Computational Fabrication

CS 491 and 591
Professor: Leah Buechley
https://handandmachine.cs.unm.edu/classes/Computational_Fabrication_Spring2021/

If you haven’t already, download & install QGIS. Link on website

Weekly artist:Bryan Czibesz

https://bryanczibesz.com/

Bryan Czibesz

Bryan Czibesz

Bryan Czibesz

Bryan Czibesz

Assignment 3 posted

questions?

Data Driven Design cont.

2D Data: Images & Maps

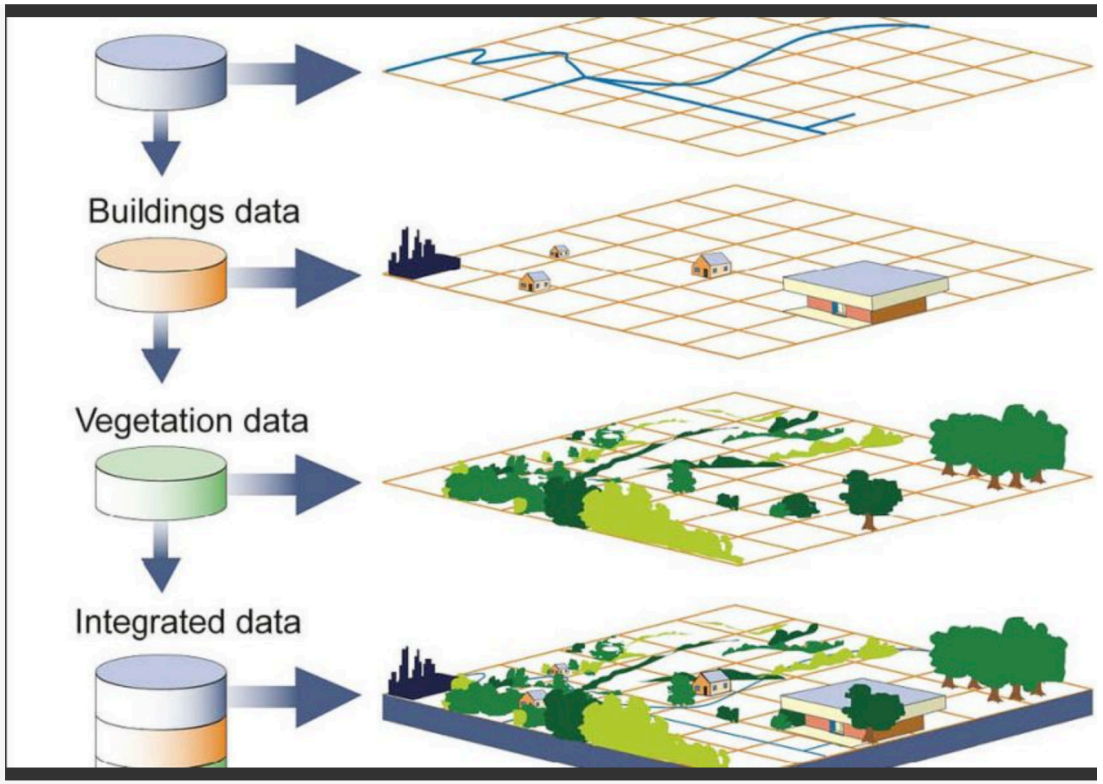
GIS, tip of the iceberg

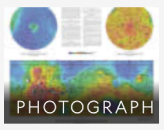
PHOTOGRAPH

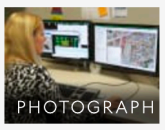
GIS

A geographic information system (GIS) is a computer system for capturing, storing, checking, and displaying data related to positions on Earth's surface. GIS can show many different kinds of data on one map, such as streets, buildings, and vegetation. This enables people to more easily see, analyze, and understand patterns and relationships.

ILLUSTRATION COURTESY OF U.S. GOVERNMENT ACCOUNTABILITY OFFICE







GIS, tip of the iceberg

- GIS = Geographic Information System
  Data may include:
  - GPS locations
  - 3D models of buildings and other structures
  - Environmental data
  - Topographical/elevation data
  - etc.

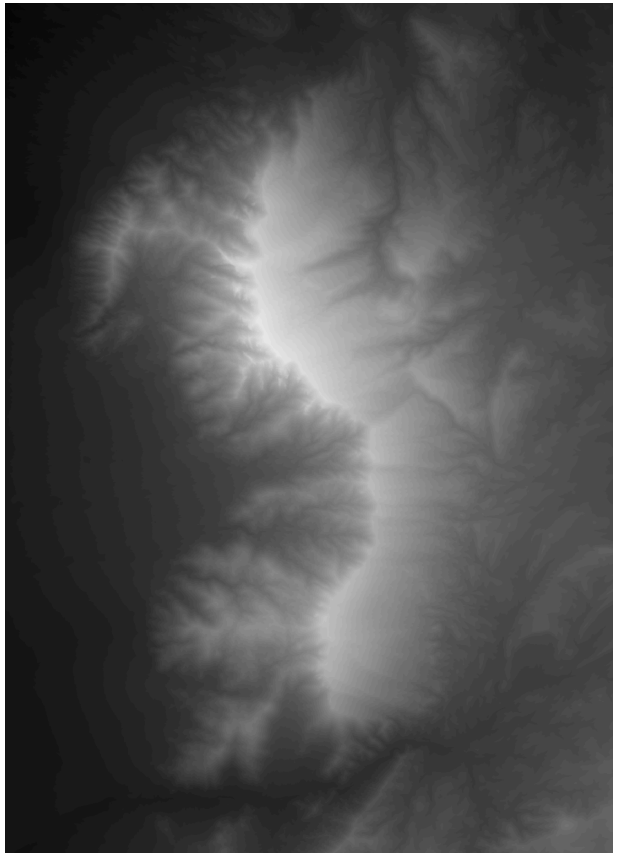
- Tons of available open data: https://apps.nationalmap.gov/
- Open source GIS software, QGIS: https://www.qgis.org/en/site/

Example Workflow Overview

- Download a GeoTIFF from: https://apps.nationalmap.gov/
- Use QGIS to convert GeoTIFF to an image. Import the GeoTIFF as a new layer in a project. Then go to Project—>Export Map as Image to save the image as a bitmap.
- Import bitmap into GH/Rhino to map pixel data to 3D printable geometry

GeoTIFFs

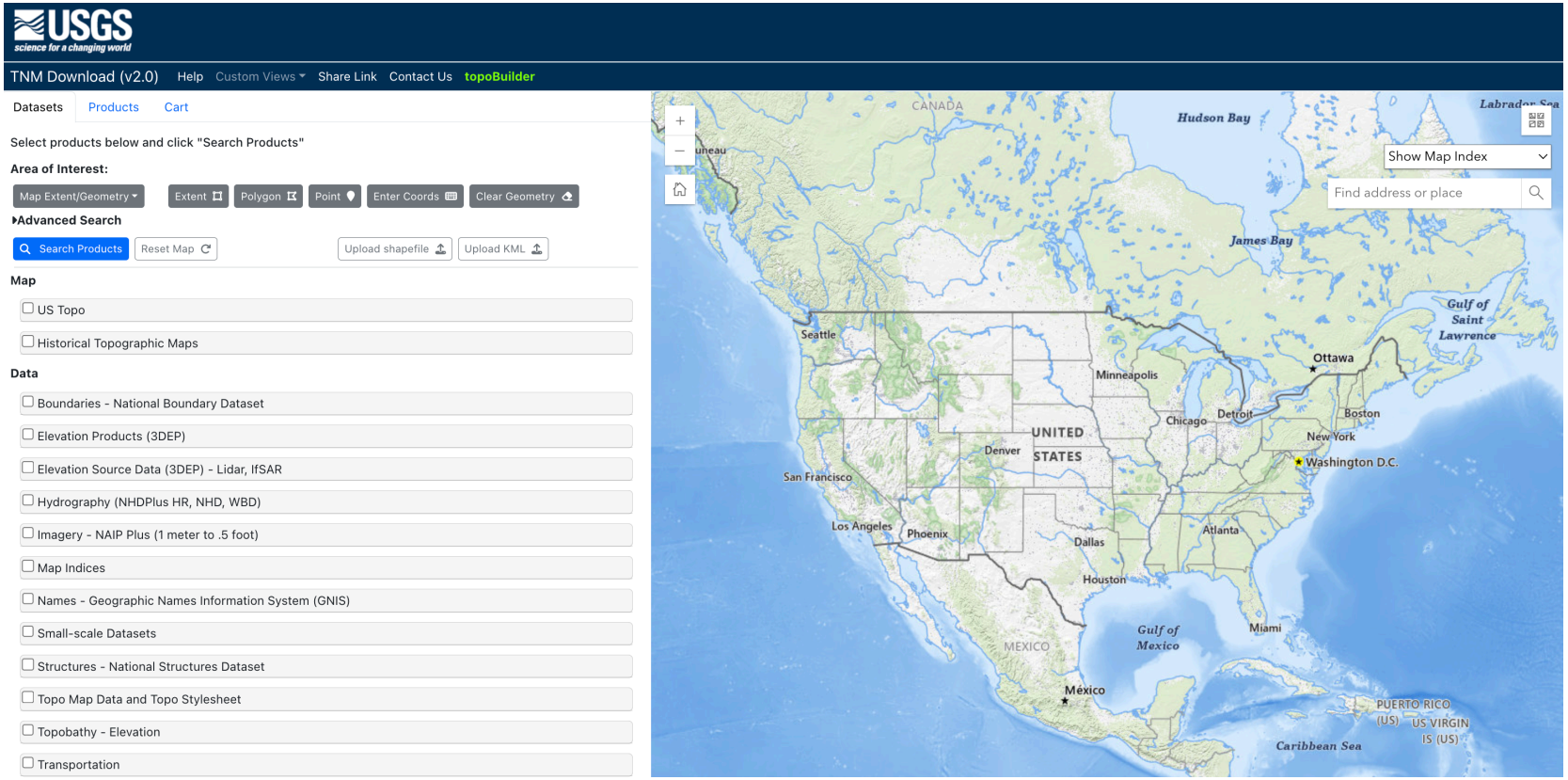
- An image that represents map elevation data
- Each pixel represents the elevation of a unit of area
- Elevation (in meters) is mapped to a greyscale color



What we're going to do today

- Download a GeoTIFF from USGS website
- Use the image to create a 3D printable solid representation of the topography of the Sandia Mountains.

