# Contents

1 Introduction 3
  1.1 Overview ................................................. 3
  1.2 What about that name? .......................... 3
  1.3 The future .............................................. 4

2 Example datasets 5

3 Getting Started 7
  3.1 TerriaJS .................................................. 7
  3.2 National Map web site .............................. 14
  3.3 Jupyter notebooks ................................. 21

4 Using GSKY through the web 23
  4.1 How to use WPS on the GEOGLAM RAPP Map .......... 23
  4.2 Constructing WPS Requests using the GEOGLAM polygon drill 23

5 GSKY interacts with ArcGIS suite 43
  5.1 Introduction .............................................. 43
  5.2 0. Prerequisites ........................................ 43
  5.3 1. Sign in using google account ...................... 43
  5.4 2. Choose a Basemap ................................ 47
  5.5 3. Load GSKY layer onto a map ...................... 51
  5.6 4. Add Oil and Gas pipeline data ................. 56
  5.7 5. View attribute table ............................. 63
  5.8 6. Style the layer with attribute .................. 66
  5.9 7. Enable and customise the Pop-up .............. 72
  5.10 8. Save the map and create a web app to share .... 78

6 GSKY interacts with QGIS 87
  6.1 Introduction .............................................. 87
  6.2 0. Prerequisite ......................................... 87
  6.3 1. Launch QGIS ........................................ 88
  6.4 2. Add GSKY WMS layer ............................ 88
  6.5 3. Add GSKY WCS layer ............................ 95

7 Installing Jupyter and Python 107
  7.1 Installing Miniconda on macOS ..................... 107
CHAPTER 1

Introduction

1.1 Overview

The NCI-developed GSKY service provides a new approach to online analysis and visualisation of environmental data. GSKY provides an ability for users to interact with datasets and the information they contain using standard community protocols.

GSKY accesses and analyses the big geospatial data on NCI’s cloud and high-performance computing systems, and then delivers it to a user device or website. For example, hundreds of time series and geospatially overlapping data can be seamlessly merged together, allowing researchers to focus on the information rather than dealing with data files. Furthermore, using GSKY’s processing capability, that data can be analysed on the fly using user-provided algorithms to extract new information over both space and time.

Behind the scenes, GSKY works out how to manipulate the datasets so that they seamlessly work together. For example, in large-scale environmental analyses, the images from different satellites can be in different shapes and sizes, environmental survey data can come in many different formats, and even urban boundary maps need to be considered. As a user of GSKY, working with data is as easy as choosing from a list of available datasets, specifying a region and time frame, and asking GSKY to analyse the information as harmonised data. GSKY then returns the results of the data required, which can be accessed over the network to the client application or for visualisation in an online map.

1.2 What about that name?

The Apollo moon landings were supported by a computer interface known as DSKY. Astronauts could input data and commands into the keypad and see the results returned on an electronic display. In much the same way, GSKY is an interface that allows human manipulation of deeply buried geospatial data. Using GSKY, a user can make complex requests and see the results in their web browser in near real-time.

GSKY cannot navigate its users to the moon – it can, however, help us understand it.
1.3 The future

Staff at NCI are already looking to the next evolution of GSKY. Increasing the number of datasets and services that GSKY employs will broaden its usage for researchers needing access to new and different types of geospatial information, and may also extend GSKY’s usefulness across multiple scientific disciplines.

With the buzz surrounding machine learning, deep learning and artificial intelligence, GSKY offers a tantalising glimpse into the future of this flourishing computer science field. Just like humans, the algorithms that enable machine learning require access to large pre-prepared data collections – something that GSKY makes short work of.
CHAPTER 2

Example datasets

GSKY currently serves data collections such as:

**Digital Earth Australia (DEA) Geoscience Earth Observations**, which include the following products of the Landsat 5, 7 and 8 satellite missions:

- Surface reflectance (NBAR/NBART true and false colour)
- Terrain corrected surface reflectance geometric median (geomedian)
- Intertidal Extents Models (ITEM)
- High and Low Tide Composites (HLTC)
- Water Observations from Space (WOfS)
- Sentinel 2 Analysis Ready Data
- Multi-sensor (Landsat and Sentinel 2) surface reflectance (Beta)

**GEOGLAM**, the GEO Global Agricultural Monitoring initiative, which include the following products:

- MODIS Total Vegetation Cover v3.1 (8-day and Monthly)
- MODIS Total Vegetation Cover Anomaly v3.1 (Monthly)
- MODIS Total Vegetation Cover Decile v3.1 (Monthly)
- MODIS Vegetation Fractional Cover 8-day v3.1 (8-day and Monthly)
- CHIRPS Precipitation v2.0 (Monthly)

You can browse and search NCI’s full collection by going to the Terria Map or National Map websites.

To view the DEA or GEOGLAM collections, click on Add Data -> My Data -> Add Web Data and enter the following URLs respectively:


http://gsky.nci.org.au/ows/geoglam
For more in-depth tutorials, please visit the *getting started section.*

We are continually adding new datasets to GSKY. If you’re interested in publishing your own datasets to GSKY, please submit an enquiry to our data collections team via help@nci.org.au
GSKY can be used with many different OGC compatible clients.

**Want to try it out?**

Below is a short video demonstrating how to add GSKY layers to **TerriaJS**.

### 3.1 TerriaJS

*TerriaJS* is an open-source framework for web-based geospatial catalog explorers. The TerriaJS portal is target software that is used by some major projects NCI collaborates with:

- GEOGLAM RAPP
- Digital Earth Australia
- National Map

As an example, we can add MODIS Fractional Cover time series as a single layer to the GEOGLAM RAPP map. The GEOGLAM data portal contacts the GSKY server to provide a monthly MODIS fractional cover layer with three bands. This MODIS FC time series dates back to 2001:
GSKY also provides access to data subsets (e.g., group of variables of interest) as data layers:
GSKY is able to serve multiple layers simultaneously overlaying on top of each other for comparative analysis:
GSKY can provide the same layer for different points of time allowing cross-sectional analysis:
Above we see a comparison of fractional coverage between August 2018 (left) and August 2017 (right). Note the increase of fractional coverage around Perth and the decrease around NSW.

GSKY’s WCS service allows users to download the raw data corresponding to a user-supplied bounding box. Note the “export” button in the following screenshot:
Simply draw a bounding box and the raw data corresponding to this selection will be downloadable:
The stand-alone visualisation of the downloaded data file, as per the selection made via a click-and-ship:
3.2 National Map web site

Go to the National Map website

Click on Add Data -> My Data -> Web Data and type:

3.2. National Map web site


2. Click on "Add data" to add data to your workbench.

3. Load your own data onto the map by clicking on "My Data".

TIP: All your active data sets will be listed here.
GSKY provides a catalogue of all the datasets and links to NCI’s ISO 19115 data catalogue entries:
You can add multiple datasets to the National Map and plot them on top of each other. Here is an example Landsat 8 terrain corrected surface reflectance true colour data overlain by Water Observations from Space (WOfS) data:

### DEA Intertidal Extents Model Relative Layer

**Data Description**

The Intertidal Extents Model (ITEM) product is a national extent of the exposed intertidal zone, the land between tide, at intervals of the observed tidal range. ITEM provides the intertidal zone of Australia's coastline (excluding the information was derived using observations in the Land has implemented an improved tide modelling framework). The expanded Landsat archive v2 (DEA) has also enabled the model extent to be increased number of offshore reefs, including the Great Barrier the Torres Strait Islands. ITEM can be a valuable comple LiDAR survey data and coarser offshore bathymetry data representation of the land and ocean interface. More details CCBY4.0 is available at https://dx.doi.org/10.4225/25/5a access to the Intertidal Extents Model v20 Relative Layer displays the modelled extents of the exposed intertidal observed tidal range (OTR). For example, the region define that only exposes at the lowest 10% of tides in relation

**Service Description**

This service relies on GSKY - A Scalable, Distributed Geo https://geonetwork.nci.org.au/geonetwork/srv/en/eng/cat 8d6f4076-a734-93ac7bc9201

**GetCapabilities URL**


**Service Contact**

National Computational Infrastructure
help@nci.org.au

**Web Map Service (WMS) URL**

This is a WMS service, which generates map images on software with this URL:

http://gsky.nci.org.au/ows

Layer name: itemrelative

* Data Source Details
* Data Service Details
The split bar feature allows one to compare different datasets as well as different dates from the same dataset. Below is an example of using the scroll bar, with Landsat 8 geomedian true colour plotted on the left and Landsat 8 geomedian false colour plotted on the right:
3.2. National Map web site
3.3 Jupyter notebooks

1. Manually using THREDDS Data Cube

To extract data through the Web Coverage Service (WCS), the request takes the following form:

```
HTTP://dapds00.nci.nih.gov/thredds/wcs?REQUEST=GetCoverage
&SERVICE=WCS&VERSION=1.0&PARAMS=1&FORMAT=JSON
```

where red indicates required fields and others need to be defined.

GetCoverage parameters:

- `PARAMS`: This parameter is used to specify the variable(s) to be retrieved. In this case, it is set to `1`, indicating that only one variable is requested.
- `FORMAT`: This parameter specifies the format in which the data will be returned. In this case, it is set to `JSON`.

