ed1ab_0
  tk                 conda-forge/osx-arm64::tk-8.6.12-he1e0b03_0
  tzdata             conda-forge/noarch::tzdata-2022g-h191b570_0
  wheel              conda-forge/noarch::wheel-0.38.4-pyhd8ed1ab_0
  xz                 conda-forge/osx-arm64::xz-5.2.6-h57fd34a_0


Proceed ([y]/n)?
```

Type 'y' to proceed and Conda will create the environment for us. We can use `cnoda env list` to show all the created environments on our machine:

```
 ~/ conda env list
# conda environments:
#
base                     /opt/homebrew/anaconda3
ortools                   /opt/homebrew/anaconda3/envs/ortools
```

Note that we need to manually activate an environemnt in order to use it: `conda activate ortools`. On my machine, the activated environment `ortools` will appear in the beginning of my prompt.

```
 ~/ conda activate ortools
(ortools)  ~/
```

1.4 Install Google OR-Tools

As of this writing, the latest version of Google OR-Tools is 9.5.2237, and we can install it in our newly created environment using the command `pip install ortools==9.5.2237`. We can use `conda list` to verify whether it is available in our environment.

```
(ortools)  ~/ conda list
# packages in environment at /opt/homebrew/anaconda3/envs/ortools:
#
# Name                    Version                   Build  Channel
absl-py                   1.4.0                     pypi_0    pypi
bzip2                     1.0.8                h3422bc3_4    conda-forge
ca-certificates           2022.12.7            h4653dfc_0    conda-forge
libffi                    3.4.2                h3422bc3_5    conda-forge
```

```
libsqlite                 3.40.0               h76d750c_0    conda-forge
libzlib                   1.2.13               h03a7124_4    conda-forge
ncurses                   6.3                  h07bb92c_1    conda-forge
numpy                     1.24.2                   pypi_0    pypi
openssl                   3.0.8                h03a7124_0    conda-forge
ortools                   9.5.2237                 pypi_0    pypi
pip                       23.0.1             pyhd8ed1ab_0    conda-forge
protobuf                  4.22.0                   pypi_0    pypi
python                    3.10.9         h3ba56d0_0_cpython    conda-forge
readline                  8.1.2                h46ed386_0    conda-forge
setuptools                67.4.0             pyhd8ed1ab_0    conda-forge
tk                        8.6.12               he1e0b03_0    conda-forge
tzdata                    2022g                h191b570_0    conda-forge
wheel                     0.38.4             pyhd8ed1ab_0    conda-forge
xz                        5.2.6                h57fd34a_0    conda-forge
```

Now we have Python and Google OR-Tools ready, we can start our next journey.

2 Introduction

This is a book created from markdown and executable code.

See Knuth (1984) for additional discussion of literate programming.

3 Environment Setup

In this chapter, we explain the steps needed to set up Python and Google OR-Tools. All the steps below are based on MacBook Air with M1 chip and macOS Ventura 13.1.

3.1 Install Homebrew

The first tool we need is Homebrew, 'the Missing Package Manager for macOS (or Linux)', and it can be accessed at https://brew.sh/. To install Homebrew, just copy the command below and run it in the Terminal.

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install
```

We can then use the `brew --version` command to check the installed version. On my system, it shows the info below.

```
  ~/ brew --version
Homebrew 3.6.20
Homebrew/homebrew-core (git revision 5f1582e4d55; last commit 2023-02-05)
Homebrew/homebrew-cask (git revision fa3b8a669d; last commit 2023-02-05)
```

3.2 Install Anaconda

Since there are several Python versions available for our use and we may end up having multiple Python versions installed on our machine, it is important to use a consistent environment to work on our project in. Anaconda is a package and environment manager for Python and it provides easy-to-use tools to facilitate our data science needs. To install Anaconda, run the below command in the Terminal.

```
  ~/ brew install anaconda
```

After the installation is done, we can use `conda --version` to verify whether it is available on our machine or not.

```
  ~/ conda --version
conda 23.1.0
```