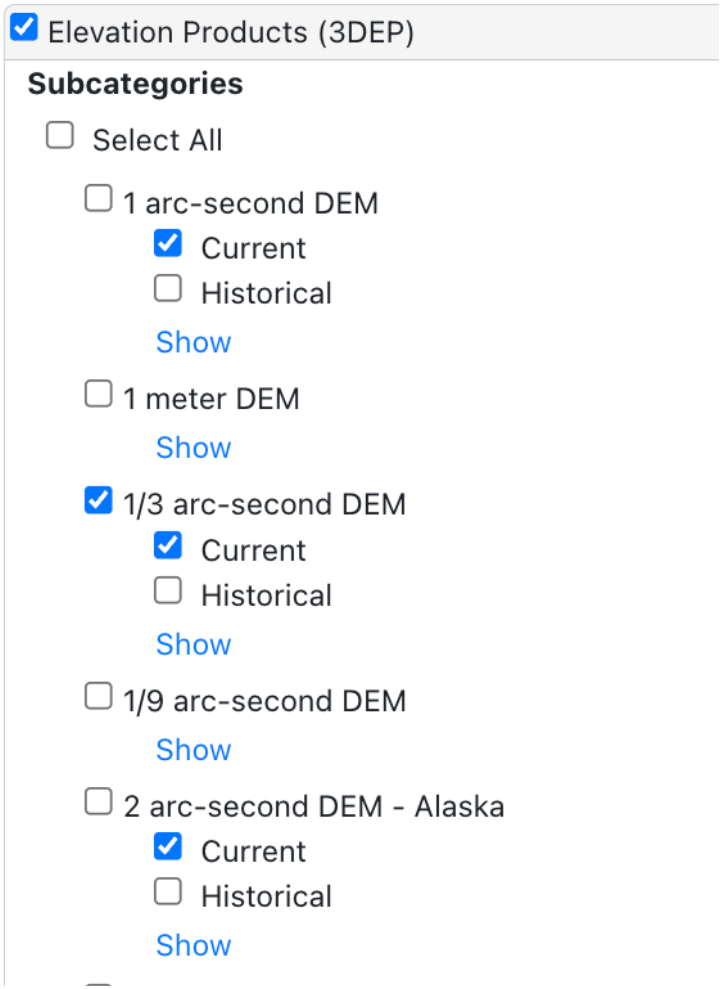
GeoTIFF Maps from USGS



Open: https://apps.nationalmap.gov/

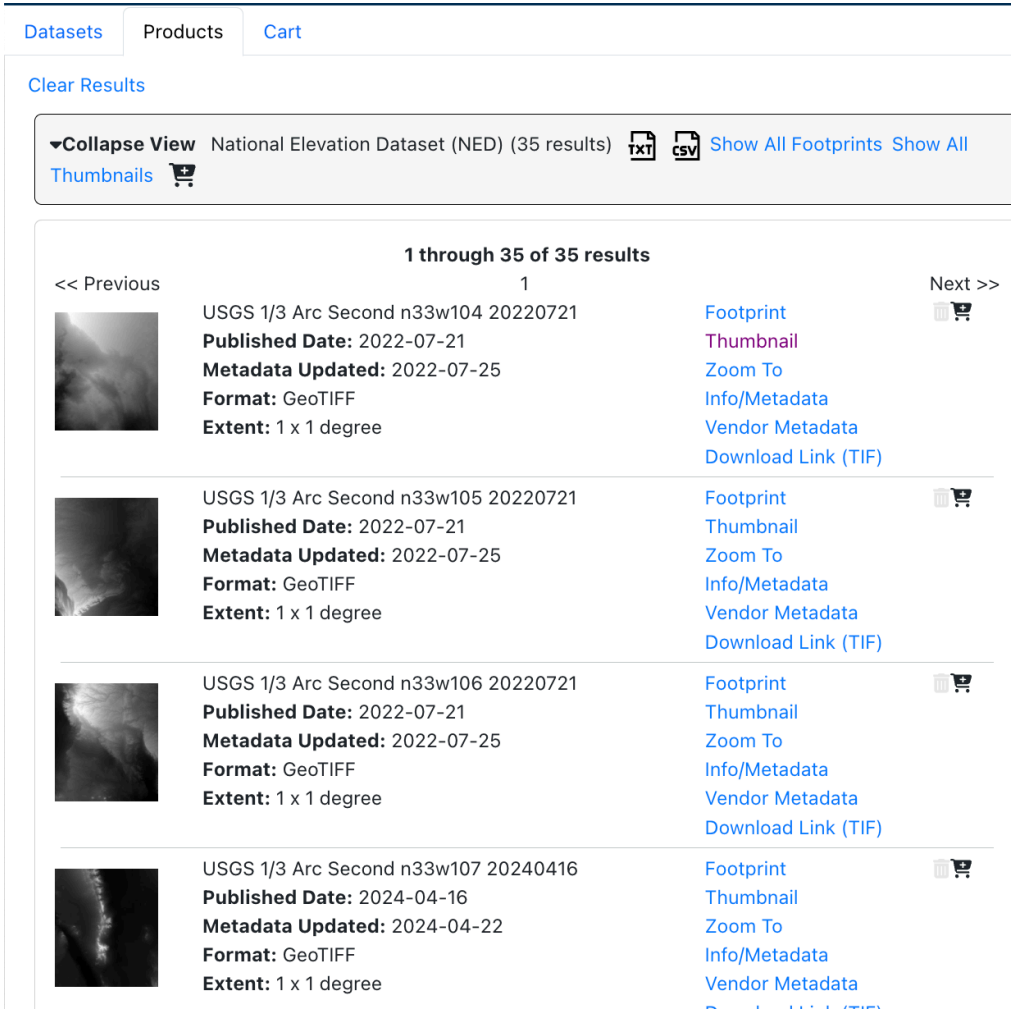
GeoTIFF Maps from USGS

- Open the TNM Download application
- Under Datasets, select:
- Elevation Products
- File Format: GeoTIFF
- 2 arc second DEM (lowest res)
- 1 arc second DEM (lower res)
- 1/3 arc second DEM (higher res)
- 1/9 arc second DEM (highest res)

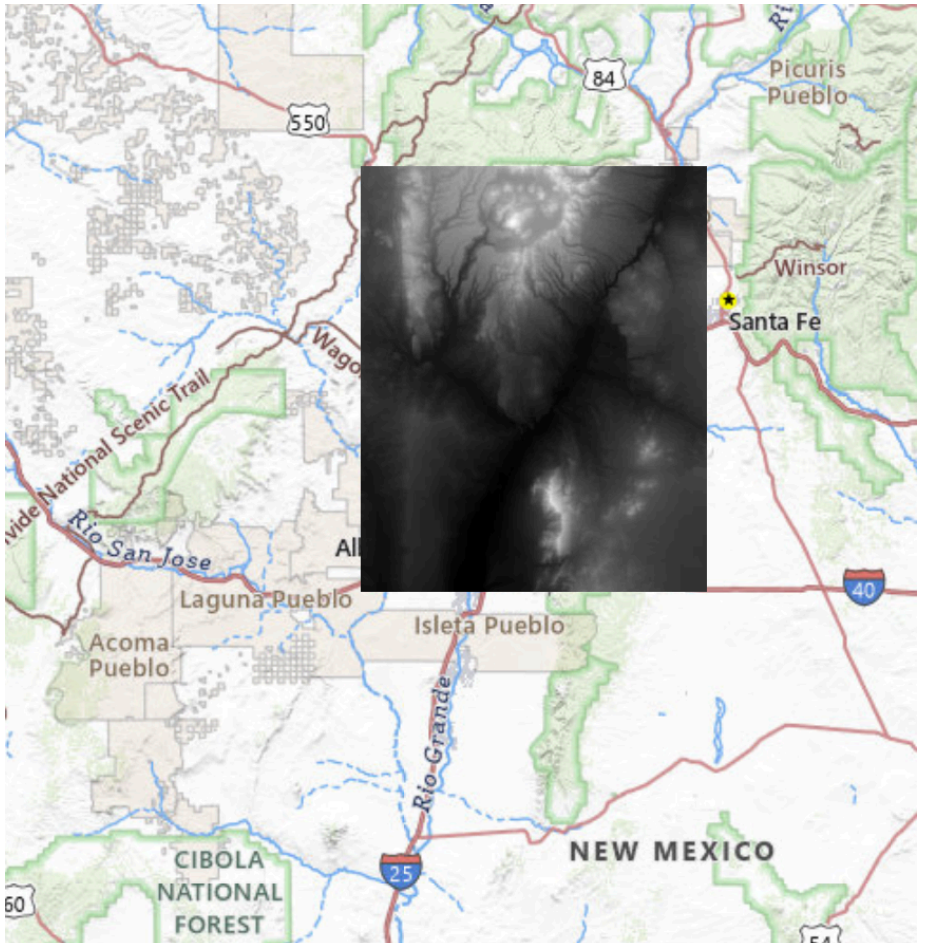


GeoTIFF Maps from USGS

- Click Search Products
- Open Products tab
- Click Thumbnail to view preview image on map
- Click Donwnload Link (TIF) to download file.
- Note: not a regular image file. Can't be opened with image software.