---

Example code for requesting GeoTiff

```python
In [7]:
# time slice
i = 22

T_s = ofam.variables['temp'][i, 0, 200:800, 1, lon_s = lon[1400:2100]
lat_s = lat[200:800]

plt.figure(figsize=(12,12))
plt.pcolormesh(lon_s, lat_s, T_s)
plt.clim(vmin=18, vmax=30)
```

Do the same for a smaller subset
Programmatic access through Jupyter notebooks is easy via GSKY’s OGC services. Please see the Jupyter notebooks section for some examples on how to use GSKY in Python. To clone/download these tutorials, please visit our gsky-demos github page.
4.1 How to use WPS on the GEOGLAM RAPP Map

RAPP Map is the spatial data platform for the Rangeland and Pasture Productivity activity which is part of the Group on Earth Observations Global Agricultural Monitoring (GEOGLAM) initiative. This online tool gives access to information about the state and condition of global rangelands. It provides time-series data on the vegetation and environmental conditions, allowing national and regional tracking of the resources which sustains livestock production. It has been developed, and is currently hosted, by Data61 with the assistance of IT resources and services from the National Computational Infrastructure (NCI), and the AusCover facility. RAPP Map is supported by CSIRO and through funding from the Australian Government’s National Landcare Programme.

4.2 Constructing WPS Requests using the GEOGLAM polygon drill

Below is a short video tutorial on how to run a WPS request using the GEOGLAM RAPP Map.

The current WPS request for GEOGLAM takes a polygon as input and calculates the band averages over time for the polygon of interest. Let’s first access the GEOGLAM WPS via the TerriaJS interface:

http://map.geo-rapp.org/
Chapter 4. Using GSKY through the web
To add data to the map, click on the ‘Add data’ tab:
Here we can see a number of both global and Australian datasets that can be added to the GEOGLAM RAPP Map. Have a play around and add some of these datasets to the map. As an example we have added the Global Rainfall (5km) dataset:
Chapter 4. Using GSKY through the web

Rainfall (Global, 5km) December 2005

GEOGLAM RAPP
Rangeland and Pasture Productivity

Data Sets: [Area Labels, Area Borders, Rainfall (Global, 5km)]

Opacity: 100%

Time: 21/12/2005, 11:00:00
You can adjust the opacity (green box) and time (blue box). To remove the dataset from the map, click the *Remove* button (red box).

Try out the split option where you can select two different dates (one left, one right) and use the *split* bar to see the changes in data between the specified dates:
Now let’s try using the **Analysis Tools**. Once again, click on the **Add data** tab and then **Analysis Tools**.
Chapter 4. Using GSKY through the web
These tools can be used to query the Vegetation Fractional Cover layer data. The output is a graph which includes NPV, PV, NPV+PV Total, BS, and Precipitation. For this example, we will draw a polygon over an area we are interested in:
Let's draw a polygon on the map and then click the **Done** button:
Chapter 4. Using GSKY through the web
This will take you back to the Analysis Tools. Now click the Run Analysis button:
Now you can visualise the time series charts of Precipitation, PV, NPV, NPV+PV Total and BS.
Chapter 4. Using GSKY through the web

Chart Vegetation Cover generated.
Chart Accumulated Precipitation generated.
To add or remove time series, click on the appropriate legend items (green box). To download the time series as a csv, click the **Download** button (red box):
This tutorial demonstrates how ArcGIS pro, ArcGIS Earth and ArcMap interact with the GSKY Web Mapping Services (WMS). In this tutorial, you will learn to add layers by calling the GSKY WMS server.

5.1 Introduction

ArcGIS

In this tutorial, we are going to demonstrate how to:

- Create a map showing evacuation routes in Houston
- Use smart mapping and pop-ups to determine areas with low vehicle ownership.
- Configure a web app to share your findings with others.

5.2 Prerequisites

In this lesson, you’ll create a map that shows hurricane evacuation routes in Houston, Texas. First, you’ll create a new map and locate Houston, Texas. Then, you’ll add a map layer that shows evacuation routes. Lastly, you’ll change the way your map and layers look to better display the data.

If you don’t have an ArcGIS account, you can sign up for a free public account or an ArcGIS free trial.

5.3 Sign in using google account

Type this link in your web browser: https://www.arcgis.com/home/createaccount.html
Click “Using Google” to login.
5.3. 1. Sign in using google account

You might need to click twice to login.
5.4 2. Choose a Basemap

Click “Map” on the top bar. You will see a map with the default location.
Chapter 5. GSKY interacts with ArcGIS suite

ArcGIS  My Map

Details  Add  | Basemap

About  Content  Legend

Make your own map

It’s easy to make your own map. Just follow these steps:

1. Choose an area.
   Pan and zoom the map to an area or search by its name or address.

2. Decide what to show.
   Choose a Basemap then Add layers on top of it.

3. Add more to your map.
   Add map notes to draw features on the map.
   Display descriptive text, images, and charts for map features in a pop-up.

4. Save and share your map.
   Give your map a name and description then share it with other people.
Choose “Terrain with Labels” as the Basemap from the list.

Type the location in the search window on the right-top corner and click enter. Dismiss the “Search result” box. We will just use the default view or search “Australia” if the default view is not Australia.
ArcGIS  My Map

Make your own map

It’s easy to make your own map. Just follow these steps:

1. **Choose an area.**
   Pan and zoom the map to an area or search by its name or address.

2. **Decide what to show.**
   Choose a Basemap then Add layers on top of it.

3. **Add more to your map.**
   Add map notes to draw features on the map.
   Display descriptive text, images, and charts for map features in a pop-up.

4. **Save and share your map.**
   Give your map a name and description then share it with other people.
5.5 3. Load GSKY layer onto a map

Click “Add | Add Layer from Web”.

You will see a list of options from the drop down box below. Choose “A WMS OGC Web Service”.

Add Layer from Web

What type of data are you referencing?

- A WMS OGC Web Service
- An ArcGIS Server Web Service

Select A WMS OGC Web Service

Add Custom Parameters

ADD LAYER CANCEL
Type “https://gsky.nci.org.au/ows/geoglam” in the URL box. Then click “ADD LAYER”. You can tick the box beneath of URL box “Use as Basemap” if you like.

Add Layer from Web

What type of data are you referencing?

A WMS OGC Web Service

URL

https://gsky.nci.org.au/ows/geoglam

Use as Basemap

Add Custom Parameters

GET LAYERS

Having trouble displaying your OGC Web service? Help us improve this site by sending us the URL via the Contact Esri link.

All data layers published through GSKY production server should now be available for you to choose. Click the grey arrow to expend the dataset list. Note if you untick the top layer “GSKY Web Map Service”, all the GSKY layers will be invisible.
5.5. 3. Load GSKY layer onto a map
You can tick/untick the layer as you like.

You can change the layer’s appearance for each layer. Put mouse on top of the “… ” under each layer, a list of options will pop up as below. For example, you can increase the Transparency by moving the percentage bar. You can use this layer as the Basemap by clicking “Move to Basemap”. You can also remove the layer by choosing “Remove” option.
5.5. 3. Load GSKY layer onto a map
5.6 4. Add Oil and Gas pipeline data

Click “Add | Browse Living Atlas Layers”.

Search “Australia” to see what database is available in this Living Atlas. Two layers are found: Oil and Gas Infrastructure and DEM SREM 1 Second Hydro Enforced.
Choose “Oil_Gas_Infrastructure” and click “Add layer”.

5.6. 4. Add Oil and Gas pipeline data
Oil_Gas_Infrastructure
Map Image Layer by geoscienceaustralia
Updated: 8 February 2016

The spatial locations of onshore oil and gas pipelines along with offshore oil and gas platforms within Australia’s territorial waters.

Description
This web service provides access to the National Oil and Gas Infrastructure datasets. These datasets present the spatial locations of onshore oil and gas pipelines for the transmission of oil and gas within mainland Australia. They also present the location of oil and gas platforms within Australia’s territorial waters.

Terms of Use
No special restrictions or limitations on using
Close the layer information (step 1 in red circle) and go back to the map view (step 2 in red circle)

Two layers are shown on the map.
Chapter 5. GSKY interacts with ArcGIS suite

ArcGIS ▼ My Map

Contents

- Oil Gas Infrastructure
- GSKY Web Map Service
  - MODIS Total Vegetation Cover Monthly v3.1
  - MODIS Total Vegetation Cover Decile Monthly v3.1
  - MODIS Total Vegetation Cover Anomaly Monthly v3.1
  - MODIS Total Vegetation Cover 8-day v3.1
  - MODIS Vegetation Fractional Cover Monthly v3.1
  - MODIS Vegetation Fractional Cover 8-day v3.1
  - Monthly Precipitation CHIRPS v2.0
- Topographic
Drag and move to re-arrange the order of different layers.

Click “Details | Contents | Layer name | Transparency” to see through this layer.
5.7 5. View attribute table

Click “Details | Content | Layer name (National Onshore Oil Pipelines) | Show Table” (click the table icon).

Move the bar up to expand the table view.
ArcGIS > My Map

Contents

- GSKY Web Map Service
- Oil Gas Infrastructure
  - National Onshore Oil Pipelines
  - National Onshore Gas Pipelines
  - National Oil Gas Platforms
- Topographic

National Onshore Oil Pipelines (Features: 73, Selected: 0)

<table>
<thead>
<tr>
<th>FEATURETYPE</th>
<th>DESCRIPTION</th>
<th>CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
</tr>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
</tr>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
</tr>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
</tr>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
</tr>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
</tr>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
</tr>
</tbody>
</table>

Chapter 5. GSKY interacts with ArcGIS suite

Pipeline A linked series of pipe, with pumps and valves, used for the conveyance of oil Underground
Select the records that you find interesting. Click “Show Selected Records”.

