# PlantCV 對高光譜影像的支援

## *Hyperspectral Image*

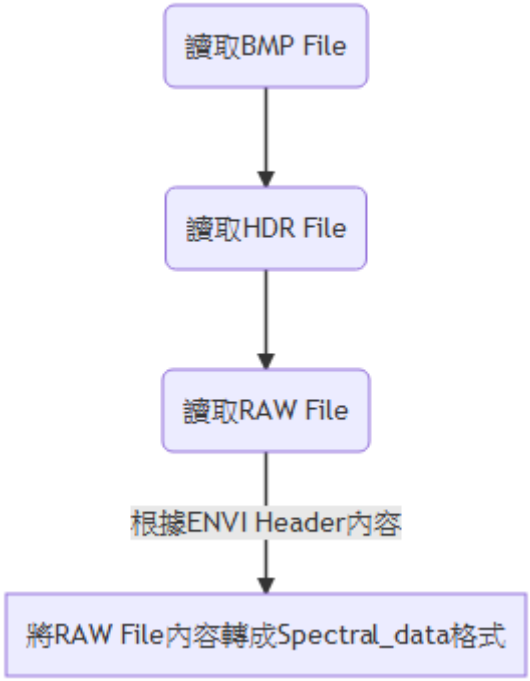
PlantCV 支援ENVI格式
ENVI格式重要欄位說明：

- Columns: The number of columns in the image
- Rows: The number of rows in the image
- Bands: The number of bands in the image
- Interleave: The data interleave (BSQ, BIL, or BIP)
- Data Type: The IDL data type of the image:
- Byte: 8-bit unsigned integer
- Long Integer: 32-bit signed integer
- Float: 32-bit single-precision
- Double: 64-bit double-precision floating-point
- Complex: Real-imaginary pair of single-precision floating-point
- Double-precision Complex: Real-imaginary pair of double precision floating-point
- Unsigned Integer (default): 16-bit
- Unsigned Long Integer: 32-bit
- 64-bit Long Integer
- 64-bit Unsigned Long Integer
- File Type: The file format of the image
- Header Offset: The number of bytes of embedded header information present in the file. ENVI skips these bytes when reading the file.

## *Module Hyperspectral of PlantCV 讀取Hyperspectral Image*

高光譜相機會產生三個主要的檔案:

- BMP File: 彩色相機的RGB影像
- HDR File: ENVI Header File, 描述ENVI高光譜影像內容
- RAW File：實際高光譜影像



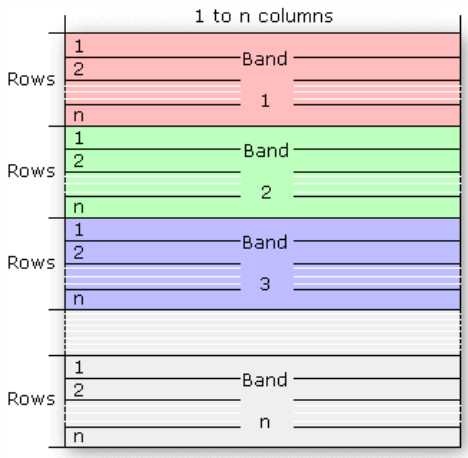

## HDR File Content

```
ENVI
samples = 1392
lines = 1541
bands = 256
header offset = 0
file type = ENVI Standard
data type = 12
interleave = bil
sensor type = Unknown
wavelength units = nm
binning = {1,1}
wavelength = {381.50,383.70,386.00,388.30,390.50,392.80,395.10,397.30,399.60,401.90,404.10,406.40,408.70,411.00,413.30,415.50,417.80,420.10,
422.40,424.70,427.00,429.30,431.60,433.90,436.20,438.50,440.80,443.10,445.40,447.70,450.00,452.30,454.60,456.90,459.30,461.60,463.90,466.20,
468.50,470.90,473.20,475.50,477.90,480.20,482.50,484.90,487.20,489.50,491.90,494.20,496.60,498.90,501.30,503.60,506.00,508.30,510.70,513.00,
515.40,517.80,520.10,522.50,524.90,527.20,529.60,532.00,534.30,536.70,539.10,541.50,543.90,546.20,548.60,551.00,553.40,555.80,558.20,560.60,
563.00,565.40,567.80,570.20,572.60,575.00,577.40,579.80,582.20,584.60,587.00,589.40,591.80,594.30,596.70,599.10,601.50,604.00,606.40,608.80,
611.20,613.70,616.10,618.50,621.00,623.40,625.90,628.30,630.80,633.20,635.60,638.10,640.60,643.00,645.50,647.90,650.40,652.80,655.30,657.80,
660.20,662.70,665.20,667.60,670.10,672.60,675.10,677.50,680.00,682.50,685.00,687.50,690.00,692.50,694.90,697.40,699.90,702.40,704.90,707.40,
709.90,712.40,714.90,717.40,720.00,722.50,725.00,727.50,730.00,732.50,735.00,737.60,740.10,742.60,745.10,747.70,750.20,752.70,755.30,757.80,
760.30,762.90,765.40,768.00,770.50,773.10,775.60,778.20,780.70,783.30,785.80,788.40,790.90,793.50,796.10,798.60,801.20,803.80,806.30,808.90,
811.50,814.00,816.60,819.20,821.80,824.40,826.90,829.50,832.10,834.70,837.30,839.90,842.50,845.10,847.70,850.30,852.90,855.50,858.10,860.70,
863.30,865.90,868.50,871.10,873.80,876.40,879.00,881.60,884.20,886.90,889.50,892.10,894.70,897.40,900.00,902.60,905.30,907.90,910.60,913.20,
915.80,918.50,921.10,923.80,926.40,929.10,931.70,934.40,937.10,939.70,942.40,945.00,947.70,950.40,953.00,955.70,958.40,961.10,963.70,966.40,
969.10,971.80,974.50,977.20,979.80,982.50,985.20,987.90,990.60,993.30,996.00,998.70,1001.40,1004.10,1006.80,1009.50,1012.20,1014.90}
```