GeoTIFF Preview: 1/3 arc second DEM



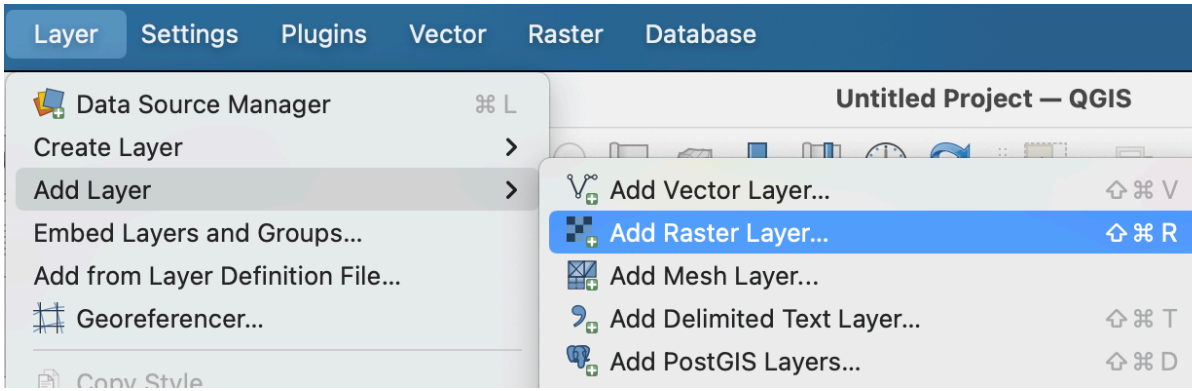
download a GeoTIFF

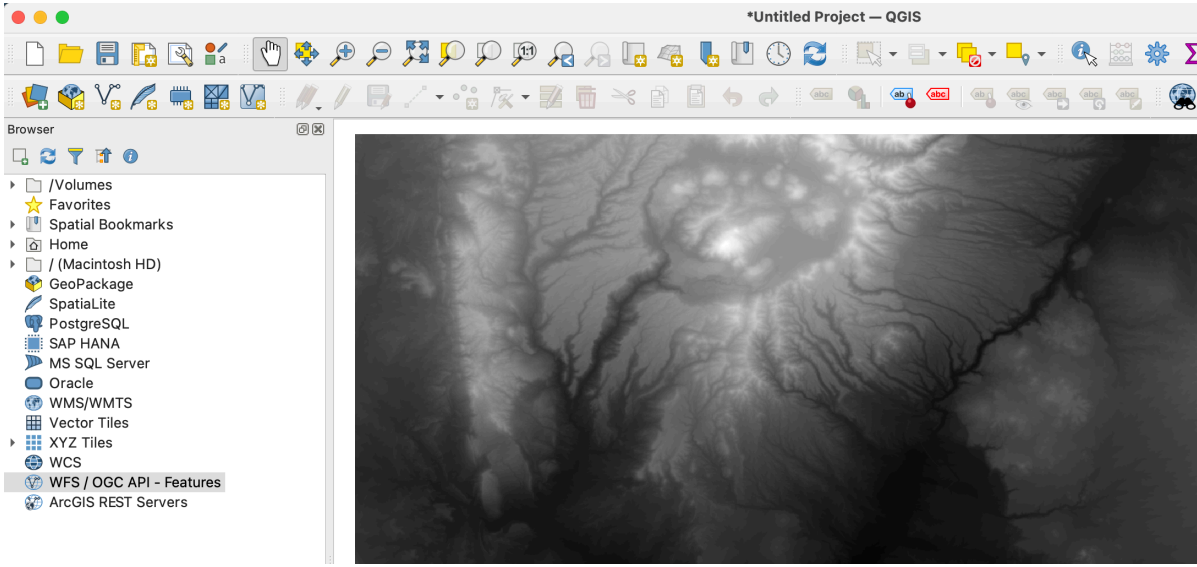
QGIS

- Powerful open source GIS software
- Similar to ArcGIS
  (commercial software)
- Previously Quantum GIS
- Disclaimer: I'm learning with you
- https://www.qgis.org/
- https://github.com/qgis/QGIS

Open GeoTIFF in QGIS

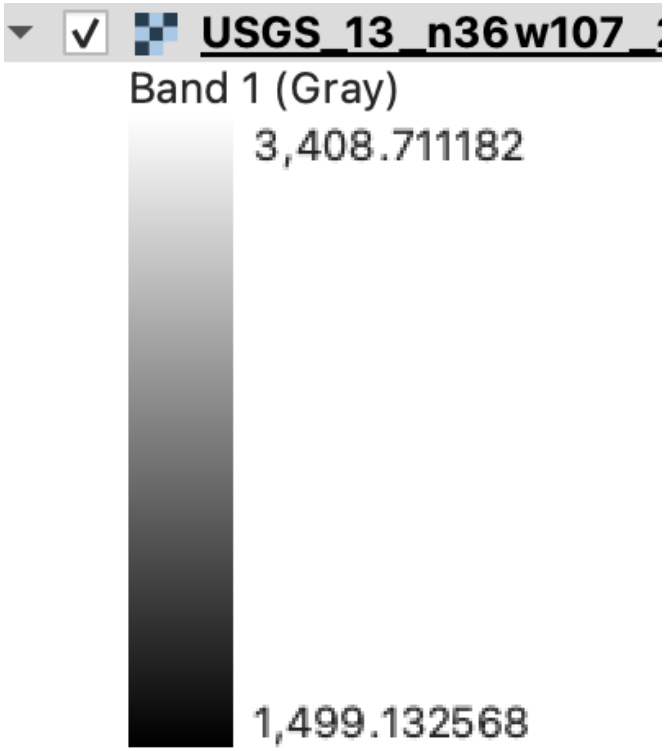
- Create a new project
- Under Layer menu:
- Select Add Layer
- Select Add Raster Layer
- Browse to GeoTIFF
- Click Add
- GeoTIFF appears in window





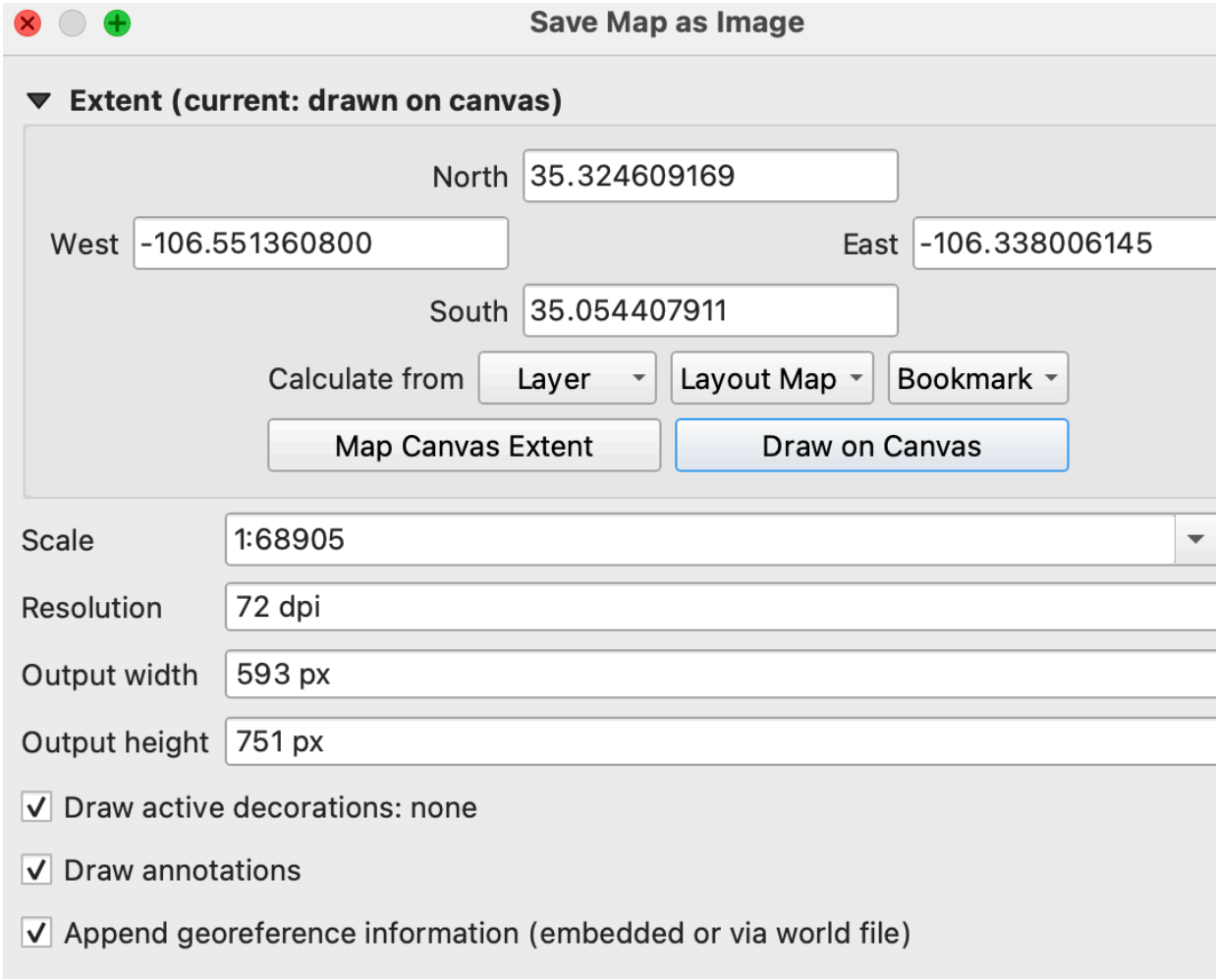
Elevation in meters

- Save this information for unit conversion.
- White: 3408.7 m
- Black: 1499.1 m



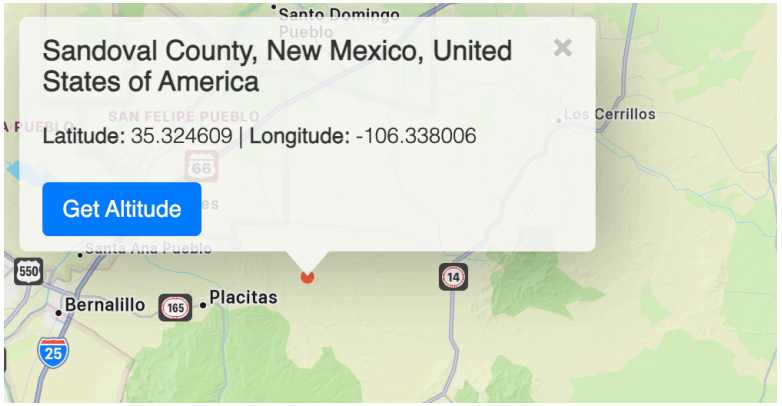
Export GEOTiff as bitmap

- Zoom into the area you want in the image
- Under Project menu:
- Select Import/Export
- Export Map to Image
- Click "Draw on Canvas" to choose the exact area to export.
- Take a screenshot of the export window
- Save image as a bitmap (.BMP)