### National Onshore Oil Pipelines (Features: 73, Selected: 4)

<table>
<thead>
<tr>
<th>FEATURETYPE</th>
<th>DESCRIPTION</th>
<th>CLASS</th>
<th>fid</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
<td>1</td>
<td>Tyabb to Corio</td>
</tr>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
<td>2</td>
<td>Dutson to Hastings</td>
</tr>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
<td>3</td>
<td>Moonie to Brisbane</td>
</tr>
<tr>
<td>Pipeline</td>
<td>A linked series of pipe, with pumps and valves, used for the conveyance of oil</td>
<td>Underground</td>
<td>4</td>
<td>Moonie to Brisbane</td>
</tr>
</tbody>
</table>

The four selected records are highlighted on the map.
5.8 6. Style the layer with attribute

Click “Details | Contents | Layer name | Change style”.
5.8. 6. Style the layer with attribute

Choose an attribute to show - Operational status. Click “Option”.
gsky Documentation

68 Chapter 5. GSKY interacts with ArcGIS suite

Change Style

Oil Gas Infrastructure - National Onshore Oil Pipelines

1. Choose an attribute to show

   OPERATIONALSTATUS

2. Select a drawing style

   Types (Unique symbols)

   OPTIONS

   Location (Single symbol)

   SELECT

   DONE CANCEL

Click the colour line to change the styles.
## Change Style

**Oil Gas Infrastructure - National Onshore Oil Pipelines**

### OPERATIONAL STATUS

<table>
<thead>
<tr>
<th>Label</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operationa</td>
<td>64</td>
</tr>
<tr>
<td>Non</td>
<td>9</td>
</tr>
<tr>
<td>Other</td>
<td>0</td>
</tr>
</tbody>
</table>

Select thick black solid line for operational lines.
Select thick grey dash line for non-operational lines.
5.8. 6. Style the layer with attribute
5.9 7. Enable and customise the Pop-up

Click “Details | Contents | Layer name (National Onshore Oil Pipelines) | More Options”.

Click “Enable Pop-up”.

5.9. 7. Enable and customise the Pop-up
Chapter 5. GSKY interacts with ArcGIS suite

Click on a tract to see the pop-up.
Click “Details | Contents | Layer name (National Onshore Oil Pipelines) | More Options | Configure Pop-up”.
Choose “A description from one field” to display as pop-up content.
Configure Pop-up

Oil_Gas_Infrastructure - National_Onshore_Oil_Pipelines

- Show Pop-ups

**Pop-up Title**

National_Onshore_Oil_Pipelines: {NAME}

**Pop-up Contents**

- Display: A description from one field
- CLASS {class}

**Configure Attributes**

**Attribute Expressions**

Adding expressions allows you to create new information from existing fields for use in pop-ups.

ADD

No expressions.
Click 'Add' to add one.

OK  CANCEL

Click on a tract to see the pop-up.
5.10 8. Save the map and create a web app to share

Click “Save”.

Chapter 5. GSKY interacts with ArcGIS suite
Type the map name and tags. Click “Save maps”.

Click “Share”.
Click “Create a web app”. The link to the map will automatically pop up.

Share

Choose who can view this map.

Your map is currently shared with these people.

☑ Everyone (public)

Link to this map

http://arcg.is/1TOrOK

☑ Share current map extent

Embed this map

[EMBED IN WEBSITE]  [CREATE A WEB APP]

Choose a template.
Create a New Web App

What do you want to do? Select a configurable app.

Show All
- Build a Story Map
- Collect/Edit Data
- Compare Maps/Layers
- Explore/Summarize Data
- Interpret Imagery
- Map Social Media
- Provide Local Information
- Route/Get Directions
- Showcase a Map

Click “Create web app”.

5.10. 8. Save the map and create a web app to share
Create a New Web App

What do you want to do?

- Build a Story Map
- Collect/Edit Data
- Compare Maps/Layers
- Explore/Summarize Data
- Interpret Imagery
- Map Social Media
- Provide Local Information
- Route/Get Directions
- Showcase a Map

Select a configurable app.

Image Mask
Creates an app to visualize changes in an imagery layer.

Enter title, summary. Click “Done” and wait until map is generated.
Configure: Australian vegetation and Oil and Gas pipeline

Select Web Map

Title for ArcGIS Online item:
Australian vegetation and oil/gas pipeline

Description for ArcGIS Online item:
Australian vegetation and oil/gas pipeline
Click “Save”. Click “Launch”. Copy the URL for sharing.
This tutorial demonstrates how to load the GSKY WMS layer into QGIS. In this tutorial, you will learn

- how to add the layer by calling GSKY WMS server
- how to load the layer by calling GSKY WCS server

### 6.1 Introduction

QGIS is a user-friendly Open Source Geographic Information System (GIS) licensed under the GNU General Public License. QGIS is an official project of the Open Source Geospatial Foundation (OSGeo). It runs on Linux, Unix, Mac OSX, Windows and Android and supports numerous vector, raster, and database formats and functionalities.

### 6.2 0. Prerequisite

You need to install QGIS on your desktop, or you can run QGIS on NCI’s VDI. We demonstrate how to load GSKY layer into QGIS on VDI in this tutorial.
6.3 1. Launch QGIS

6.4 2. Add GSKY WMS layer

Click “Layer | Add Layer | Add WMS/WMTS Layer”
In the “Data Source Manager”, click “New”
Type “Name” and “URL” https://gsky.nci.org.au/ows/geoglam, click “OK”.
In the “Data Source Manager”, click “Connect”
A list of GSKY layers is presented in the box.
Click “Add” and then “close” the window. Choose “WMS/WMTS | NCI GSKY | Layer name”.

6.4. 2. Add GSKY WMS layer
Chapter 6. GSKY interacts with QGIS
You can save the layer by click “Save” button on the menu bar.

6.5 3. Add GSKY WCS layer

Click “Layer | Add Layer | Add WCS Layer”

In the “Data Source Manager”, click “New”
Type “Name” GSKY WCS and “URL” https://gsky.nci.org.au/ows/geoglam, click “OK”.
In the “Data Source Manager”, click “Connect”
A list of GSKY layers is presented in the box.
Time, CRS, and Format can be selected from this window.
Click “Time” and then scroll up and down to choose the time
3. Add GSKY WCS layer
Once you are happy with your choice, click “Add”.

Use zoom in button to select the area of your interests.
You can save the layer by click “Layer I save as . . . ” button on the menu bar.
You can define your output parameters.
There are a number of output formats can be selected from the list.
<table>
<thead>
<tr>
<th>GeoTIFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>GeoPackage</td>
</tr>
<tr>
<td>ARC Digitized Raster Graphics</td>
</tr>
<tr>
<td>CTable2 Datum Grid Shift</td>
</tr>
<tr>
<td>ELAS</td>
</tr>
<tr>
<td>ENVI .hdr Labelled</td>
</tr>
<tr>
<td>ERMapper .ers Labelled</td>
</tr>
<tr>
<td>ESRI .hdr Labelled</td>
</tr>
<tr>
<td>Erdas .LAN/ GIS</td>
</tr>
<tr>
<td>Erdas Imagine Images (.img)</td>
</tr>
<tr>
<td>Flexible Image Transport System</td>
</tr>
<tr>
<td>Geospatial PDF</td>
</tr>
<tr>
<td>Golden Software 7 Binary Grid (.grd)</td>
</tr>
<tr>
<td>Golden Software Binary Grid (.grd)</td>
</tr>
<tr>
<td>HDF4 Dataset</td>
</tr>
<tr>
<td>ILWIS Raster Map</td>
</tr>
<tr>
<td>ISCE raster</td>
</tr>
<tr>
<td>Idrisi Raster A1</td>
</tr>
<tr>
<td>Image Data and Analysis</td>
</tr>
<tr>
<td>Intergraph Raster</td>
</tr>
<tr>
<td>KOLOR Raw</td>
</tr>
<tr>
<td>Leveller heightfield</td>
</tr>
<tr>
<td>MBTiles</td>
</tr>
<tr>
<td>MS Windows Device Independent Bitmap</td>
</tr>
<tr>
<td>Meta Raster Format</td>
</tr>
<tr>
<td>NOAA Vertical Datum .GTX</td>
</tr>
<tr>
<td>NTv2 Datum Grid Shift</td>
</tr>
<tr>
<td>National Imagery Transmission Format</td>
</tr>
<tr>
<td>Network Common Data Format</td>
</tr>
<tr>
<td>Northwood Numeric Grid Format .grd/tab</td>
</tr>
<tr>
<td>PCI .aux Labelled</td>
</tr>
<tr>
<td>PCIDSK Database File</td>
</tr>
<tr>
<td>PCRaster Raster File</td>
</tr>
<tr>
<td>Portable pixmap Format (.netpbm)</td>
</tr>
<tr>
<td>ROI_PAC raster</td>
</tr>
<tr>
<td>Raster Matrix Format</td>
</tr>
<tr>
<td>SAGA GIS Binary Grid (.sdat)</td>
</tr>
<tr>
<td>SGI Image File Format 1.0</td>
</tr>
<tr>
<td>Terragen heightfield</td>
</tr>
<tr>
<td>USGS Astrogeology ISIS cube (Version 2)</td>
</tr>
<tr>
<td>USGS Astrogeology ISIS cube (Version 3)</td>
</tr>
<tr>
<td>VTR .ht (Binary Terrain) 1.3 Format</td>
</tr>
</tbody>
</table>
We recommend installing Miniconda, a mini version of Anaconda which conveniently installs:

- Python
- Jupyter Notebook
- Other commonly used packages for scientific computing and data science

Miniconda includes only conda and its dependencies. If you would like to have conda plus over 720 open source packages, install Anaconda.