## interleave格式為BIP

See an example of a BIP file

Band sequential format stores information for the image one band at a time. In other words, data for all the pixels for band 1 is stored first, then data for all pixels for band 2, and so on.



## Raw File To 高光譜影像

```
raw_data = np.fromfile('newrawfile20191207155328.raw', dtype=np.uint16, -1)

array_data = raw_data.reshape(int(header_dict["lines"]),
                                  int(header_dict["bands"]),
                                  int(header_dict["samples"])).transpose((0, 2, 1))

spectral_array = Spectral_data(array_data=array_data,
                                   max_wavelength=float(str(header_dict["wavelength"]
[-1]).rstrip()),
                                   min_wavelength=float(str(header_dict["wavelength"]
[0]).rstrip()),
                                   d_type=header_dict["data type"],
                                   wavelength_dict=wavelength_dict,
samples=int(header_dict["samples"]),
                                   lines=int(header_dict["lines"]),
interleave=header_dict["interleave"],
                                   wavelength_units=header_dict["wavelength units"],
array_type="datacube",
                                   pseudo_rgb=None, filename=filename,
default_bands=default_bands)

# Make pseudo-rgb image and replace it inside the class instance object
pseudo_rgb = _make_pseudo_rgb(spectral_array)
```

### spectral_array 為PlantCV中代表高光譜影像的數據

## Pseudo高光譜影像

## 高光譜影像中數值代表的波長

{381.5: 0.0, 383.7: 1.0, 386.0: 2.0, 388.3: 3.0, 390.5: 4.0, 392.8: 5.0, 395.1: 6.0, 397.3: 7.0, 399.6: 8.0, 401.9: 9.0, 404.1: 10.0, 406.4: 11.0, 408.7: 12.0, 411.0: 13.0, 413.3: 14.0, 415.5: 15.0, 417.8: 16.0, 420.1: 17.0, 422.4: 18.0, 424.7: 19.0, 427.0: 20.0, 429.3: 21.0, 431.6: 22.0, 433.9: 23.0, 436.2: 24.0, 438.5: 25.0, 440.8: 26.0, 443.1: 27.0, 445.4: 28.0, 447.7: 29.0, 450.0: 30.0, 452.3: 31.0, 454.6: 32.0, 456.9: 33.0, 459.3: 34.0, 461.6: 35.0, 463.9: 36.0, 466.2: 37.0, 468.5: 38.0, 470.9: 39.0, 473.2: 40.0, 475.5: 41.0, 477.9: 42.0, 480.2: 43.0, 482.5: 44.0, 484.9: 45.0, 487.2: 46.0, 489.5: 47.0, 491.9: 48.0, 494.2: 49.0, 496.6: 50.0, 498.9: 51.0, 501.3: 52.0, 503.6: 53.0, 506.0: 54.0, 508.3: 55.0, 510.7: 56.0, 513.0: 57.0, 515.4: 58.0, 517.8: 59.0, 520.1: 60.0, 522.5: 61.0, 524.9: 62.0, 527.2: 63.0, 529.6: 64.0, 532.0: 65.0, 534.3: 66.0, 536.7: 67.0, 539.1: 68.0, 541.5: 69.0, 543.9: 70.0, 546.2: 71.0, 548.6: 72.0, 551.0: 73.0, 553.4: 74.0, 555.8: 75.0, 558.2: 76.0, 