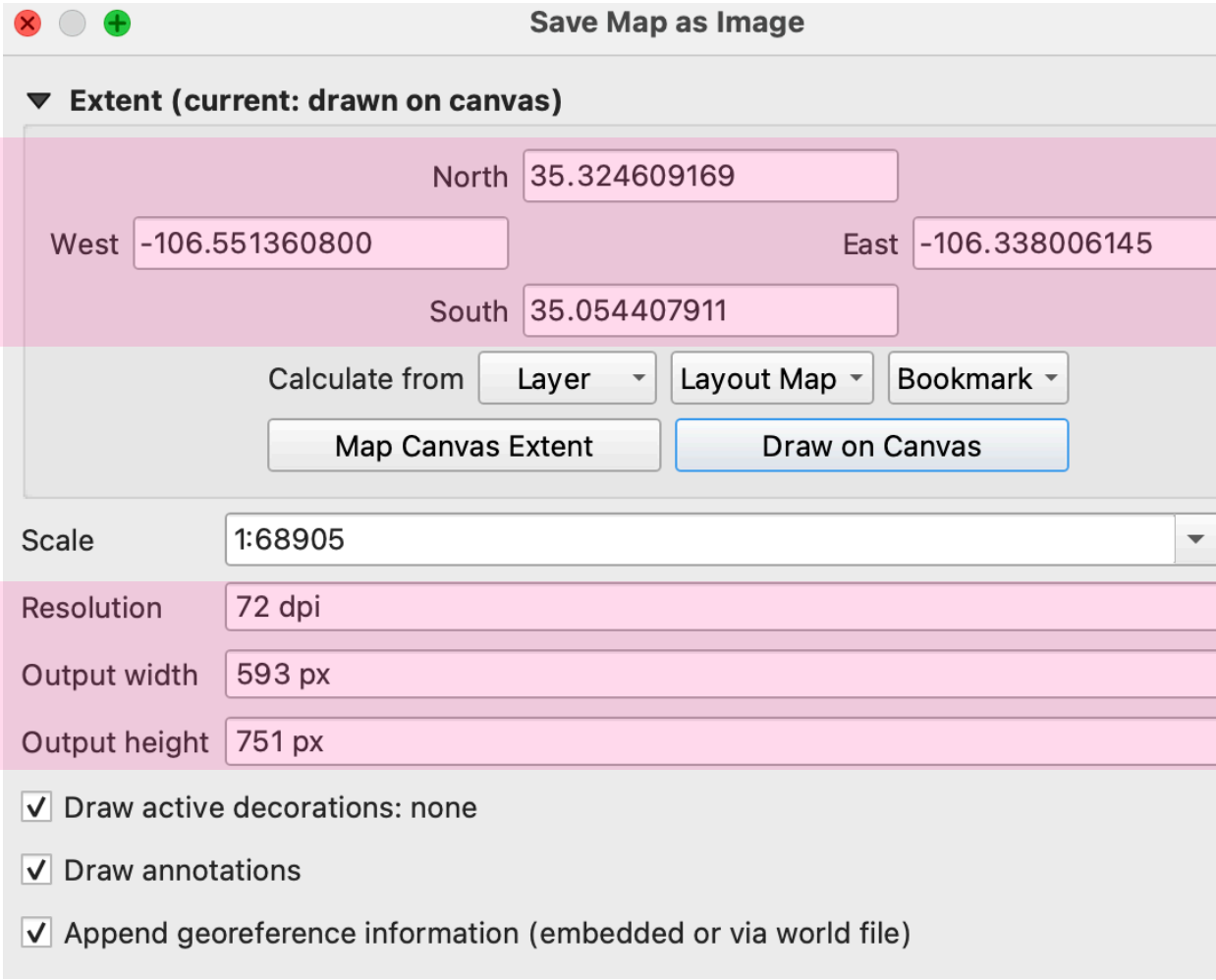


XY Units

- Save the information in this window for unit conversion.
- Convert latitude (NS) and longitude (WE) coordinates to pixels (xy)



https://www.gps-coordinates.net/



Bitmaps in GH and Rhino

What we're going to do

- Create a 2D surface from the bitmap image
- Surface height corresponds to elevation (pixel brightness)
- Surface x and y correspond to distance (# pixels)

What we're going to do

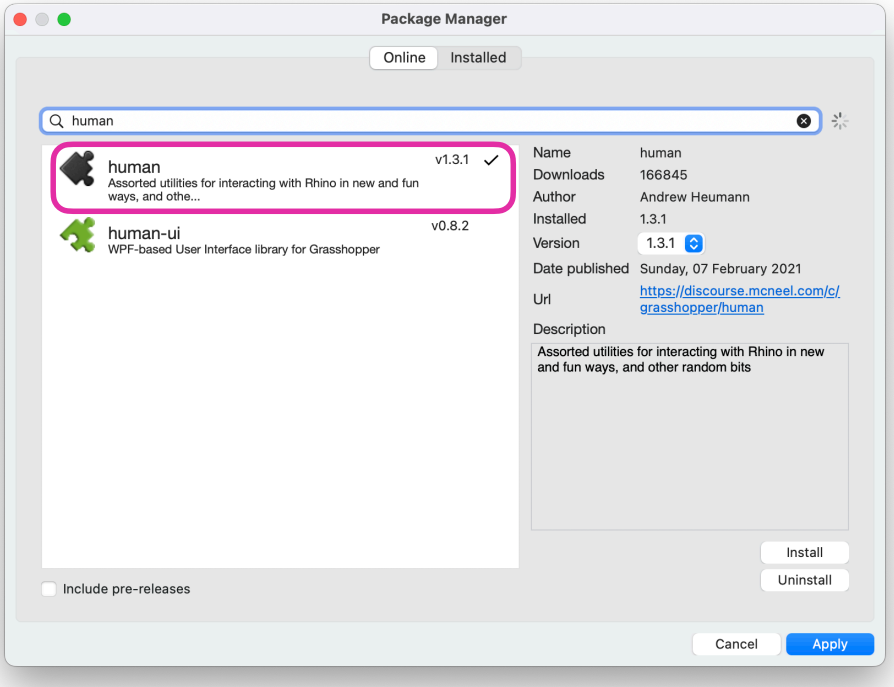
- Download and install Human plugin to work with bitmaps
- Open the bitmap image
- Get the brightness of the pixels across the image
  Choose a sampling rate (ie: every 10 pixels) that won't crash Rhino
- Map brightness to x, y, and z. Create arrays of points from data.
- Create lines from points
- Loft lines to create an elevation surface
- Create a set of bottom surfaces for the elevation surface.
  Need a closed form for 3D printing

Step 1: installing human GH plugin

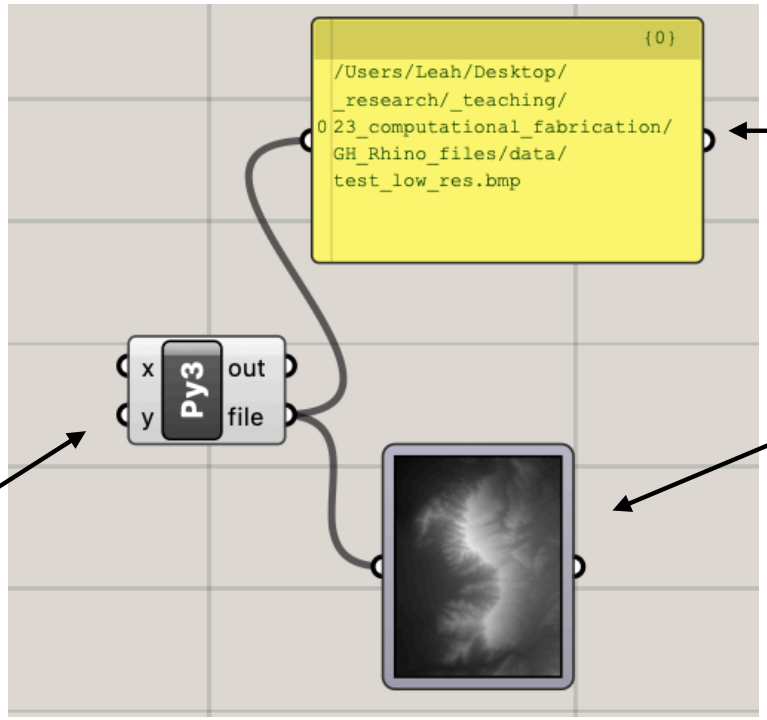
Install the human GH plugin

- **IMPORTANT:** must follow these exact steps to install the correct version of the library. Do not install the Food4Rhino version.
- In Rhino, open the PackageManager by typing "Package Manager" in the command line.
- Choose "human" v1.3.1 in the box that pops up and click Apply.
- Quit and restart Rhino.