### 7.1 Installing Miniconda on macOS

1. **Download the installers**
   - Miniconda installer for macOS

2. **Install**
   - **Miniconda** via terminal:

     ```
     $ cd ~/Downloads
     $ bash Miniconda3-latest-MacOSX-x86_64.sh
     ```

3. Close and re-open your terminal window.

4. To test that the installation was successful, run the following command:

   ```
   $ conda list
   ```
7.2 Setting up virtual environment

Let’s first set up a virtual environment to work in where we can install all the required modules for the tutorials:

```
$ conda create -n GSKY python=3.6
```

Now let’s activate our virtual environment:

```
$ source activate GSKY
```

7.3 Installing Jupyter

Run the following command to install Jupyter:

```
$ conda install jupyter
```

7.4 Clone GSKY Jupyter notebooks

Open up or create a working directory where we can clone the GSKY Jupyter notebooks:

```
$ git clone https://github.com/nci-training/gsky-demos.git
$ cd gsky-demos
```

For these tutorials we will need to install ipyleaflet, ipywidgets, owslib, matplotlib, gdal, PIL, netCDF4, and Bokeh

```
$ conda install -c conda-forge ipyleaflet ipywidgets owslib matplotlib
g$ conda install gdal pillow netcdf4
$ conda install -c bokeh bokeh
```

7.5 Running notebooks

We should now be ready to run the Jupyter notebook tutorials. To launch the notebooks:

```
$ jupyter notebook < notebook.ipynb >
```

For example, to launch Notebook_GSKY_WMS_ipyleaflet.ipynb:

```
$ jupyter notebook Notebook_GSKY_WMS_ipyleaflet.ipynb
```
8.1 Requesting map images through NCI’s GSKY Data Server

NCI’s GSKY Data Server supports the Open Geospatial Consortium (OGC) Web Map Service (WMS), which is a standard protocol for serving geospatial data as images (e.g., PNG).

The following libraries will need to be imported for this example:

```
[1]: from owslib.wms import WebMapService
    from PIL import Image
    import matplotlib.pyplot as plt
    %matplotlib inline
```

To start, we will need the base GSKY server URL:

```
```

Using OWSLib, we can begin by inspecting the service metadata:

```
[3]: wms = WebMapService(gsky_url, version='1.3.0')
```

To find out the available data layers that can be requested:

```
[4]: for layer in list(wms.contents):
    print("Layer Name:", wms[layer].name)
    print("Title:", wms[layer].title, "\n")
Layer Name: blend_sentinel2_landsat_nbart_daily
Title: Multi-sensor (Landsat and Sentinel 2) surface reflectance (Beta)

Layer Name: hltc_high
Title: DEA High Tide Composite 25m v2.0

Layer Name: hltc_low
Title: DEA Low Tide Composite 25m v2.0
```

(continues on next page)
Layer Name: item_relative
Title: DEA Intertidal Extents Model Relative Layer 25m v2.0

Layer Name: item_stddev
Title: DEA Intertidal Extents Model Confidence Layer 25m v2.0

Layer Name: landsat5_geomedian
Title: DEA Landsat 5 terrain corrected surface reflectance geometric median

Layer Name: landsat5_nbar_16day
Title: 16-day DEA Landsat 5 surface reflectance

Layer Name: landsat5_nbar_daily
Title: Daily DEA Landsat 5 surface reflectance

Layer Name: landsat5_nbart_16day
Title: 16-day DEA Landsat 5 terrain corrected surface reflectance

Layer Name: landsat5_nbart_daily
Title: Daily DEA Landsat 5 terrain corrected surface reflectance

Layer Name: landsat7_geomedian
Title: DEA Landsat 7 terrain corrected surface reflectance geometric median

Layer Name: landsat7_nbar_16day
Title: 16-day DEA Landsat 7 surface reflectance

Layer Name: landsat7_nbar_daily
Title: Daily DEA Landsat 7 surface reflectance

Layer Name: landsat7_nbart_16day
Title: 16-day DEA Landsat 7 terrain corrected surface reflectance

Layer Name: landsat7_nbart_daily
Title: Daily DEA Landsat 7 terrain corrected surface reflectance

Layer Name: landsat8_geomedian
Title: DEA Landsat 8 terrain corrected surface reflectance geometric median

Layer Name: landsat8_nbar_16day
Title: 16-day DEA Landsat 8 surface reflectance

Layer Name: landsat8_nbar_daily
Title: Daily DEA Landsat 8 surface reflectance

Layer Name: landsat8_nbart_16day
Title: 16-day DEA Landsat 8 terrain corrected surface reflectance

Layer Name: landsat8_nbart_daily
Title: Daily DEA Landsat 8 terrain corrected surface reflectance

Layer Name: sentinel2_nbart_daily
Title: Sentinel 2 Analysis Ready Data

Layer Name: wofs
Title: DEA Water Observation Feature Layer
We can also view metadata that is available about a selected layer. For example, you can view the abstract associated with that data layer.

```python
[5]: layer = "landsat8_nbart_16day"

[6]:
print(wms[layer].abstract)
```

This product has been corrected to remove the influences of the atmosphere, the time of year, terrain shadow and satellite view angles using the methods described in Li et al. 2012 https://doi.org/10.1016/j.rse.2012.06.018. Landsat 8 Operational Land imager (OLI) data is available from March 2013 and onwards. More detailed information about the terrain corrected surface reflectance product suite produced using Digital Earth Australia including CCBY4.0 is available at http://dx.doi.org/10.4225/25/5a7a76d2e129e. This service provides access to Landsat 8 OLI terrain corrected surface reflectance data. The image composites are made from images acquired within a 16 day period, and may include clouds.

Or view the CRS options, bounding box, and time positions available (these details will be needed to construct the GetMap request):

```python
[7]:
print("CRS Options: ")
crs = sorted(wms[layer].crsOptions)
print('  ', crs, '

print("Bounding Box: ")
bbox = wms[layer].boundingBox
print('  ', bbox, '

print("Time Positions: ")
time = wms[layer].timepositions
print('  ', time[:10], '

print("Styles: ")
styles = wms[layer].styles
print('  ', styles, '
```
Now let’s use the information above to construct and make the GetMap request.

We’ll need to define a bounding box for our request:

```
subset_bbox = (148.75, -35.93, 149.4, -35.13)
```

Note that there are currently server side limits on the size of a WMS request. If you request bounding boxes that are too large, you will run into a **"zoom into view this layer"** error message. This means you would need to reduce the size of your bounding box request.

**OWSLib**’s library can now be used to make the GetMap request:

```
output = wms.getmap(layers=[layer],
                   Styles='fc',
                   srs='EPSG:4326',
                   bbox=(subset_bbox[0], subset_bbox[1], subset_bbox[2], subset_bbox[3]),
                   size=(512, 512),
                   format='image/png',
                   time='2013-04-07T00:00:00.000Z')
```

To view the above constructed URL:

```
print(output.geturl())
```

http://gsky.nci.org.au/ows/dea?SERVICE=WMS&service=WMS&version=1.3.0&request=GetMap&layers=landsat8_nbart_16day&styles=&width=512&height=512&crs=EPSG%3A4326&bbox=-35.93%2C148.75%2C-35.13%2C149.4&format=image%2Fpng&transparent=FALSE&bgcolor=0xFFFFFF&exceptions=XML&time=2013-04-07T00:00:00.000Z&Styles=fc

Lastly, we need to write the GetMap result to a file:

```
with open(pngfile, 'wb') as out:
    out.write(output.read())
```

And if we’d like to confirm the result:

```
im = Image.open(pngfile)
plt.figure(figsize=(10, 10))
plt.imshow(im)
```
For more information on the OGC WMS standard specifications and the Python OWSLib package: [http://www.opengeospatial.org/standards/wms](http://www.opengeospatial.org/standards/wms) [https://geopython.github.io/OWSLib/#wms](https://geopython.github.io/OWSLib/#wms)
The following libraries will need to be imported for the example below.

```python
[1]: from __future__ import print_function
    from ipyleaflet import WMSLayer, Map, LayersControl
    from ipywidgets import interact, interactive, fixed, interact_manual
    import ipywidgets as widgets
    from IPython.display import display
    from owslib.wms import WebMapService

    wms = WebMapService(gsky_url, version='1.3.0')
    available_layers = list(wms.contents)
```

### 9.1 Defining widgets

Let’s begin by defining some widgets for GSKY layers, times, latitudes and longitudes:

```python
[3]: # LS5 NBAR & NBART layers
    layer = widgets.Select(
        options = available_layers,
        description='Layer:',
        disabled=False
    )

    # Latitudes
    lat_slider = widgets.IntSlider(
        value=37,
        min=-43,
```
9.2 Selecting Layers

Now we can start by first selecting the layer and (lat, lon) region we would like to view:

```python
[4]: display(layer)
display(lat_slider)
display(lon_slider)
Select(description='Layer:', options=('blend_sentinel2_landsat_nbart_daily', 'hltc_high', 'hltc_low', 'item_re...)
IntSlider(value=-10, continuous_update=False, description='Lat:', max=-10, min=-43)
IntSlider(value=113, continuous_update=False, description='Lon:', max=153, min=113)
```