560.6: 77.0, 563.0: 78.0, 565.4: 79.0, 567.8: 80.0, 570.2: 81.0, 572.6: 82.0, 575.0: 83.0, 577.4: 84.0, 579.8: 85.0, 582.2: 86.0, 584.6: 87.0, 587.0: 88.0, 589.4: 89.0, 591.8: 90.0, 594.3: 91.0, 596.7: 92.0, 599.1: 93.0, 601.5: 94.0, 604.0: 95.0, 606.4: 96.0, 608.8: 97.0, 611.2: 98.0, 613.7: 99.0, 616.1: 100.0, 618.5: 101.0, 621.0: 102.0, 623.4: 103.0, 625.9: 104.0, 628.3: 105.0, 630.8: 106.0, 633.2: 107.0, 635.6: 108.0, 638.1: 109.0, 640.6: 110.0, 643.0: 111.0, 645.5: 112.0, 647.9: 113.0, 650.4: 114.0, 652.8: 115.0, 655.3: 116.0, 657.8: 117.0, 660.2: 118.0, 662.7: 119.0, 665.2: 120.0, 667.6: 121.0, 670.1: 122.0, 672.6: 123.0, 675.1: 124.0, 677.5: 125.0, 680.0: 126.0, 682.5: 127.0, 685.0: 128.0, 687.5: 129.0, 690.0: 130.0, 692.5: 131.0, 694.9: 132.0, 697.4: 133.0, 699.9: 134.0, 702.4: 135.0, 704.9: 136.0, 707.4: 137.0, 709.9: 138.0, 712.4: 139.0, 714.9: 140.0, 717.4: 141.0, 720.0: 142.0, 722.5: 143.0, 725.0: 144.0, 727.5: 145.0, 730.0: 146.0, 732.5: 147.0, 735.0: 148.0, 737.6: 149.0, 740.1: 150.0, 742.6: 151.0, 745.1: 152.0, 747.7: 153.0, 750.2: 154.0, 752.7: 155.0, 755.3: 156.0, 757.8: 157.0, 760.3: 158.0, 762.9: 159.0, 765.4: 160.0, 768.0: 161.0, 770.5: 162.0, 773.1: 163.0, 775.6: 164.0, 778.2: 165.0, 780.7: 166.0, 783.3: 167.0, 785.8: 168.0, 788.4: 169.0, 790.9: 170.0, 793.5: 171.0, 796.1: 172.0, 798.6: 173.0, 801.2: 174.0, 803.8: 175.0, 806.3: 176.0, 808.9: 177.0, 811.5: 178.0, 814.0: 179.0, 816.6: 180.0, 819.2: 181.0, 821.8: 182.0, 824.4: 183.0, 826.9: 184.0, 829.5: 185.0, 832.1: 186.0, 834.7: 187.0, 837.3: 188.0, 839.9: 189.0, 842.5: 190.0, 845.1: 191.0, 847.7: 192.0, 850.3: 193.0, 852.9: 194.0, 855.5: 195.0, 858.1: 196.0, 860.7: 197.0, 863.3: 198.0, 865.9: 199.0, 868.5: 200.0, 871.1: 201.0, 873.8: 202.0, 876.4: 203.0, 879.0: 204.0, 881.6: 205.0, 884.2: 206.0, 886.9: 207.0, 889.5: 208.0, 892.1: 209.0, 894.7: 210.0, 897.4: 211.0, 900.0: 212.0, 902.6: 213.0, 905.3: 214.0, 907.9: 215.0, 910.6: 216.0, 913.2: 217.0, 915.8: 218.0, 918.5: 219.0, 921.1: 220.0, 923.8: 221.0, 926.4: 222.0, 929.1: 223.0, 931.7: 224.0, 934.4: 225.0, 937.1: 226.0, 939.7: 227.0, 942.4: 228.0, 945.0: 229.0, 947.7: 230.0, 950.4: 231.0, 953.0: 232.0, 955.7: 233.0, 958.4: 234.0, 961.1: 235.0, 963.7: 236.0, 966.4: 237.0, 969.1: 238.0, 971.8: 239.0, 974.5: 240.0, 977.2: 241.0, 979.8: 242.0, 982.5: 243.0, 985.2: 244.0, 987.9: 245.0, 990.6: 246.0, 993.3: 247.0, 996.0: 248.0, 998.7: 249.0, 1001.4: 250.0, 1004.1: 251.0, 1006.8: 252.0, 1009.5: 253.0, 1012.2: 254.0, 1014.9: 255.0}

