

# Clean Air in the Time of COVID

## Use Cases for Google Earth Engine & Sentiment Analysis

Ben Best, PhD

2020-05-25



# Contents



# Overview

These materials are for a 45 minute teaching demonstration on 2020-05-26 oriented towards students in the new Masters in Environmental Data Science at UCSB.

## Motivation

This is an exciting time for the emerging field of Environmental Data Science. Environmental problems are increasingly complex and require advanced technical skills to translate the ever expanding flow of data into actionable insights.

## Technologies

The two specific technologies featured in this teaching demonstration highlight some of the most promising aspects of truly :

- **Google Earth Engine** leverages the massive storage and computational capabilities of the Google Cloud to analyze petabytes of the publicly available satellite data. For instance, global climatologies averaging across 40 years of temperature can be calculated and mapped in seconds. This is a truly *big data* platform!
- **TensorFlow** is the machine learning software made open-source by Google. It is the most commonly used software for audio, text and image analysis. More specifically tensorflow allows construction of *convolutional neural networks*, which represent a layering of nodes to enable complex pattern matching. These *deep learning* models have been popularized by their ability to beat the best human at the most complex game of Go, self-drive vehicles and respond to your beck and call through Alexa's voice commands.

## Motivating Use Case

For the first time in decades, Mount Everest was visible from Kathmandu due to improved air quality.

— Tom Patterson (@MtnMapper) May 21, 2020

The positive effect of the Lockdown in India's environment can be seen in New Delhi. This picture depicts the air quality of New Delhi before & after Lockdown. Source - <https://t.co/VoG0UbpFzE>

— GurumaujSatsangi (@GurumaujS) April 23, 2020

## Prerequisites

See the Setup for required software. For this demonstration to make the most sense, it's preferable that you're familiar with:

- **R**
- **GIS**

# Setup

## 0.1 Install software

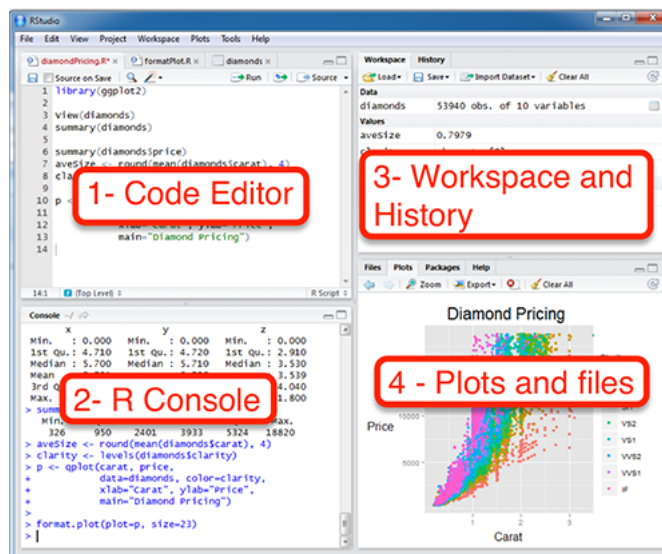
This workshop will require the following software installed on your machine:

- R
- RStudio

Please download the appropriate stable release for your operating system.

## 0.2 Launch RStudio

RStudio is an integrated development environment (IDE) for editing text in the code editor, running commands in the R console, viewing defined objects in the workspace and past commands in the history, and viewing plots, files and help. Here's a layout of the panes in RStudio, each of which can have several tabs for alternate functionality:



Check out the Help > Cheatsheets > RStudio IDE Cheat Sheet.

### 0.3 Fork and Clone the Github Repository

Please visit <https://github.com/bbest/meds-demo>, signed into Github. Then into your own writable user space.

Here is the Github repository of the source files for this demonstration:

<https://github.com/bbest/meds-demo>

You are encouraged to fork the repo in Github and clone it in RStudio.

Then you can follow along in RStudio by evaluating the chunks of R code, while referencing the website:

<http://benbestphd.com/meds-demo>

### 0.4 Install R Packages

Here's a bit of code to install packages that we'll use throughout the workshop. Please copy and paste this code into your console.

```
# use librarian to load libraries, installing if needed
if (!require("librarian")) install.packages("librarian")
library("librarian")
```



```

# load packages
pkgs <- c(
  # general
  "tidyverse", "jsonlite", "glue", "here", "units",

  # satellite
  "sf",

  # sentiment
  "rtweet", "tidytext")
shelf(pkgs)

# report on versions of software & packages
sessionInfo()

## R version 3.5.2 (2018-12-20)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14.6
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats      graphics  grDevices  utils      datasets  methods   base
##
## other attached packages:
## [1] tidytext_0.2.2  rtweet_0.7.0    sf_0.8-0        units_0.6-6
## [5] here_0.1        glue_1.4.1      jsonlite_1.6.1  forcats_0.4.0
## [9] stringr_1.4.0   dplyr_0.8.5     purrr_0.3.4     readr_1.3.1
## [13] tidyr_1.1.0     tibble_3.0.1    ggplot2_3.3.0   tidyverse_1.3.0
## [17] librarian_1.7.0
##
## loaded via a namespace (and not attached):
## [1] Rcpp_1.0.4.6    lubridate_1.7.8  lattice_0.20-38  class_7.3-15
## [5] assertthat_0.2.1 rprojroot_1.3-2  digest_0.6.25    R6_2.4.1
## [9] cellranger_1.1.0 backports_1.1.7  reprex_0.3.0     evaluate_0.14
## [13] e1071_1.7-3     blogdown_0.17    httr_1.4.1       pillar_1.4.4
## [17] rlang_0.4.6     readxl_1.3.1     rstudioapi_0.11  Matrix_1.2-18
## [21] rmarkdown_2.1   munsell_0.5.0    broom_0.5.5      janeaustenr_0.1.5
## [25] compiler_3.5.2  modelr_0.1.5     xfun_0.14        pkgconfig_2.0.3
## [29] vembedr_0.1.3   htmltools_0.4.0  tidyselect_1.1.0 bookdown_0.16