https://www.food4rhino.com/en/app/human



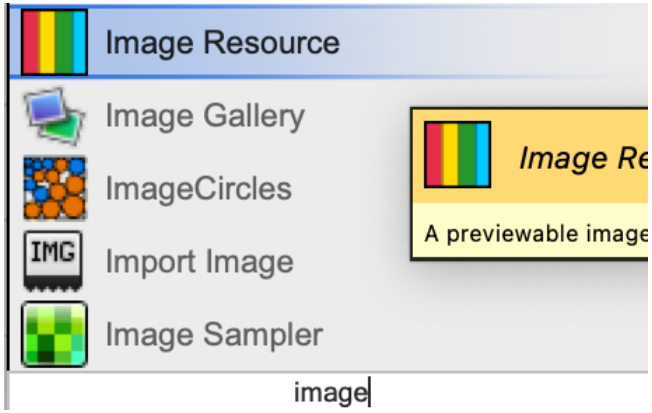
Open the bitmap in GH



panel block
to see path to file

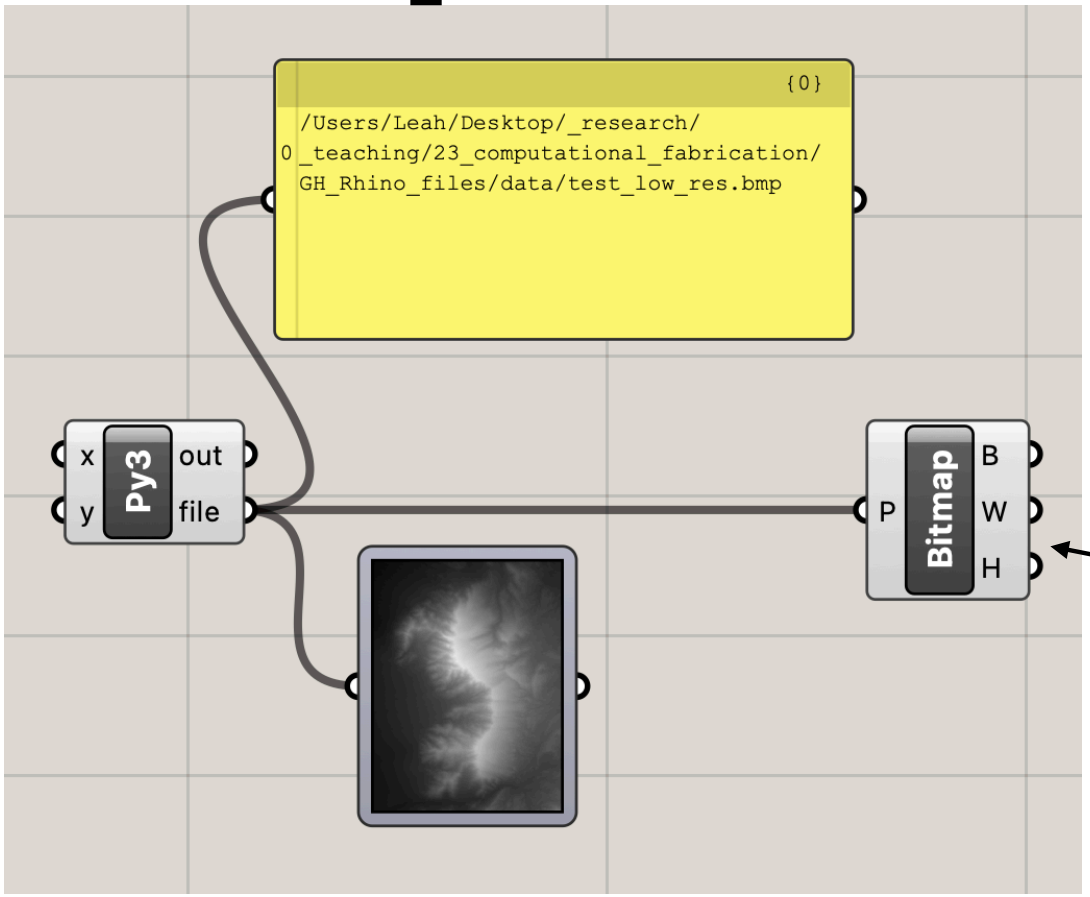
image resource block

open file

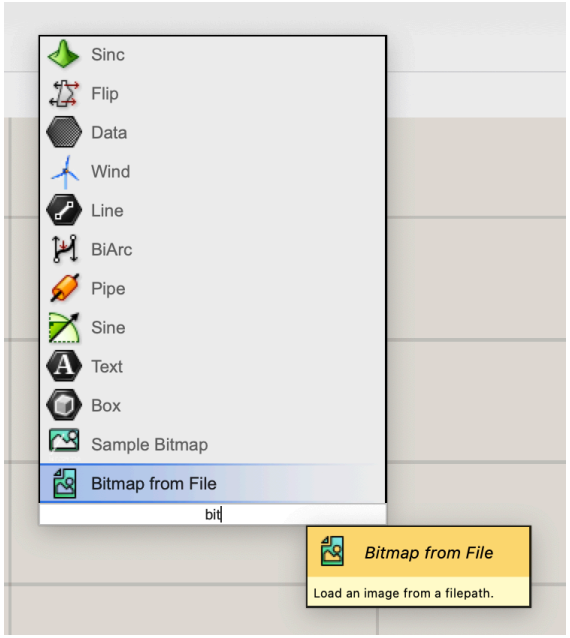



```
1 import rhinoscriptsyntax as rs
2
3 filter = "IMG file (*.bmp)|*.pbm|*.png|All Files (*.*)|*.*||"
4 file = rs.OpenFileName("Open Image File", filter)
```

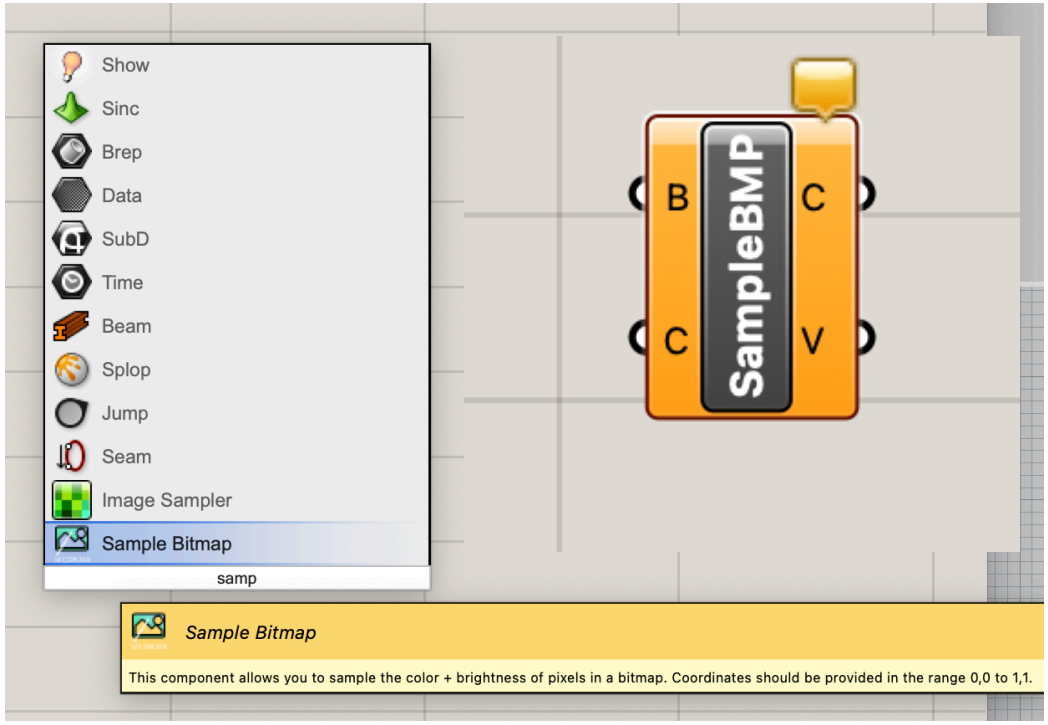
Open the bitmap in GH



Bitmap from File block
(requires human plugin)



We want to get pixel information using Sample Bitmap



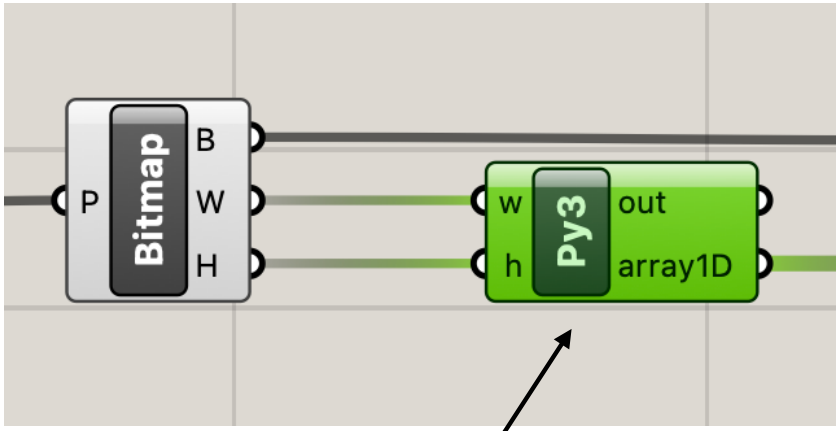
returns pixel color (C) and brightness (V) information

B input = bitmap

C input = 1D array point/pixel index (in range of 0-1)

Need to generate UV array (1D array of points) first.

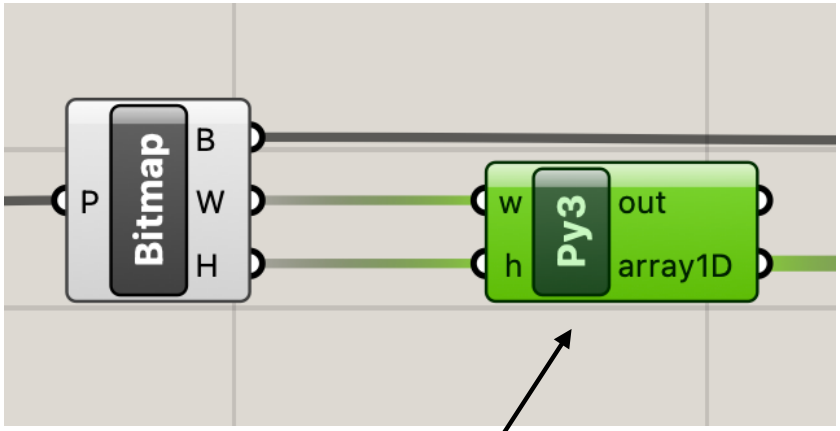
Create point array




```
array1D = []
for i in range (0,w,10):
    row = []
    for j in range (0,h,10):
        row.append(rs.CreatePoint(i,j,0))
        array1D.append(rs.CreatePoint(i/float(w),j/float(h),0))
```

The 1D array creates a list of points in the range 0-1. This will be used to access pixel information.

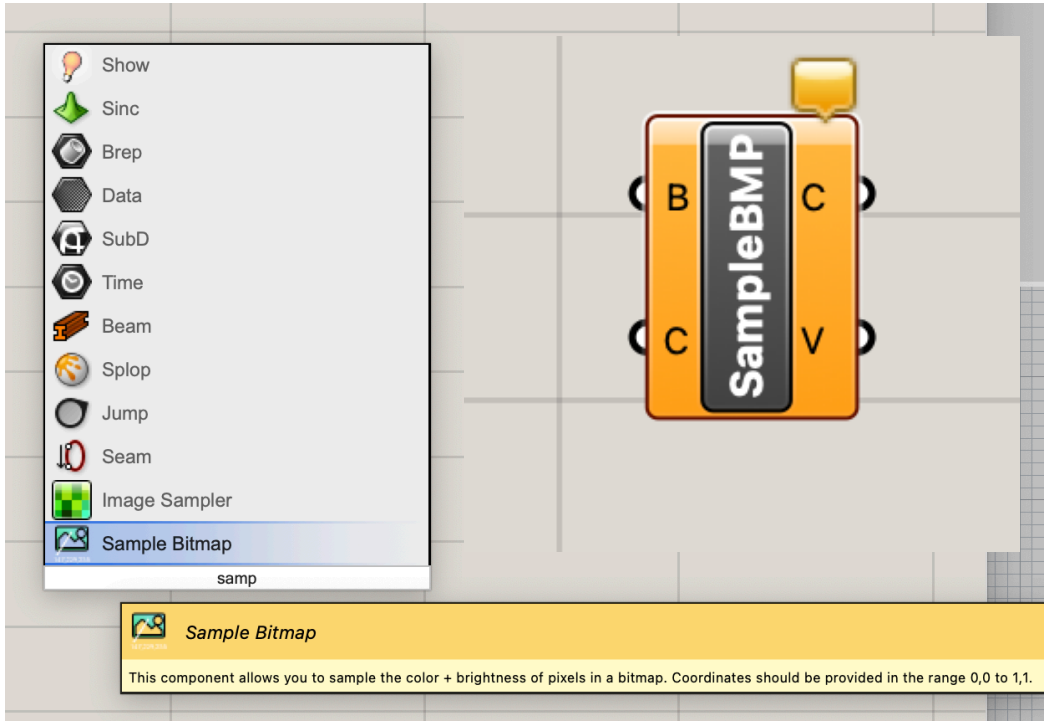
Create point arrays




```
array1D = []
for i in range (0,w,10):
    row = []
    for j in range (0,h,10):
        row.append(rs.CreatePoint(i,j,0))
        array1D.append(rs.CreatePoint(i/float(w),j/float(h),0))
```

Important: **make sure you are incrementing the i and j variables by 10.** Otherwise your program will generate too many points and run very very slowly.