Next, we'll ask GSKY what the available time positions are and select one:

```python
[5]: date = widgets.Select(
    options=wms[format(layer.value)].timepositions,
    description='Date:',
    disabled=False
)
```

```python
[6]: display(date)
Select(description='Date:', options=('1986-01-01T00:00:00.000Z', '1986-01-02T00:00:00.000Z', '1986-01-03T00:00...)
```

```python
[7]: styles = widgets.Select(
    options=wms[layer.value].styles,
    description='Style:',
    disabled=True
)
```
Let's confirm the variables we have chosen:

```
[9]: print('latitude = {0}'.format(lat_slider.value))
    print('longitude = {0}'.format(lon_slider.value))
    print('layer = {0}'.format(layer.value))
    print('date = {0}'.format(date.value))
    print('style = {0}'.format(styles.value))
```

```
latitude = -34
longitude = 127
layer = blend_sentinel2_landsat_nbart_daily
date = 2012-04-12T00:00:00.000Z
style = {'title': 'Normalised Difference Water Index'}
```

### 9.3 Adding layers to the ipyleaflet Map

Let's use the ipyleaflet Map function to plot a map with a center of our lat and lon values.

```
[10]: wms = WMSLayer(url='https://gsky.nci.org.au/ows/dea',
                  layers=layer.value, 
                  name='False Colour',
                  time=date.value, 
                  tile_size=35,
                  transparent=False, 
                  opacity=1, 
                  styles='tc')
```

```
m = Map(layers=(wms,), center=(lat_slider.value, lon_slider.value), zoom=9)
m
```

# Note: we have to zoom in sufficiently so that our gsky wms layer will show.

Map(basemap={'url': 'https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', 'max_zoom': 19, 'attribution': 'Map ...'}

We can add additional layers from the collection and add them to a layer manager.

```
[11]: wms1 = WMSLayer(url='https://gsky.nci.org.au/ows/dea',
                     layers='landsat8_nbar_16day', 
                     name='False Colour',
                     styles='fc',
                     time=date.value, 
                     transparent=True)
```

```
wms2 = WMSLayer(url='http://gsky.nci.org.au/ows/dea',
                layers='landsat8_nbar_16day',
                name='NDVI',
```

(continues on next page)
The following material uses Geoscience Australia’s Earth Observation Collection. Additional information about this data collection can be found [here](#).

For this tutorial we will explore the ipyleaflet Jupyter widgets. For information on ipyleaflet and how to clone and installing, please visit [here](#).
CHAPTER 10

Using GSKY’s WCS in Python

10.1 Requesting data subsets through NCI’s GSKY Data Server

NCI’s GSKY Data Server supports the Open Geospatial Consortium (OGC) Web Coverage Service (WCS), which is a standard protocol for serving geospatial data in common formats such as NetCDF and GeoTIFF.

The following libraries will need to be imported for the below example.

[1]:
```python
from owslib.wcs import WebCoverageService
from PIL import Image
%matplotlib inline
```

To start, we will need the base GSKY server URL:

[2]:
```
gsky_url = 'http://gsky.nci.org.au/ows/dea'
```

Now using OWSLib, we can begin by inspecting the service metadata:

[3]:
```
wcs = WebCoverageService(gsky_url, version='1.0.0')
```

Find out the available data layers that can be requested:

[4]:
```python
for layer in list(wcs.contents):
    print("Layer Name:", layer)
    print("Title:", wcs[layer].title, \\
```
```
Layer Name: blend_sentinel2_landsat_nbart_daily
Title: Multi-sensor (Landsat and Sentinel 2) surface reflectance (Beta)

Layer Name: hltc_high
Title: DEA High Tide Composite 25m v2.0

Layer Name: hltc_low
Title: DEA Low Tide Composite 25m v2.0
```

(continues on next page)
Layer Name: item_relative
Title: DEA Intertidal Extents Model Relative Layer 25m v2.0

Layer Name: item_stddev
Title: DEA Intertidal Extents Model Confidence Layer 25m v2.0

Layer Name: landsat5_geomedian
Title: DEA Landsat 5 terrain corrected surface reflectance geometric median

Layer Name: landsat5_nbar_16day
Title: 16-day DEA Landsat 5 surface reflectance

Layer Name: landsat5_nbar_daily
Title: Daily DEA Landsat 5 surface reflectance

Layer Name: landsat5_nbart_16day
Title: 16-day DEA Landsat 5 terrain corrected surface reflectance

Layer Name: landsat5_nbart_daily
Title: Daily DEA Landsat 5 terrain corrected surface reflectance

Layer Name: landsat7_geomedian
Title: DEA Landsat 7 terrain corrected surface reflectance geometric median

Layer Name: landsat7_nbar_16day
Title: 16-day DEA Landsat 7 surface reflectance

Layer Name: landsat7_nbar_daily
Title: Daily DEA Landsat 7 surface reflectance

Layer Name: landsat7_nbart_16day
Title: 16-day DEA Landsat 7 terrain corrected surface reflectance

Layer Name: landsat7_nbart_daily
Title: Daily DEA Landsat 7 terrain corrected surface reflectance

Layer Name: landsat8_geomedian
Title: DEA Landsat 8 terrain corrected surface reflectance geometric median

Layer Name: landsat8_nbar_16day
Title: 16-day DEA Landsat 8 surface reflectance

Layer Name: landsat8_nbar_daily
Title: Daily DEA Landsat 8 surface reflectance

Layer Name: landsat8_nbart_16day
Title: 16-day DEA Landsat 8 terrain corrected surface reflectance

Layer Name: landsat8_nbart_daily
Title: Daily DEA Landsat 8 terrain corrected surface reflectance

Layer Name: sentinel2_nbart_daily
Title: Sentinel 2 Analysis Ready Data

Layer Name: wofs
Title: DEA Water Observation Feature Layer
We can also view metadata that is available about a selected layer. For example, you can view the abstract associated with that data layer.

```python
layer = "landsat8_nbar_16day"
```

```python
print(wcs[layer].abstract)
```

This product has been corrected to remove the influences of the atmosphere, the time, of year and satellite view angles using the methods described in Li et al. 2010. Landsat 8 Operational Land Imager (OLI) data is available from March 2013 and onwards. More detailed information about the surface reflectance product suite produced using Digital Earth Australia, including CCBY4.0 is available at http://dx.doi.org/10.4225/25/5a7a501e1c5af. This service provides access to Landsat 8 OLI surface reflectance data. The image composites are made from images acquired within a 16 day period, and may include clouds.

```python
print(wcs[layer].styles)
```

```
None
```

Or view the CRS options, bounding box, and time positions available (these details will be needed to construct the GetMap request).

```python
print("CRS Options: ")
crs = sorted(wcs[layer].supportedCRS)
print (\"t\', crs, \"n")

print ("Bounding Box: ")
bbox = wcs[layer].boundingBoxWGS84
print (\"t\', bbox, \"n")

print ("Time Positions: ")
time = wcs[layer].timepositions
print (\"t\', time[-10:], \"n")
```

```
CRS Options:
[urn:ogc:def:crs:EPSG::4326]

Bounding Box:
(-180.0, -90.0, 180.0, 90.0)

Time Positions:
['2019-11-15T00:00:00.000Z', '2019-12-01T00:00:00.000Z', '2019-12-17T00:00:00.000Z', '2020-01-02T00:00:00.000Z', '2020-01-18T00:00:00.000Z', '2020-02-03T00:00:00.000Z', '2020-02-19T00:00:00.000Z', '2020-03-06T00:00:00.000Z', '2020-03-22T00:00:00.000Z', '2020-04-07T00:00:00.000Z']
```

Let’s use the information above to construct and make GetCoverage requests.

The below sections will demonstrate both a request in GeoTIFF and NetCDF formats.

We’ll need to define a bounding box for our request:

```python
subset_bbox = (137,-37,139,-35)
```

OWSLib’s library can now be used to make the GetCoverage request:
To view the above constructed URL:

```
[11]: print(output.geturl())
```

```text
    Coverage=landsat8_nbar_16day&BBox=137%2C-37%2C139%2C-35&time=2013-06-23T00%3A00
    %3A00.000Z&crs=EPSG%3A4326&format=GeoTIFF&width=256&height=256&Styles=fc
```

Write the result to a file:

```
[12]: filename = './output/gsky_wcs.tiff'
    with open(filename, 'wb') as f:
        f.write(output.read())
```

And if we’d like to confirm the result, we can open and view the GeoTIFF with the Python GDAL library for example:

```
[13]: import gdal
    import matplotlib.pyplot as plt
    %matplotlib inline

[14]: ds = gdal.Open(filename)
    band = ds.GetRasterBand(1).ReadAsArray()

    fig = plt.figure(figsize=(10,10))
    plt.imshow(band, cmap='gist_earth')
```

```
<matplotlib.image.AxesImage at 0x122576860>
```
To request a coverage returned as in the NetCDF format, a similar GetCoverage request is constructed with the format parameter now specifying the NetCDF option.

```
[15]: subset_bbox = (137,-37,139,-35)
output = wcs.getCoverage(identifier=layer, Styles='tc',
time=[wcs[layer].timepositions[6]],
bbox=subset_bbox, format='NetCDF',
crs='EPSG:4326', width=256, height=256)
```

```
[16]: print(output.geturl())
   Coverage=landsat8_nbar_16day&BBox=137%2C-37%2C139%2C-35&time=2013-06-23T00%3A00.000Z&
crs=EPSG:4326&format=NetCDF&width=256&height=256&Styles=tc
```

Again, write the output to a file to save:

10.1. Requesting data subsets through NCI's GSKY Data Server
To confirm or inspect the contents of the NetCDF file, libraries such as NetCDF4 Python or GDAL can be used.

```python
[18]: from netCDF4 import Dataset
    import numpy as np

[19]: with Dataset(filename) as ds:
    
    # Set figure size
    plt.figure(figsize=(12,12))

    # Plot image
    plt.imshow(band, extent=[lon[0], lon[-1], lat[-1], lat[0]])

    # Add figure title and labels
    # We can make use of the defined variable attributes to do this
    plt.title(layer.title(), fontsize=20)
    plt.xlabel(lon.long_name+' ('+lon.units+') ', fontsize=16)
    plt.ylabel(lat.long_name+' ('+lat.units+') ', fontsize=16)

    # Adjust tick mark size
    plt.tick_params(labelsize=16)
```

10.1. Requesting data subsets through NCI's GSKY Data Server
The following libraries will need to be imported for the below example.

```
[1]: from owslib.wms import WebMapService
    from owslib.wcs import WebCoverageService
    import ipywidgets as widgets
    from IPython.display import display
    from PIL import Image
    import matplotlib.pyplot as plt
    from netCDF4 import Dataset
    import numpy as np
    %matplotlib inline
```

### 11.1 Preview available data

Let’s preview the available data first with WMS

```
[3]: wms = WebMapService(gsky_url, version='1.3.0')

Find out the available data layers that can be requested:

```
[4]: for layer in list(wms.contents):
    print("Layer Name:", wms[layer].name)
    print("Title:", wms[layer].title, '\n')

Layer Name: blend_sentinel2_landsat_nbart_daily
Title: Multi-sensor (Landsat and Sentinel 2) surface reflectance (Beta)

Layer Name: hltc_high
Title: DEA High Tide Composite 25m v2.0
```
Layer Name: hltc_low
Title: DEA Low Tide Composite 25m v2.0

Layer Name: item_relative
Title: DEA Intertidal Extents Model Relative Layer 25m v2.0

Layer Name: item_stddev
Title: DEA Intertidal Extents Model Confidence Layer 25m v2.0

Layer Name: landsat5_geomedian
Title: DEA Landsat 5 terrain corrected surface reflectance geometric median

Layer Name: landsat5_nbar_16day
Title: 16-day DEA Landsat 5 surface reflectance

Layer Name: landsat5_nbar_daily
Title: Daily DEA Landsat 5 surface reflectance

Layer Name: landsat5_nbart_16day
Title: 16-day DEA Landsat 5 terrain corrected surface reflectance

Layer Name: landsat5_nbart_daily
Title: Daily DEA Landsat 5 terrain corrected surface reflectance

Layer Name: landsat7_geomedian
Title: DEA Landsat 7 terrain corrected surface reflectance geometric median

Layer Name: landsat7_nbar_16day
Title: 16-day DEA Landsat 7 surface reflectance

Layer Name: landsat7_nbar_daily
Title: Daily DEA Landsat 7 surface reflectance

Layer Name: landsat7_nbart_16day
Title: 16-day DEA Landsat 7 terrain corrected surface reflectance

Layer Name: landsat7_nbart_daily
Title: Daily DEA Landsat 7 terrain corrected surface reflectance

Layer Name: landsat8_geomedian
Title: DEA Landsat 8 terrain corrected surface reflectance geometric median

Layer Name: landsat8_nbar_16day
Title: 16-day DEA Landsat 8 surface reflectance

Layer Name: landsat8_nbar_daily
Title: Daily DEA Landsat 8 surface reflectance

Layer Name: landsat8_nbart_16day
Title: 16-day DEA Landsat 8 terrain corrected surface reflectance

Layer Name: landsat8_nbart_daily
Title: Daily DEA Landsat 8 terrain corrected surface reflectance

Layer Name: sentinel2_nbart_daily
Title: Sentinel 2 Analysis Ready Data
Layer Name: wofs
Title: DEA Water Observation Feature Layer

We can also view metadata that is available about a selected layer. For example, you can view the abstract associated with that data layer.

```
[5]: layer = "landsat8_nbart_16day"
```

```
[6]: print(wms[layer].abstract)
```

This product has been corrected to remove the influences of the atmosphere, the time of year, terrain shadow and satellite view angles using the methods described in Li et al. 2012 https://doi.org/10.1016/j.rse.2012.06.018. Landsat 8 Operational Land Imager (OLI) data is available from March 2013 and onwards. More detailed information about the terrain corrected surface reflectance product suite produced using Digital Earth Australia including CCBY4.0 is available at http://dx.doi.org/10.4225/25/5a7a76d2e129e. This service provides access to Landsat 8 OLI terrain corrected surface reflectance data. The image composites are made from images acquired within a 16 day period, and may include clouds.

```
[7]: print(wms[layer].styles)
```

```
```

Let’s request a preview of the dataset through WMS

WMS function

```
[8]: def get_map(layer, bbox, time, Styles):
   output = wms.getmap(layers=[layer],
                      srs=wms[layer].crsOptions[0],
                      bbox=bbox,
                      size=(256, 256),
                      format='image/png',
                      time=time,
                      Styles=style)
   pngfile = './output/gsky_getMap.png'
   with open(pngfile, 'wb') as out:
       out.write(output.read())
```

11.1. Preview available data
return pngfile

Make a request

Note that there are currently server side limits on the size of a WMS request. If you request bounding boxes that are too large, you will run into a *"zoom into view this layer"* error message. This means you would need to reduce the size of your bounding box request.

```python
print("Time Positions: ")
time = wms[layer].timepositions
print(' 
	', time[:10], '
')
Time Positions:
['2013-01-01T00:00:00.000Z', '2013-01-17T00:00:00.000Z', '2013-02-02T00:00:00.000Z',
 '2013-02-18T00:00:00.000Z', '2013-03-06T00:00:00.000Z', '2013-03-22T00:00:00.000Z',
 '2013-04-07T00:00:00.000Z', '2013-04-23T00:00:00.000Z', '2013-05-09T00:00:00.000Z',
 '2013-05-25T00:00:00.000Z']
```

```python
print(wms[layer].styles)
 {'fc': {'title': 'False colour'}, 'tc': {'title': 'True colour'}, 'NDVI': {
 'title': 'Normalised Difference Vegetation Index', 'legend': 'http://gsky.nci.org.au/ows/dea?
 service=WMS&request=GetLegendGraphic&version=1.3.0&layers=landsat8_nbart_16day&
styles=NDVI', 'legend_width': '160', 'legend_height': '320', 'legend_format': 'image/png'},
layers=landsat8_nbart_16day&styles=NDWI', 'legend_width': '160', 'legend_height': '320', 'legend_format': 'image/png'},
layers=landsat8_nbart_16day&styles=MNDWI', 'legend_width': '160', 'legend_height': '320', 'legend_format': 'image/png'},
 'NBR': {'title': 'Normalised Burn Ratio', 'legend': 'http://gsky.nci.org.au/ows/dea?
 service=WMS&request=GetLegendGraphic&version=1.3.0&layers=landsat8_nbart_16day&
styles=NBR', 'legend_width': '160', 'legend_height': '320', 'legend_format': 'image/png'}}
```

```python
style = 'tc'
request_time = '2013-04-07T00:00:00.000Z'
pngfile = get_map(layer, subset_bbox, request_time, style)
```

11.1.1 And if we'd like to confirm the result:

```python
im = Image.open(pngfile)
plt.figure(figsize=(10, 10))
plt.imshow(im, extent=[subset_bbox[0], subset_bbox[1], subset_bbox[1], subset_bbox[0]])
```
def get_coverage(layer, Styles, bbox, time):
    wcs = WebCoverageService(gsky_url, version='1.0.0')
    output = wcs.getCoverage(identifier=layer,
                              Styles=style,
                              time=time,
                              bbox=subset_bbox,
                              format='NetCDF',
                              crs='EPSG:4326',
                              width=256,
                              height=256)
    filename = './output/gsky_wcs.nc'
    with open(filename, 'wb') as f:
        f.write(output.read())

(continues on next page)
return filename

11.2 Select/click region

```python
[17]: lon_range = widgets.IntRangeSlider(
    value=[135, 145],
    min=110,
    max=155,
    step=1,
    description='Longitude:',
    disabled=False,
    continuous_update=False,
    orientation='horizontal',
    readout=True,
    readout_format='d',
)

lat_range = widgets.IntRangeSlider(
    value=[-30, -25],
    min=-45,
    max=-10,
    step=1,
    description='Latitude:',
    disabled=False,
    continuous_update=False,
    orientation='horizontal',
    readout=True,
    readout_format='d',
)

[18]: display(lon_range)
display(lat_range)

IntRangeSlider(value=(135, 145), continuous_update=False, description='Longitude:',
               max=155, min=110)
IntRangeSlider(value=(-30, -25), continuous_update=False, description='Latitude:',
               max=-10, min=-45)

[19]: print('latitude = {0}'.format(lat_range.value))
print('longitude = {0}'.format(lon_range.value))

latitude = (-30, -25)
longitude = (135, 145)

[20]: subset = (lon_range.value[0], lat_range.value[0], lon_range.value[1], lon_range.value[1])

[21]: style = 'fc'
```
11.3 And ship!