```

## [33]	fansi_0.4.1	crayon_1.3.4	dbplyr_1.4.2	withr_2.2.0
## [37]	SnowballC_0.6.0	grid_3.5.2	nlme_3.1-142	gtable_0.3.0
## [41]	lifecycle_0.2.0	DBI_1.1.0	magrittr_1.5	tokenizers_0.2.1
## [45]	scales_1.1.1	KernSmooth_2.23-16	cli_2.0.2	stringi_1.4.6
## [49]	fs_1.3.1	xml2_1.2.2	ellipsis_0.3.1	generics_0.0.2
## [53]	vctrs_0.3.0	tools_3.5.2	hms_0.5.3	yaml_2.2.1
## [57]	colorspace_1.4-1	classInt_0.4-3	rvest_0.3.5	knitr_1.28
## [61]	haven_2.2.0			

## 0.5 Create Rmarkdown file

Rmarkdown is a dynamic document format that allows you to knit chunks of R code with formatted text (aka markdown). We recommend that you use this format for keeping a reproducible research document as you work through the lessons To get started, File > New File > Rmarkdown... and go with default HTML document and give any title you like (default “Untitled” or “test” or “First Rmd” is fine).

Check out the Help > Markdown Quick Reference and Help > Cheatsheets > R Markdown Cheat Sheet.

Here’s a 1 minute video on the awesomeness of Rmarkdown:

# Chapter 1

## Satellite

### Objectives

#### Question

- How have emissions related to air quality changed since COVID-19 lockdowns were put in place?

#### Study area: Delhi, India

- We'll use **Delhi, India** as our initial city study area. Prime Minister Modi issued a nationwide lockdown on 24 March, 2020.

#### News:

- Air pollution falls by unprecedented levels in major global cities during coronavirus lockdowns - CNN
- India: air pollution in the north has hit a 20-year low, NASA report says - CNN
- Fact Check: Is The COVID-19 Lockdown Decreasing Delhi Air Pollution? - Smart Air Filters

#### Learning outcomes

1. Browse Google Earth Engine's data catalogue
2. Map satellite imagery dataset layer
3. Filter by date
4. Average values over time
5. Upload a polygon into assets

6. Use polygon to extract values for study area
7. Average values in space within the polygon
8. Generate a time series chart for satellite data
9. Download data as a text file (csv)

## Prerequisites

A **Google Earth Engine account** that is associated with a Google account, such as from Gmail, is required to log into <https://code.earthengine.google.com>.

If you need a GEE account, please visit <https://signup.earthengine.google.com>.

You may need to log out and back into your web browser as the preferred Google account to request permissions. This approval process can take days to weeks unfortunately.

### 1.1 Get city boundary

The first step is to define our study area. I made a little R helper function `city2zip()` to:

1. Fetch the administrative boundary from the Nominatim OpenStreetMap API given a city name.
2. Extract the polygon information, convert to shapefile and zip for upload into GEE.

```
source("functions.R")  
city2zip("Delhi, India")
```

The function returns the paths to files generated. You will use this zip file to upload into GEE as an asset.

### 1.2 Browse datasets, tag no2

Visit <https://earthengine.google.com> > Datasets (upper right). Be sure to explore the many datasets available here.

Since we know we want Nitrogen Dioxide (NO<sub>2</sub>), click on Browse by tags and Filter by “no2”. You should see two datasets:

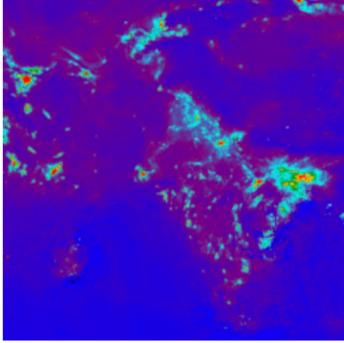
Earth Engine Data Catalog

Home View all datasets Browse by tags Landsat MODIS Sentinel API Docs

## Datasets tagged no2 in Earth Engine

Filter list of datasets

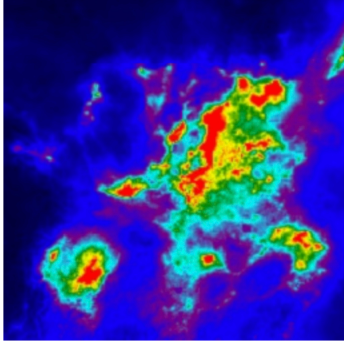
Sentinel-5P NRTI NO2: Near Real-Time Nitrogen Dioxide



Sentinel-5 Precursor Sentinel-5 Precursor is a satellite launched on 13 October 2017 by the European Space Agency to monitor air pollution. The onboard sensor is frequently referred to as Tropomi (TROPOspheric Monitoring Instrument). All of the S5P datasets, except CH4, have two versions: Near Real-Time ...

tropomi no2 nitrogen-dioxide pollution air-quality eu

Sentinel-5P OFFL NO2: Offline Nitrogen Dioxide



Sentinel-5 Precursor Sentinel-5 Precursor is a satellite launched on 13 October 2017 by the European Space Agency to monitor air pollution. The onboard sensor is frequently referred to as Tropomi (TROPOspheric Monitoring Instrument). All of the S5P datasets, except CH4, have two versions: Near Real-Time ...

tropomi no2 nitrogen-dioxide pollution air-quality eu

Figure 1.1: Screenshot of GEE data catalog filtered by tag “no2” showing two datasets.