Get pixel information using Sample Bitmap block

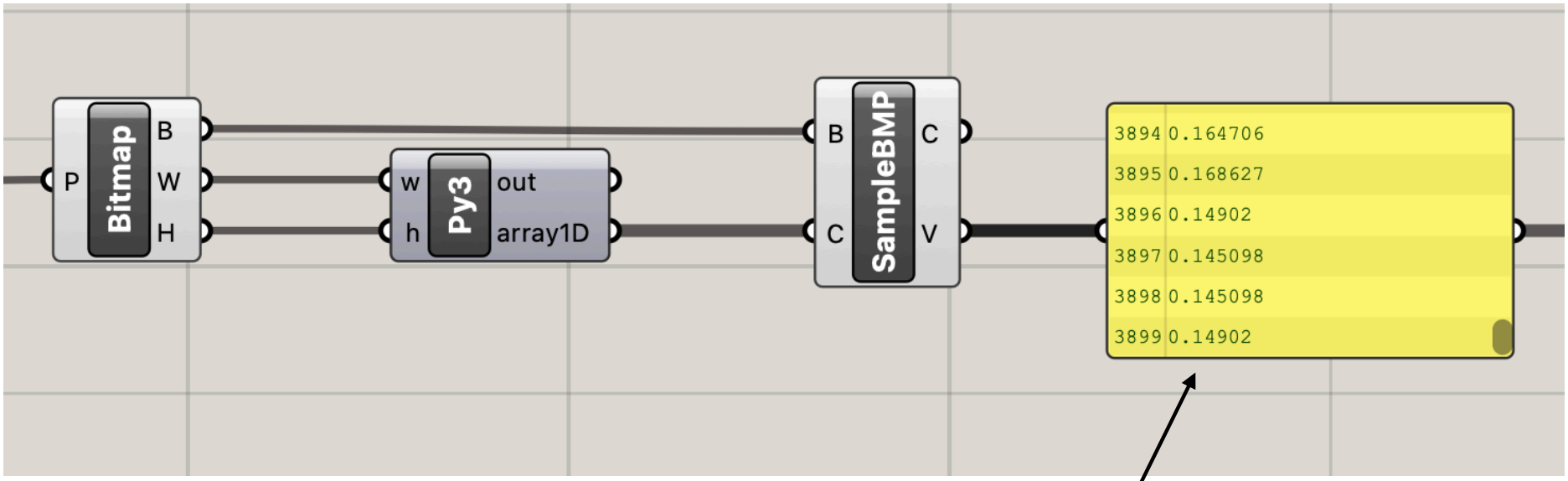


returns pixel color (C) and brightness (V) information

B input = bitmap

C input = 1D array point/pixel index (in range of 0-1)

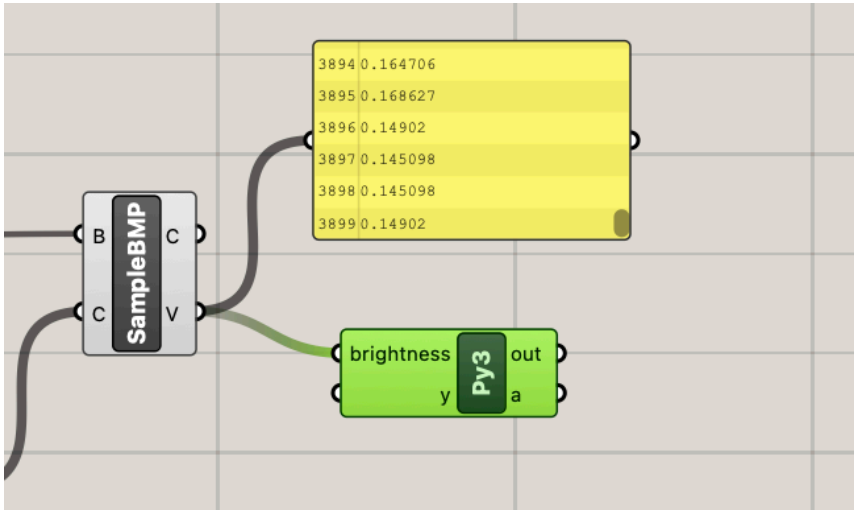
Get pixel information



brightness of each pixel
using a yellow Panel
text box to look at the data.

Generate geometry from pixel info

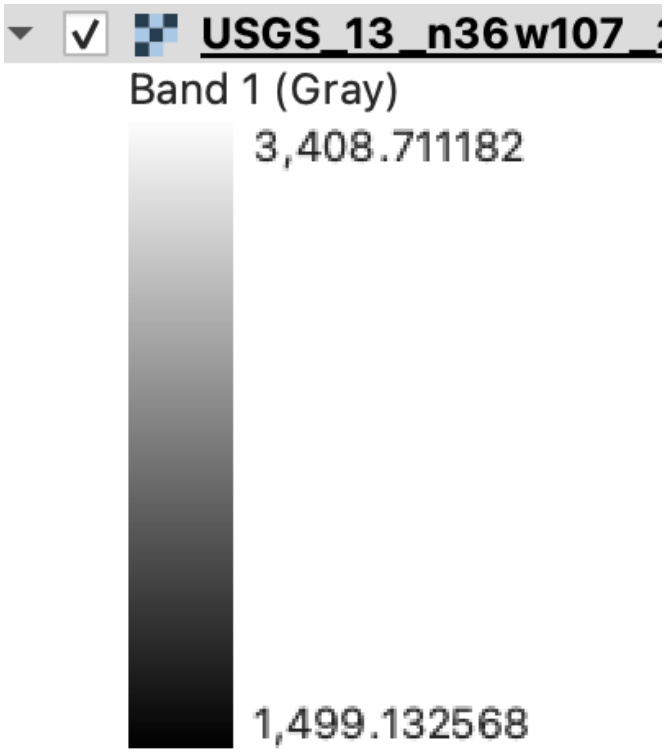
Connect pixel info to new Python block

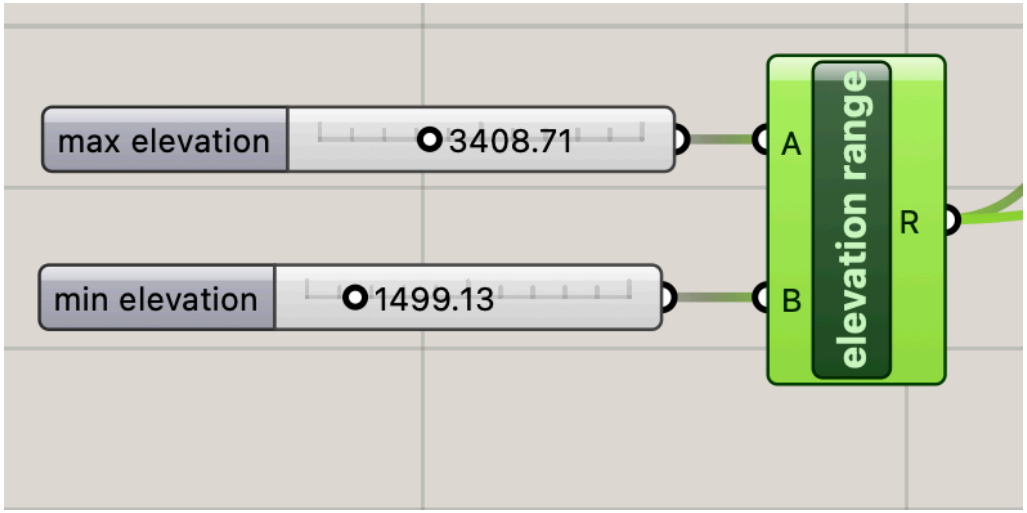


connect brightness info to
new python block
choose list access
Type hint: float

add inputs for w, h (size of image)
elevation_range:
pixel to elevation mapping

Elevation Mapping (z units)





elevation mapping =
subtract min map elevation from max elevation

Create points based on x,y, and elevation

```
curves = []
b = 0
for i in range (0,w,10):
    points = []
    for j in range (0,h,10):
        point = rs.CreatePoint(i/10, j/10, brightness[b]*elevation_range/100)
        points.append(point)
        b = b+1
```

Brightness = z component of new point

```
curves = []
b = 0
for i in range (0,w,10):
    points = []
    for j in range (0,h,10):
        point = rs.CreatePoint(i/10, j/10, brightness[b]*elevation_range/100)
        points.append(point)
        b = b+1
```

z component is multiplied by elevation_range

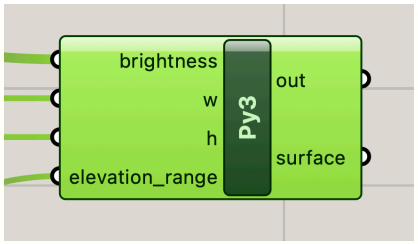
Create curves from points

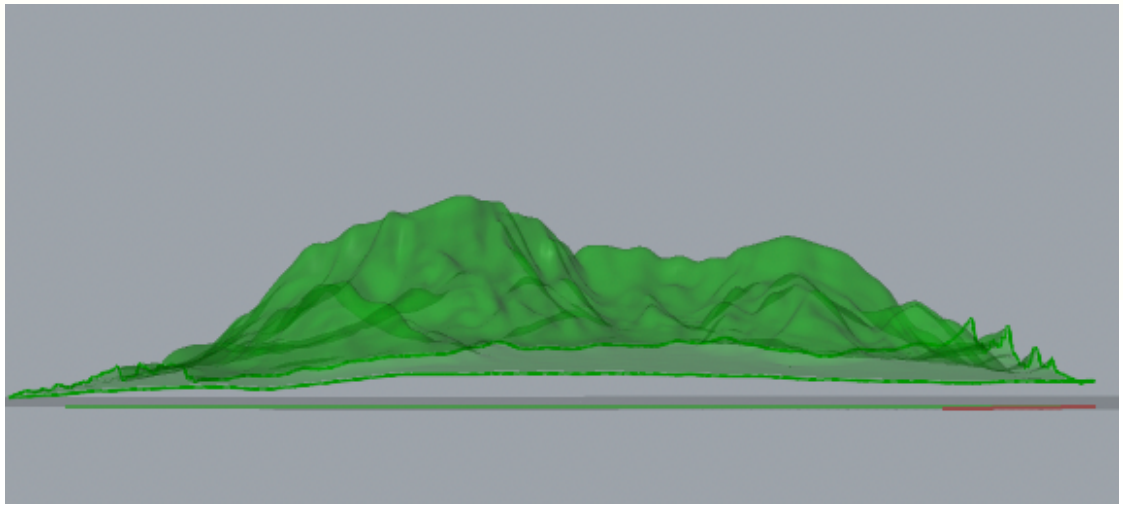
```
curves = []
b = 0
for i in range (0,w,10):
    points = []
    for j in range (0,h,10):
        point = rs.CreatePoint(i/10, j/10, brightness[b]*elevation_range/100)
        points.append(point)
        b = b+1
    curve = rs.AddCurve(points)
    curves.append(curve)
```

Create lofted surface from curves

```
curves = []
b = 0
for i in range (0,w,10):
    points = []
    for j in range (0,h,10):
        point = rs.CreatePoint(i/10, j/10, brightness[b]*elevation_range/100)
        points.append(point)
        b = b+1
    curve = rs.AddCurve(points)
    curves.append(curve)

surface = rs.AddLoftSrf(curves)
```





questions?

