Guide: Using PGC GitHub: NDVI
URL: https://www.pgc.umn.edu/guides/pgc-coding-and-utilities/using-pgc-github-ndvi/
Last Modified: December 17, 2019
Export Date: November 7, 2026

Introduction

In this guide, you will learn what software you need to run the pgc Normalized Difference Vegetation Index (NDVI) script, where to access the required software, and how to use the script with a sample workflow.

The PGC hosts several open source codes on GitHub, a company that hosts software development. Projects, source codes, changes and versions can be accessed in PGC's Repositories online.

Quick Links

- Polar Geospatial Center GitHub Homepage
- Polar Geospatial Center GitHub Imagery Utilities
- Mac and Linux: Installation of PGC's GDAL Stack
- Windows: OSGeo4W Installation

About

The Normalized Difference Vegetation Index (NDVI) is a numerical indicator used to determine the presence and state of vegetation. This index uses the normalized difference between the near-infrared band and the red band of a multispectral image. The reason these two bands are used is due to how these wavelengths interact with vegetation. Chlorophyll in plant cells strongly absorbs red wavelengths, while the leaf cell structure strongly reflects near-infrared wavelengths. The formula for NDVI is as follows:

$$NDVI = (NIR - RED) / (NIR + RED)$$

Generally, the output of this index is saved as 32-bit float, which constrains the range of possible values between -1 and 1. Values near one indicate green vegetation, values near zero indicate dead vegetation or soil, and negative values usually correspond with water.

The Polar Geospatial Center has developed a Python script that batch calculate NDVI from multispectral satellite imagery. This tool is designed to run on data that has already been orthorectified using the pgc_ortho.py utility.

Requirements

The pgc_ndvi.py tool can run within a personal computing environment (desktop computer) or in a cluster computing environment. The code is built to run primarily on a Linux HPC cluster running Maui/Torque for queue management. This tool will also work on a windows platform.

> Please note that the code is tightly coupled to the systems on which it was developed. You should have no expectation of it running on another system without some patching.

Software:

This tool is built on the GDAL/OGR image processing API using Python. GDAL 2.1 is required for this tool to function.

If you are using a Linux system you will need to download the PGC optimized GDAL toolchain. The list of software installed with the optimized GDAL toolchain can be found here. A script is provided to install all required packages. If you have not ran a shell script in a Linux terminal follow this guide here. After installation of the PGC optimized GDAL toolchain, the NDVI tool can be ran through the Linux terminal. There are plenty of free, online tutorials for Linux terminal if you are new to command line interfaces.

If you are using a Windows system, it is recommended that you use OSGeo4W. This will provide a Windows environment to use the tool. You can get the installers here. The express installation will provide the most high profile OSGeo4W packages. However, it will not allow for control over install location, proxies, and cache directory selection. The advanced install will allow for more control. The PGC NDVI tool will run with either install type. After installation of OSGeo4W, the NDVI tool can be ran through the OSGeo4W Shell. As with Linux, there are numerous online resources for using a Windows command line interface.

Script Details

The pgc_ndvi utility runs batch image orthorectification, conversion, and pansharpening or submits them to a PBS or SLURM HPC cluster for processing. First, the utility applies the orthorectification process to both the panchromatic (if provided) and multispectral image in a pair and then pansharpens them using the GDAL tool gdal_pansharpen. Submission scripts to PBS and SLURM can be found at the PGC GitHub page. Including the command **“- -pbs”** will submit the task to PBS, and including the command **“- -slurm”** will submit the task to SLURM. When submitting a job to a cluster where there is storage local to the processing node it is recommended to include the **“- -wd”** command. This will allow you to set a local working directory which will allow for increased processing time.

A description of the commands can be found here. Information regarding common commands are detailed in the sample workflow below.

Sample Workflow

Before you begin you will need to gather all your orthorectified imagery and place them in a single folder. If you used the pgc_ortho utility this should automatically be done. Note for these commands that Linux users will have forward slashes (/) in the pathname, and pathnames will begin with “/mnt/” instead of using the drive name as in Windows (D:\).

1. Once you have gathered all orthorectified image files, you will need to open either the OSGeo4W shell, if you are using Windows, or your Linux terminal if you are using Linux. Once you have opened the window, type “python” followed by the pathname for the pgc_NDVI.py script.
   ```
   C:\>python user\pathname\pgc_ndvi.py
   ```

   This will tell the computer to use Python to run the script, which is found in the location you specified. Dragging and dropping the pgc_NDVI.py file into the terminal will automatically populate the file pathname.

2. The next step is to identify the output bit depth. The available bit depths are 16 and 32. A larger bit

depth means that a larger array of possible color values for each pixel can be represented. The most common output type is 32-bit float, which will constrain the NDVI value between -1 and 1. If you decide to use a 16-bit output, the output values will be scaled from -1000 to 1000. Available options for this command include:

Int16 – 16 bit output
Float32 – 32 bit floating point output

To designate an output bit depth we can use the **“-t”** or **“- -outtype=”** followed by one of the previously mentioned command options. For example:

```
C:\>python user\pathname\pgc_ndvi.py -t Int16
```

Or

```
C:\>python user\pathname\pgc_ndvi.py -t Float32
```

3. The last step in setting up the pgc_ndvi tool is specifying the location of the input imagery and the desired output location. Type the pathname to the folder containing the orthorectified imagery followed by the pathname to the desired output folder. You can drag and drop the folders into the terminal to have their pathname’s appear. Lastly, make sure that the proper syntax is used (like spaces between commands). The example below lists commands with proper syntax:

   ```
   C:\>python user\pathname\pgc_ndvi.py -t Float32 D:\input\imagery\orthorectified D:\output\folder\ndvi
   ```

   To run the tool, press the Enter key. It is important to determine the exact commands you will need before running the tool.

Additional Resources

NDVI Publication

https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19780024582.pdf

NDVI Information

https://earthobservatory.nasa.gov/features/MeasuringVegetation/measuring_vegetation_2.php

Bit Depth Explained:

https://apollomapping.com/2012/September/article8.html