```python
wcs_output = get_coverage(layer, style, subset, request_time)

with Dataset(wcs_output) as ds:
    band = ds['Band1']
    lon = ds['lon']
    lat = ds['lat']

    # Set figure size
    plt.figure(figsize=(12, 12))

    # Plot image
    plt.imshow(band, extent=[lon[0], lon[-1], lat[-1], lat[0]])

    # Add figure title and labels
    plt.title(layer.title(), fontsize=20)
    plt.xlabel(lon.long_name + ('+' + lon.units +')', fontsize=16)
    plt.ylabel(lat.long_name + ('+' + lat.units +')', fontsize=16)

    # Adjust tick mark size
    plt.tick_params(labelsize=16)
```
For more information on the OGC WMS standard specifications and the Python OWSLib package: [http://www.opengeospatial.org/standards/wms](http://www.opengeospatial.org/standards/wms) [https://geopython.github.io/OWSLib/#wms](https://geopython.github.io/OWSLib/#wms)
In this notebook,

- request images through GSKY WMS endpoints
- create a GIF using those images

NCI’s GSKY Data Server supports the Open Geospatial Consortium (OGC) Web Map Service (WMS), which is a standard protocol for serving geospatial data as images (e.g., PNG).

In this example, we extract data from the “Multi-sensor (Landsat and Sentinel 2) surface reflectance false colour (Beta)” (DEA products) to view a bushfire in NSW, Australia over two days, 12-13 of September, 2019.

### 12.1 You can view this example via Terria Map.

### 12.2 The following libraries will need to be imported for this example:

```python
[1]: from owslib.wms import WebMapService 
    from PIL import Image, ImageDraw, ImageFilter 
    import numpy 
    import matplotlib.pyplot as plt 
    %matplotlib inline
```

To start, we will need the base GSKY server URL:

```python
```

Using OWSLib, we can begin by inspecting the service metadata to find out the available data layers:

```python
[3]: wms = WebMapService(gsky_url, version='1.3.0') 
    for layer in list(wms.contents): 
        print("Layer Name:", wms[layer].name) 
        print("Title:", wms[layer].title, \\
```

```
```
Layer Name: blend_sentinel2_landsat_nbart_daily
Title: Multi-sensor (Landsat and Sentinel 2) surface reflectance (Beta)

Layer Name: hltc_high
Title: DEA High Tide Composite 25m v2.0

Layer Name: hltc_low
Title: DEA Low Tide Composite 25m v2.0

Layer Name: item_relative
Title: DEA Intertidal Extents Model Relative Layer 25m v2.0

Layer Name: item_stddev
Title: DEA Intertidal Extents Model Confidence Layer 25m v2.0

Layer Name: landsat5_geomedian
Title: DEA Landsat 5 terrain corrected surface reflectance geometric median

Layer Name: landsat5_nbar_16day
Title: 16-day DEA Landsat 5 surface reflectance

Layer Name: landsat5_nbar_daily
Title: Daily DEA Landsat 5 surface reflectance

Layer Name: landsat5_nbart_16day
Title: 16-day DEA Landsat 5 terrain corrected surface reflectance

Layer Name: landsat5_nbart_daily
Title: Daily DEA Landsat 5 terrain corrected surface reflectance

Layer Name: landsat7_geomedian
Title: DEA Landsat 7 terrain corrected surface reflectance geometric median

Layer Name: landsat7_nbar_16day
Title: 16-day DEA Landsat 7 surface reflectance

Layer Name: landsat7_nbar_daily
Title: Daily DEA Landsat 7 surface reflectance

Layer Name: landsat7_nbart_16day
Title: 16-day DEA Landsat 7 terrain corrected surface reflectance

Layer Name: landsat7_nbart_daily
Title: Daily DEA Landsat 7 terrain corrected surface reflectance

Layer Name: landsat8_geomedian
Title: DEA Landsat 8 terrain corrected surface reflectance geometric median

Layer Name: landsat8_nbar_16day
Title: 16-day DEA Landsat 8 surface reflectance

Layer Name: landsat8_nbar_daily
Title: Daily DEA Landsat 8 surface reflectance

Layer Name: landsat8_nbart_16day
Title: 16-day DEA Landsat 8 terrain corrected surface reflectance

Layer Name: landsat8_nbart_daily
Title: Daily DEA Landsat 8 terrain corrected surface reflectance

(continues on next page)
Layer Name: landsat8_nbart_daily  
Title: Daily DEA Landsat 8 terrain corrected surface reflectance

Layer Name: sentinel2_nbart_daily  
Title: Sentinel 2 Analysis Ready Data

Layer Name: wofs  
Title: DEA Water Observation Feature Layer

We can also view metadata that is available about a selected layer. For example, you can view the abstract associated with that data layer.

```
[4]: layer = "blend_sentinel2_landsat_nbart_daily"
```

Or view the CRS options, bounding box, and time positions available (these details will be needed to construct the GetMap request):

```
[5]: print("CRS Options: ")
crs = sorted(wms[layer].crsOptions)
print('"t', crs, '"')

print("Bounding Box: ")
bbox = wms[layer].boundingBox
print('"t', bbox, '"')

print("Time Positions: ")
time = wms[layer].timepositions
print('"t', time[12300:12320], '"')
```

CRS Options:

```
['EPSG:3857', 'EPSG:4326']
```

Bounding Box:
```
(-180.0, -90.0, 180.0, 90.0, 'EPSG:4326')
```

Time Positions:
```
['2019-09-05T00:00:00.000Z', '2019-09-06T00:00:00.000Z', '2019-09-07T00:00:00.000Z', '2019-09-08T00:00:00.000Z', '2019-09-09T00:00:00.000Z', '2019-09-10T00:00:00.000Z', '2019-09-11T00:00:00.000Z', '2019-09-12T00:00:00.000Z', '2019-09-13T00:00:00.000Z', '2019-09-14T00:00:00.000Z', '2019-09-15T00:00:00.000Z', '2019-09-16T00:00:00.000Z', '2019-09-17T00:00:00.000Z', '2019-09-18T00:00:00.000Z', '2019-09-19T00:00:00.000Z', '2019-09-20T00:00:00.000Z', '2019-09-21T00:00:00.000Z', '2019-09-22T00:00:00.000Z', '2019-09-23T00:00:00.000Z', '2019-09-24T00:00:00.000Z']
```

Now, we can construct the bounding box and time selection for our bushfire example in NSW Australia, over two days, 12-13 of September, 2019.

We'll need to define a bounding box for our request:

```
[6]: subset_bbox = (152, -30, 152.45, -30.2)
```

OWSLib’s library can now be used to make the GetMap request and print out the images:

```
12.2. The following libraries will need to be imported for this example:
```
```python
# 2019-09-12 - 09-13
dates = ['2019-09-11T00:00:00.000Z','2019-09-12T00:00:00.000Z','2019-09-13T00:00:00.000Z']
images = []

for date in dates:
    print(date)
    img = wms.getmap(layers=[layer],
                     srs='EPSG:4326',
                     bbox=(subset_bbox[0], subset_bbox[1], subset_bbox[2], subset_bbox[3]),
                     size=(512, 256),
                     format='image/png',
                     time=date)
    im = Image.open(img)
    images.append(im)
    im.save('gsky_getMap%.10s.png' % date)

    im = Image.open(img)
    fig = plt.figure
    imgplot = plt.imshow(im, extent=[subset_bbox[0], subset_bbox[2], subset_bbox[1], subset_bbox[3]])

images[0].save('bushfire_NSW.gif', save_all=True, append_images=images[1:],
               optimize=False, duration=2, loop=0)
```

Another way to make GIF is using https://giphy.com/create/gifmaker, where you can upload all .png files, and create a fancier GIF. For example, you can add annotation.

For more information on the OGC WMS standard specifications and the Python OWSLib package:
- http://www.opengeospatial.org/standards/wms
- https://geopython.github.io/OWSLib/#wms
NCI’s GSKY Data Server supports the Open Geospatial Consortium (OGC) Web Map Service (WMS), which is a standard protocol for serving geospatial data as images (e.g., PNG).

### 13.1 Constructing WMS Requests

To request a spatial subset of data through GSKY using WMS, a **GetMap** URL defining the subset has to be constructed. This URL can be used within a web browser to communicate to the GSKY Data Server, which will return the requested map image of the subset.

**For example, the GetMap request takes the following form:**

```
```

where red indicates required fields, blue are optional, and green are where input values relevant to the dataset and user request need to be defined.

**GetMap parameters:**
### Parameter Table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required/Optional</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>service</td>
<td>Required</td>
<td>WMS</td>
</tr>
<tr>
<td>version</td>
<td>Required</td>
<td>1.1.1 **, 1.3.0 (default)</td>
</tr>
<tr>
<td>request</td>
<td>Required</td>
<td>GetMap</td>
</tr>
<tr>
<td>layers</td>
<td>Required</td>
<td>&lt;variable&gt;</td>
</tr>
<tr>
<td>crs</td>
<td>Required</td>
<td>&lt;crs_value&gt;</td>
</tr>
<tr>
<td>format</td>
<td>Required</td>
<td>image/png, image/png;mode=32bit, image/gif, image/jpeg</td>
</tr>
<tr>
<td>bbox</td>
<td>Required</td>
<td>&lt;ymin,xmin,ymax,xmax&gt;</td>
</tr>
<tr>
<td>width</td>
<td>Required</td>
<td>Image width in pixels</td>
</tr>
<tr>
<td>height</td>
<td>Required</td>
<td>Image height in pixels</td>
</tr>
<tr>
<td>colorscalerange</td>
<td>Optional</td>
<td>&lt;min,max&gt;</td>
</tr>
<tr>
<td>transparent</td>
<td>Optional</td>
<td>True, False</td>
</tr>
<tr>
<td>time</td>
<td>Optional</td>
<td>&lt;time_value&gt;</td>
</tr>
<tr>
<td>Styles</td>
<td>Optional</td>
<td>&lt;style_name&gt;</td>
</tr>
</tbody>
</table>

** Syntax differences for v1.1.1 GetMap requests:
- Use src in place of crs
- The order of bbox values are: <xmin, ymin, xmax, ymax>

### 13.2 WMS GetCapabilities request

Where do you find valid input values?

In order to construct the GetMap URL, a GetCapabilities request can be made to the server. This requests returns an xml describing the available WMS parameters (metadata, services, and data) made available by NCI’s GSKY server.

GetCapabilities example:

```
```

<Service>
    <Name>EWS</Name>
    <Title>GSKY Web Map Service</Title>

<Abstract>
    This service relies on GSKY
    https://geonetwork.nci.org.au/8d6f-4b76-a734-93ac7fbc9
</Abstract>

<KeywordList>
    <Keyword>WFS</Keyword>
    <Keyword>WMS</Keyword>
    <Keyword>GSKY</Keyword>
</KeywordList>

<OnlineResource xlink:type="simple" xlink:role="default" xlink:title="GSKY Web Map Service"/>
</Service>
</WMS_Capabilities>
13.3 WMS GetMap request

Using the information returned from a GetCapabilities request, a GetMap URL can be constructed and then entered into the address bar of any web browser.

Example GetMap request:

```
http://gsky.nci.org.au/ows/dea?
  service=WMS&
  layers=landsat8_nbart_16day&
  crs=EPSG:4326&
  format=image/png&
  request=GetMap&
  height=512&
  width=512&
  version=1.3.0&
  bbox=-37,147,-35,148&
  time=2013-04-20T00:00:00.000Z&
  transparent=FALSE&
  Styles=tc
```

```
  &request=GetMap&height=512&width=512&version=1.3.0&bbox=-37,147,-35,148&
  &time=2013-04-20T00:00:00.000Z&transparent=FALSE&Styles=tc
```

If the URL is correctly formed, the requested map will be displayed.
13.3. WMS GetMap request
CHAPTER 14

Web Coverage Services (WCS)

NCI’s GSKY Data Server supports the Open Geospatial Consortium (OGC) Web Coverage Service (WCS), which is a standard protocol for serving geospatial data in common formats such as NetCDF and GeoTIFF.

14.1 Constructing WCS requests

GSKY’s Web Coverage Service (WCS) allows users to request data or subsets of data in either NetCDF3 or GeoTIFF format. The request is made by constructing a GetCoverage URL, which is then used within a web browser to communicate to the GSKY Data Server.

For example, the GetCoverage request takes the following form:

http://gsky.nci.org.au/ows/dea?service=WCS&version=1.0.0&request=GetCoverage&coverage=value&format=value&Styles=value&bbox=value&crs=value

where red indicates required fields, blue are optional, and green are where input values relevant to the dataset and user request need to be defined.

14.2 GetCoverage parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required/Optional</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>service</td>
<td>Required</td>
<td>WCS</td>
</tr>
<tr>
<td>version</td>
<td>Required</td>
<td>1.0.0</td>
</tr>
<tr>
<td>request</td>
<td>Required</td>
<td>GetCoverage</td>
</tr>
<tr>
<td>coverage</td>
<td>Required</td>
<td>&lt;variable&gt;</td>
</tr>
<tr>
<td>Styles</td>
<td>Required</td>
<td>&lt;variable&gt;</td>
</tr>
<tr>
<td>format</td>
<td>Required</td>
<td>GeoTIFF, GeoTIFF_Float, NetCDF3</td>
</tr>
<tr>
<td>bbox *</td>
<td>Required</td>
<td>&lt;xmin,ymin,xmax,ymax&gt;</td>
</tr>
<tr>
<td>time *</td>
<td>Required/Optional</td>
<td>&lt;time_value&gt;</td>
</tr>
<tr>
<td>srs, or crs</td>
<td>Required/Optional</td>
<td>&lt;srs_value&gt; or &lt;crs_value&gt;</td>
</tr>
</tbody>
</table>
For large files and/or files with a time dimension, these might be necessary. If \texttt{bbox} is not defined, the entire spatial domain will be returned (if server limits allow) and if \texttt{time} is not specified, either the first or sometimes the last timestep is returned.

14.3 WCS GetCapabilities and DescribeCoverage

Where do you find valid input values?

In order to construct the \texttt{GetCoverage} URL, a \texttt{GetCapabilities} request can be made to the server. This request returns an xml describing the available WCS parameters (metadata, services, and data) made available by NCI’s GSKY server. Additional metadata information can also be requested about a specific coverage layer by making a \texttt{DescribeCoverage} request.

\textbf{GetCapabilities example:}

http://gsky.nci.org.au/ows/dea?service=WCS&version=1.0.0&request=GetCapabilities
14.3. WCS GetCapabilities and DescribeCoverage

```
<WCS_Capabilities xmlns="http://www.opengis.net/wcs" xmlns:gml="http://www.opengis.net/gml" xmlns:xlink="http://www.w3.org/1999/xlink" version="1.0.0">
  <Service>
    <fes>NONE</fes>
    <accessConstraints>NONE</accessConstraints>
  </Service>

  <Capability>
    <Request>
      <GetCapabilities>
        <DCPType>
          <HTTP>
            <Get>
              <OnlineResource xlink:href="http://130.56.242.16/ows"/>
            </Get>
          </HTTP>
        </DCPType>
      </GetCapabilities>
    </GetCapabilities>

    <DescribeCoverage>
      <DCPType>
        <HTTP>
          <Get>
            <OnlineResource xlink:href="http://130.56.242.16/ows"/>
          </Get>
        </HTTP>
      </DCPType>
    </DescribeCoverage>

    <GetCoverage>
      <DCPType>
        <HTTP>
          <Get>
            <OnlineResource xlink:href="http://130.56.242.16/ows"/>
          </Get>
        </HTTP>
      </DCPType>
    </GetCoverage>

    <Exception>
      <Format>application/vnd.ogc.se_xml</Format>
    </Exception>
  </Capability>

  <contentMetadata>
    <CoverageOfferingBrief>
      <description>
        Fractional Cover - MODIS, CSIRO Land and Water algorithm, Australia coverage
      </description>
    </CoverageOfferingBrief>
  </contentMetadata>
</WCS_Capabilities>
```
DescribeCoverage example:

http://gsky.nci.org.au/ows/dea?service=WCS&version=1.0.0&coverage=landsat5_nbart_16day&request=
DescribeCoverage

This XML file does not appear to have any style information associated with it. The document text:

```xml
  <CoverageOffering>
    <description>
      This product has been corrected to remove the influences of the atmosphere, the time and satellite view angles using the methods described in Li et al. 2012. https://doi.org/10.1016/j.rse.2012.06.018. Landsat 5 Thematic Mapper (TM) data is available from March to November 2011. More detailed information about the terrain corrected surface reflectance produced using Digital Earth Australia including CCBY4.0 is available at http://dx.doi.org/10.4225/25/5a7a76d2e129e. This service provides access to Landsat 5 terrain corrected surface reflectance data. The image composites are made from images in a 16-day period, and may include clouds.
    </description>
    <name>landsat5_nbart_16day</name>
  </CoverageOffering>
</CoverageDescription>
```

16-day DEA Landsat 5 terrain corrected surface reflectance

100-90.0 180.0

1986-08-15T00:00:00.000Z

2011-11-08T00:00:00.000Z

16-day EnvelopeWithTimePeriod

-180.0 180.0

1986-08-15T00:00:00.000Z

2011-11-08T00:00:00.000Z

EnvelopeWithTimePeriod

EPSC:4326

GridEnvelope

GridLimits

x

y

origin

-999.9875000000001 -1400.0125
14.4 GetCoverage request

Using the information returned from the GetCapabilities and DescribeCoverage requests, a GetMap URL can be constructed and then entered into the address bar of any web browser.

Example GetCoverage (NetCDF format)

http://gsky.nci.org.au/ows/dea?
   service=WCS&
   coverage=landsat8_nbart_16day&
   crs=EPSG:4326&
   format=NetCDF&
   request=GetCoverage&
   height=256&
   width=256&
   version=1.0.0&
   bbox=147,-37,148,-35&
   time=2013-04-20T00:00:00.00Z&
   Styles=tc

   &height=256&width=256&version=1.0.0&bbox=147,-37,148,-35&coverage=landsat8_nbart_16day
   &time=2013-04-20T00:00:00.00Z&Styles=tc.

Example GetCoverage (GeoTIFF format)

http://gsky.nci.org.au/ows/dea?
   service=WCS&
   coverage=landsat8_nbart_16day&
   crs=EPSG:4326&
   format=GeoTIFF&
   request=GetCoverage&
   height=256&
   width=256&
   version=1.0.0&
   bbox=147,-37,148,-35&
   time=2013-04-20T00:00:00.00Z&
   Styles=tc

   &height=256&width=256&version=1.0.0&bbox=147,-37,148,-35&coverage=landsat8_nbart_16day
   &time=2013-04-20T00:00:00.00Z&Styles=tc.
Please visit the GSKY GitHub page for the source code and compilation instructions.
License

Copyright 2016, 2017, 2018 Australian National University

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this package except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.
CHAPTER 17

Need help?

If you have trouble, please submit a help request to the NCI User Services (help@nci.org.au).