What we're going to do

- Download and install Human plugin to work with bitmaps
- Open the bitmap image
- Get the brightness of the pixels across the image
  Choose a sampling rate (ie: every 10 pixels) that won't crash Rhino
- Map brightness to x, y, and z. Create arrays of points from data.
- Create lines from points
- Loft lines to create an elevation surface
- **Create a set of bottom surfaces for the elevation surface.**
  Need a closed form for 3D printing

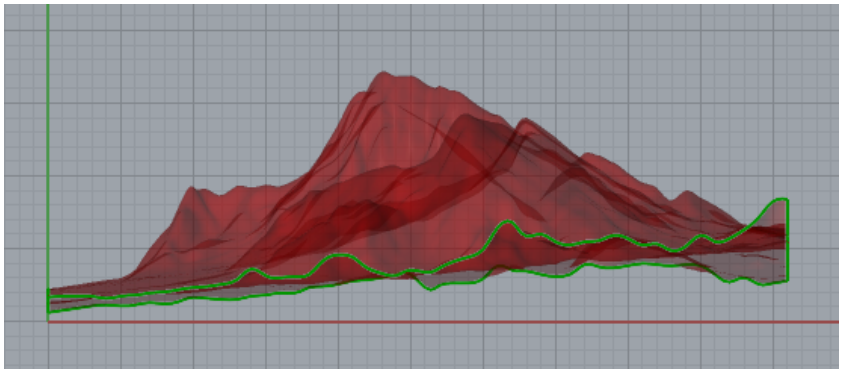
Steps to create a bottom

- Find the four edge curves of the surface
- Create four new surfaces for each edge that go from edge curve to xy plane.
- Create a bottom surface on xy plane
- Weld surfaces together to create a solid

Get edge curves of surface

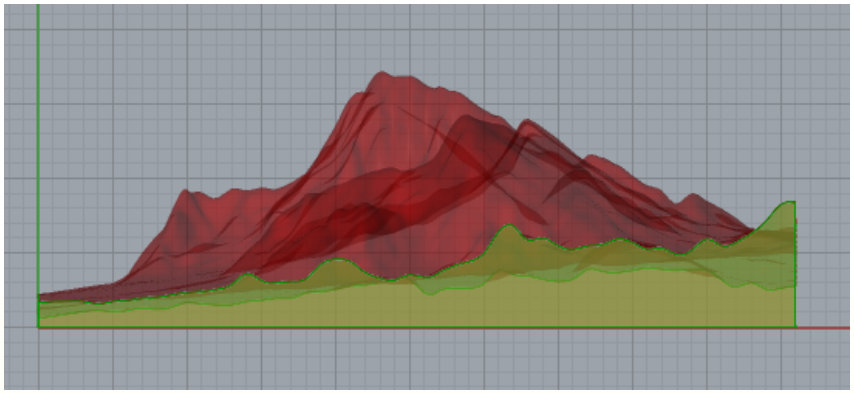
- Find the four edge curves of the surface

```
# create a bottom for the surface
# find edges of surface
edges = rs.DuplicateEdgeCurves(surface)
```



Create edge surfaces

- For each edge curve,
  create three lines that define a surface
  from edge curve to z=0



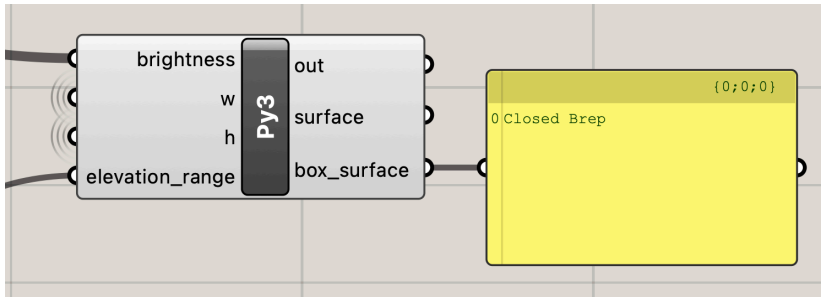
```
# edge surfaces
edge_surfaces = []
for i in range (len(edges)):
    startPoint = rs.CurveStartPoint(edges[i])
    endPoint=rs.CurveEndPoint(edges[i])
    curve1 = rs.AddLine([startPoint.X,startPoint.Y,0],[startPoint.X,startPoint.Y,startPoint.Z])
    curve2 = edges[i]
    curve3 = rs.AddLine([endPoint.X,endPoint.Y,endPoint.Z],[endPoint.X,endPoint.Y,0])
    curve4 = rs.AddLine([endPoint.X,endPoint.Y,0],[startPoint.X,startPoint.Y,0])
    edge_surface = rs.AddEdgeSrf([curve1, curve2, curve3, curve4])
    edge_surfaces.append(edge_surface)
```

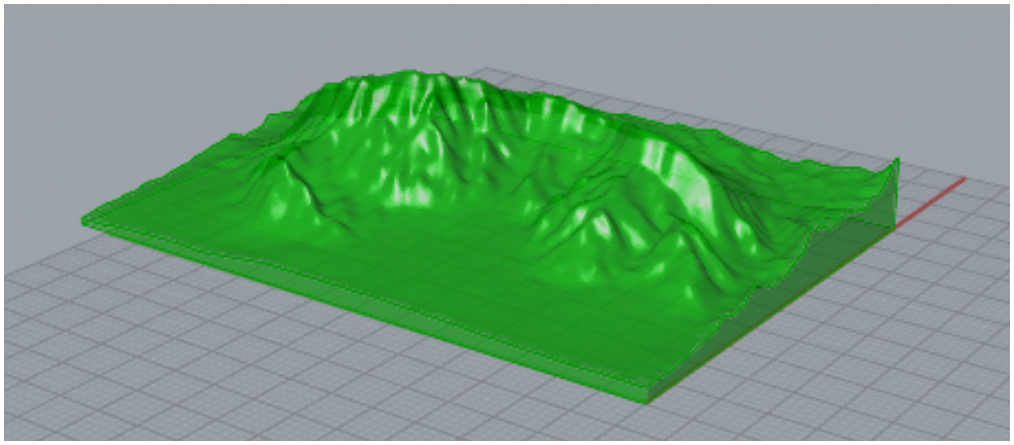
Create a solid

- Join all edge surfaces
- Join edge surfaces to main surface
- Cap this joined surface to create a solid

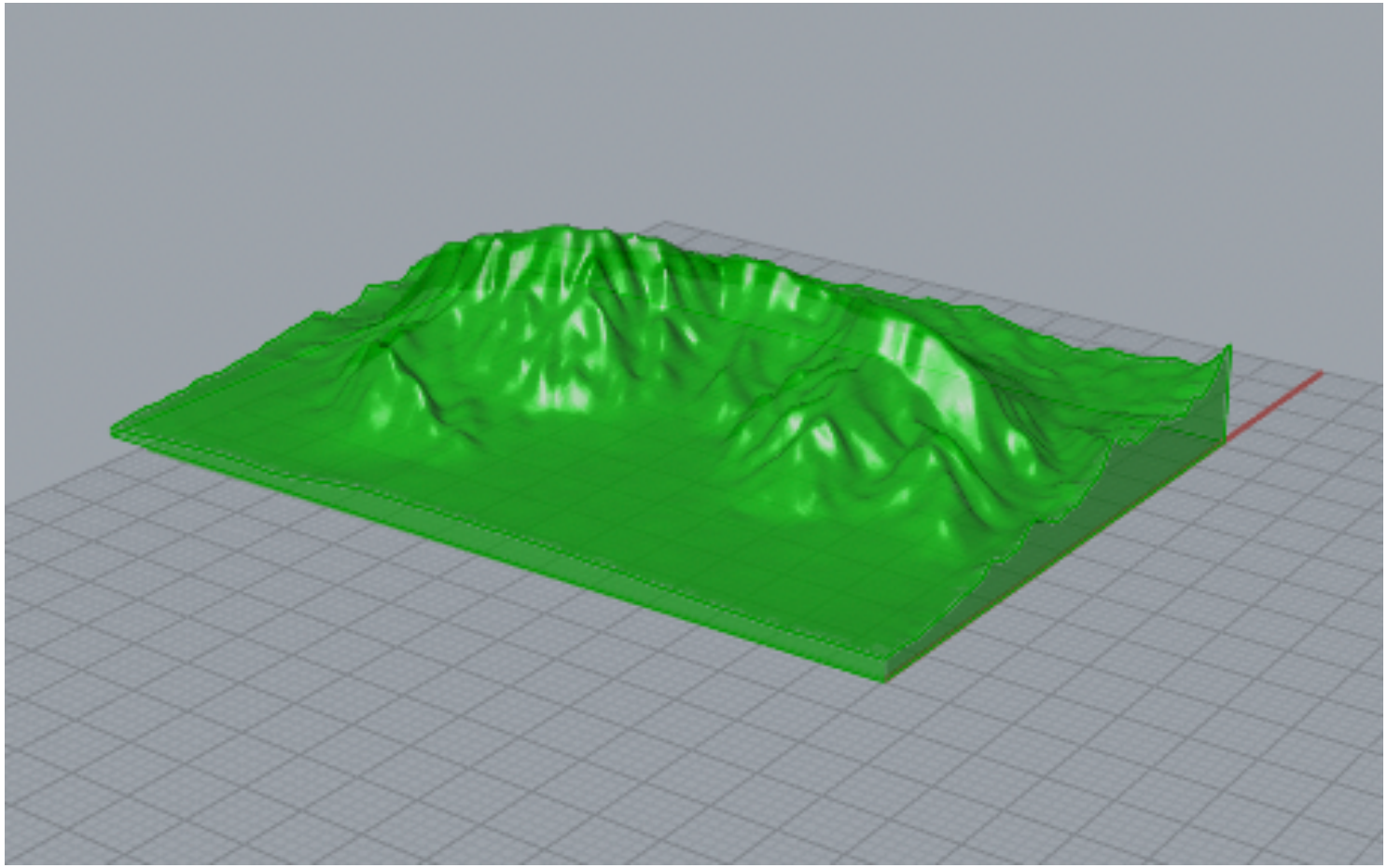
```
edge_surface = rs.JoinSurfaces(edge_surfaces)

box_surface = rs.JoinSurfaces([surface,edge_surface])
rs.CapPlanarHoles(box_surface)
```