3.3 Create a Conda Environment

Now we will create a Conda environment named 'ortools'. Execute the below command in the Terminal, which effectively creates the required environment with Python version 3.10.

```
  ~/ conda create -n ortools python=3.10
Retrieving notices: ...working... done
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /opt/homebrew/anaconda3/envs/test

  added / updated specs:
    - python=3.10


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    setuptools-67.4.0          |     pyhd8ed1ab_0         567 KB  conda-forge
    ------------------------------------------------------------
                                           Total:         567 KB

The following NEW packages will be INSTALLED:

  bzip2              conda-forge/osx-arm64::bzip2-1.0.8-h3422bc3_4
  ca-certificates    conda-forge/osx-arm64::ca-certificates-2022.12.7-h4653dfc_0
  libffi             conda-forge/osx-arm64::libffi-3.4.2-h3422bc3_5
  libsqlite          conda-forge/osx-arm64::libsqlite-3.40.0-h76d750c_0
  libzlib            conda-forge/osx-arm64::libzlib-1.2.13-h03a7124_4
  ncurses            conda-forge/osx-arm64::ncurses-6.3-h07bb92c_1
  openssl            conda-forge/osx-arm64::openssl-3.0.8-h03a7124_0
  pip                conda-forge/noarch::pip-23.0.1-pyhd8ed1ab_0
  python             conda-forge/osx-arm64::python-3.10.9-h3ba56d0_0_cpython
```

```
  readline           conda-forge/osx-arm64::readline-8.1.2-h46ed386_0
  setuptools         conda-forge/noarch::setuptools-67.4.0-pyhd8ed1ab_0
  tk                 conda-forge/osx-arm64::tk-8.6.12-he1e0b03_0
  tzdata             conda-forge/noarch::tzdata-2022g-h191b570_0
  wheel              conda-forge/noarch::wheel-0.38.4-pyhd8ed1ab_0
  xz                 conda-forge/osx-arm64::xz-5.2.6-h57fd34a_0


Proceed ([y]/n)?
```

Type 'y' to proceed and Conda will create the environment for us. We can use `cnoda env list` to show all the created environments on our machine:

```
 ~/ conda env list
# conda environments:
#
base                     /opt/homebrew/anaconda3
ortools                   /opt/homebrew/anaconda3/envs/ortools
```

Note that we need to manually activate an environemnt in order to use it: `conda activate ortools`. On my machine, the activated environment `ortools` will appear in the beginning of my prompt.

```
 ~/ conda activate ortools
(ortools)  ~/
```

3.4 Install Google OR-Tools

As of this writing, the latest version of Google OR-Tools is 9.5.2237, and we can install it in our newly created environment using the command `pip install ortools==9.5.2237`. We can use `conda list` to verify whether it is available in our environment.

```
(ortools)  ~/ conda list
# packages in environment at /opt/homebrew/anaconda3/envs/ortools:
#
# Name                    Version                   Build  Channel
absl-py                   1.4.0                     pypi_0    pypi
bzip2                     1.0.8                h3422bc3_4    conda-forge
ca-certificates           2022.12.7            h4653dfc_0    conda-forge
libffi                    3.4.2                h3422bc3_5    conda-forge
```

```
libsqlite                 3.40.0               h76d750c_0    conda-forge
libzlib                   1.2.13               h03a7124_4    conda-forge
ncurses                   6.3                  h07bb92c_1    conda-forge
numpy                     1.24.2                   pypi_0    pypi
openssl                   3.0.8                h03a7124_0    conda-forge
ortools                   9.5.2237                 pypi_0    pypi
pip                       23.0.1             pyhd8ed1ab_0    conda-forge
protobuf                  4.22.0                   pypi_0    pypi
python                    3.10.9        h3ba56d0_0_cpython    conda-forge
readline                  8.1.2                h46ed386_0    conda-forge
setuptools                67.4.0             pyhd8ed1ab_0    conda-forge
tk                        8.6.12               he1e0b03_0    conda-forge
tzdata                    2022g                h191b570_0    conda-forge
wheel                     0.38.4             pyhd8ed1ab_0    conda-forge
xz                        5.2.6                h57fd34a_0    conda-forge
```

Now we have Python and Google OR-Tools ready, we can start our next journey.

Part I

Benders Decomposition

4 Benders Decomposition

In this chapter, we will explain the theories behind Benders decomposition and illustrate its application using a simple linear programming problem. As its name suggests, Benders decomposition is a solution approach that breaks down an original optimization problem into smaller subproblems.

A natural question is, when should we use Benders decomposition? Like any other decomposition algorithms, recognizing when Benders decomposition algorithm is applicable is the most critical step in its successfully application. Often times, the theories are straightforward once we know that a problem structure is special enough that invites a decomposition strategy like Benders decomposition. However, this does require us to build up enough intuition so that Benders decomposition comes up into our mind when situation arises.

Generally speaking, Benders decomposition is a good solution candidate when the resulting problem is much easier to solve if some of the variables in the original problem are fixed. We will illustrate this point using an example in the following sections. In the optimization world, the first candidate that should come to mind when we say a problem is easy to solve is a linear programming formulation, which is indeed the case in Benders decomposition applications.

4.1 The Decomposition Logic

To explain the reasoning of Benders decomposition, let us look at the standard form of linear programming problems that involve two variables, $\mathbf{x}$ and $\mathbf{y}$. Note that both of them represent a column vector of variables of dimensions p and q, respectively.

$$
\begin{aligned}
(\mathbf{P}) \qquad \text{min.} \quad & \mathbf{c}^T\mathbf{x} + \mathbf{f}^T\mathbf{y} && (4.1)\\
\text{s.t.} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} = \mathbf{b} && (4.2)\\
& \mathbf{x} \geq 0, \mathbf{y} \geq 0 && (4.3)
\end{aligned}
$$

Knuth, Donald E. 1984. “Literate Programming.” *Comput. J.* 27 (2): 97–111. https://doi.org/10.1093/comjnl/27.2.97.