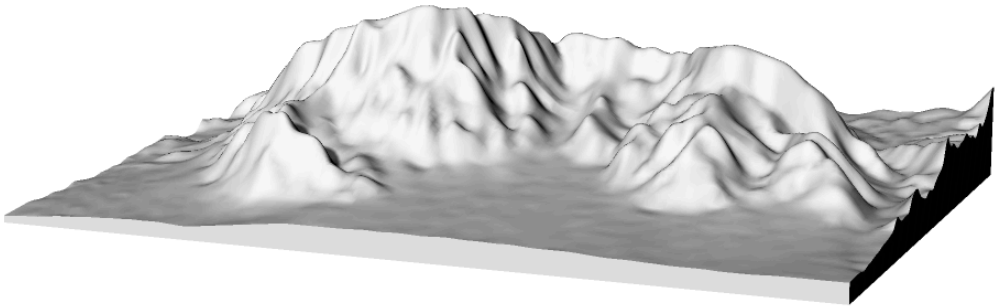


Create a solid

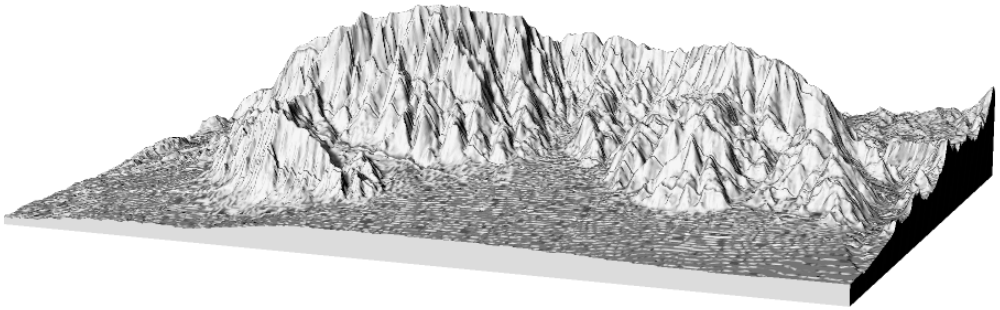


Create a higher-res version

- Change sampling rate



sampling every 10 pixels



sampling every pixel

Units

- Make sure you are able to accurately label x, y, and z axes with units
- Go back to export information and find x and y units in meters

QGIS & other map data

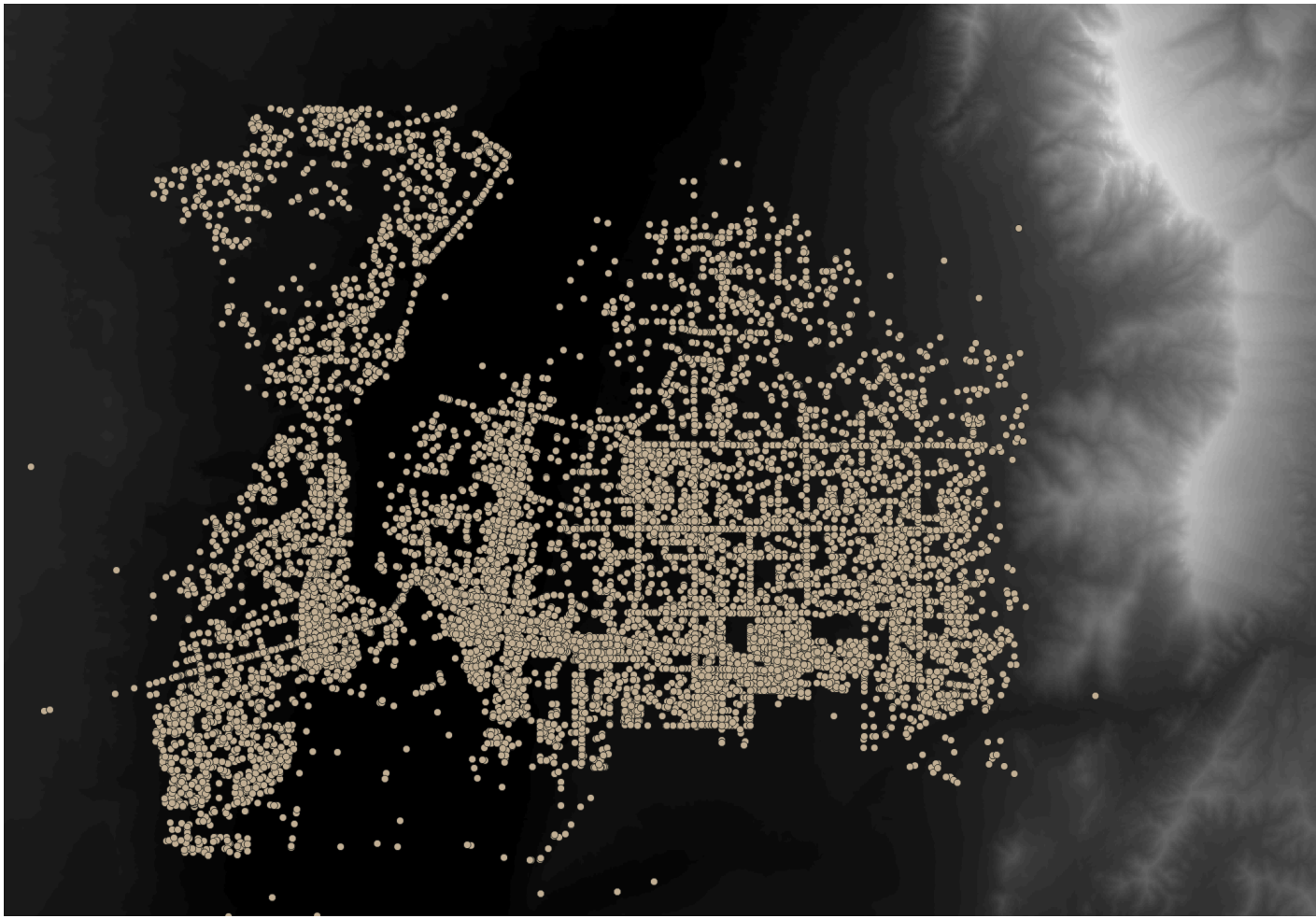
2023 ABQ Crime Incidents

File downloaded from city of ABQ public data repository
https://www.cabq.gov/abq-data/


```
"features": [
 {
  "attributes": {
   "OBJECTID": 61868299,
   "BlockAddress": "2000 BLOCK SOUTH PLAZA ST NW",
   "IncidentType": "ONSITE SUSPICIOU",
   "ReportDateTime": "2023-04-28 00:04:26"
  },
  "geometry": {
   "x": -11874426.0821,
   "y": 4176919.0609000027
  }
 },
 {
  "attributes": {
   "OBJECTID": 61868300,
   "BlockAddress": "COCHITI RD SE / LOUISIANA BL SE",
   "IncidentType": "ONSITE AUTO THEF",
   "ReportDateTime": "2023-04-28 00:15:27"
  },
  "geometry": {
   "x": -11863157.442699999,
   "y": 4174052.2463999987
  }
 },
 {
  "attributes": {
   "OBJECTID": 61868301,
   "BlockAddress": "6TH ST NW / CENTRAL AV NW",
   "IncidentType": "ONSITE SUSPICIOU",
   "ReportDateTime": "2023-04-28 00:20:17"
```

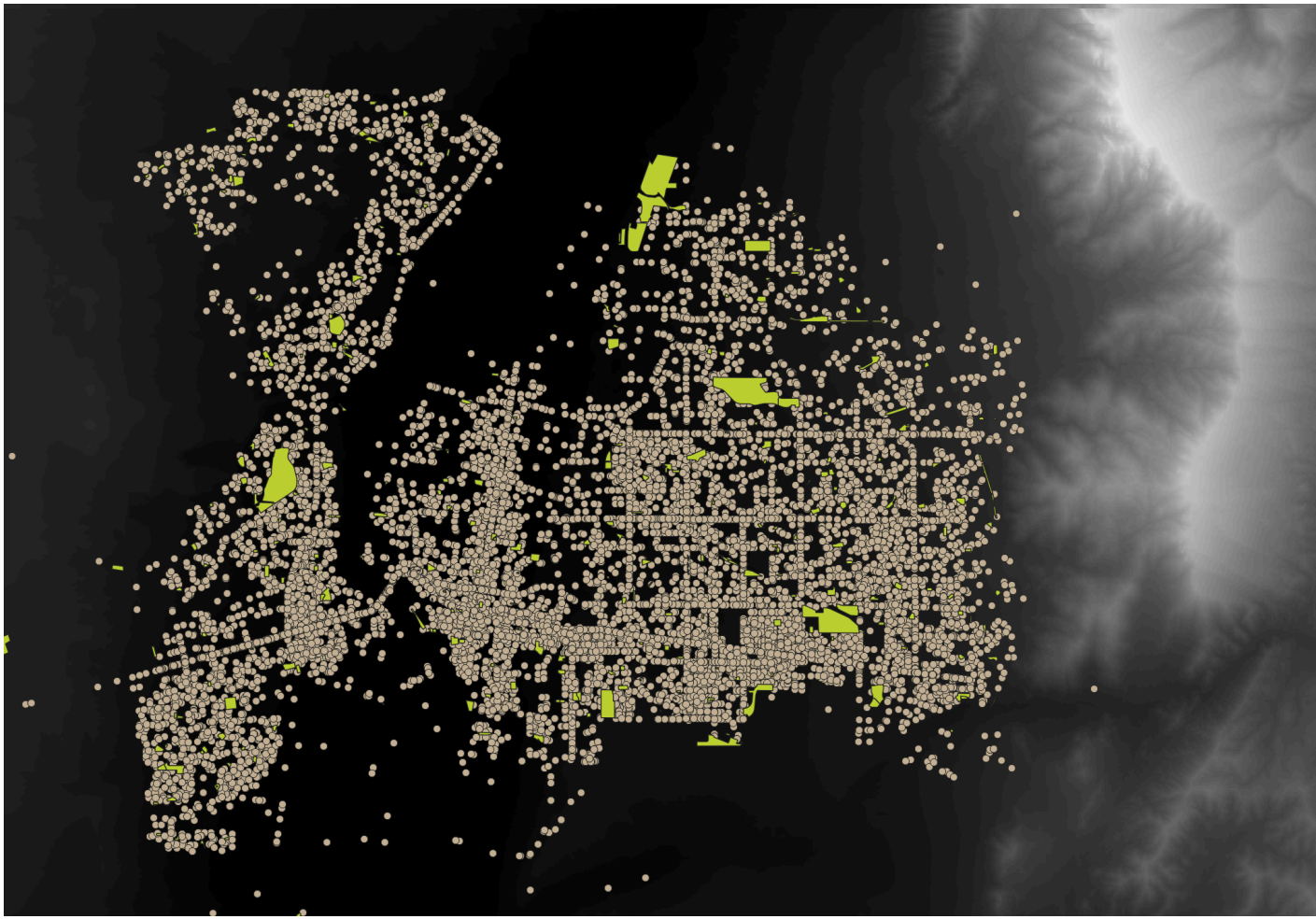
2023 ABQ Crime Incidents

- JSON file opens in QGIS
- Marks all incidents on our topo map



Add ABQ Parks...

- Tons of geographic data you can make use of with these tools!.
- Park data also from city of ABQ: https://www.cabq.gov/abq-data/



Thank you!

CS 491 and 591
Professor: Leah Buechley
https://handandmachine.cs.unm.edu/classes/Computational_Fabrication_Spring2021/