## 歸一化植被指數（Normalized Difference Vegetation Index；NDVI）

> NDVI通常是用衛星遙感數據計算，以評估目標地區綠色植被的生長狀況。

> 計算方式是利用紅光與近紅外光的反射，能顯示出植物生長、生態系的活力與生產力等資訊。數值愈大表示植物生長愈多。公式如下：

$$NDVI = \frac{(1+L)\times(\rho_{NIR}-\rho_{RED})}{\rho_{NIR}+\rho_{RED+L}}$$

> NIR為近紅外光反射；RED為紅光反射，NDVI之值介於-1到1之間。 當RED=0時，有最大值1；反之，當NIR=0時，有最小值-1。

## 土壤調整植生指數 (Soil-adjusted vegetation index; SAVI)

> 公式如下：

$$SAVI = \frac{\rho_{NIR}-\rho_{RED}}{\rho_{NIR}+\rho_{RED}}$$

> 其中L是樹冠背景調整因子。發現反射空間中的L值為0.5，可最大程度地減少土壤亮度變化並消除對不同土壤進行額外校準的需要。發現該轉變幾乎消除了土壤引起的植被指數變化

## 綠色差值值被指數 (Green Difference Vegetation Index; GDVI)

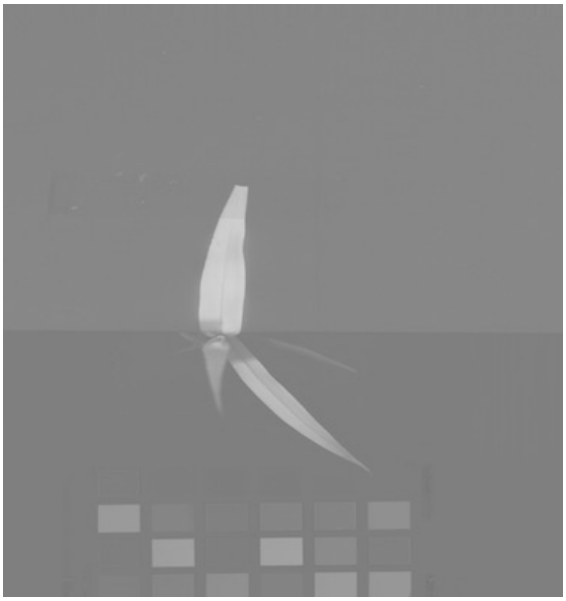
> 該指數最初是用來以彩色與紅外影像車預測谷物的氮含量，可以更好反映荒漠低植區植被覆蓋狀況。

> 公式如下：

$$GDVI = \frac{{\rho_{NIR}}^2-{\rho_{Red}}^2}{{\rho_{NIR}}^2+{\rho_{Red}}^2}$$

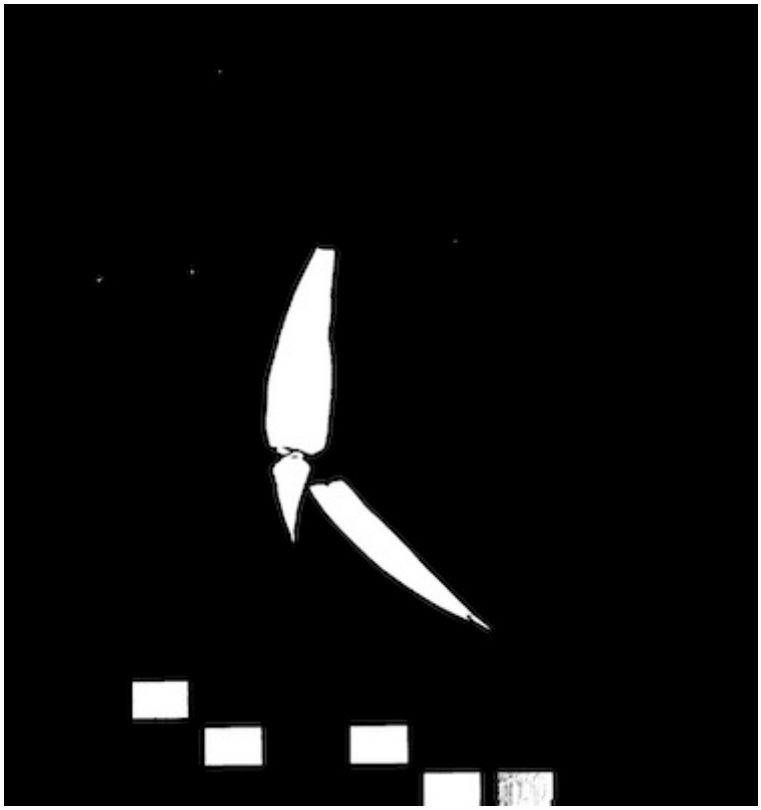
```
# Extract the Green Difference Vegetation Index from the datacube

# Inputs:
#   array        - Hyperspectral data instance
#   index        - Index of interest
#   distance     - How lenient to be if the required wavelengths
#                  for a specific index are not available
index_array_gdvi = pcv.hyperspectral.extract_index(array=spectral_array,
                                                   index="GDVI",
                                                   distance=20)
```

| Pseudo RGB Image | GDVI Image |
| --- | --- |

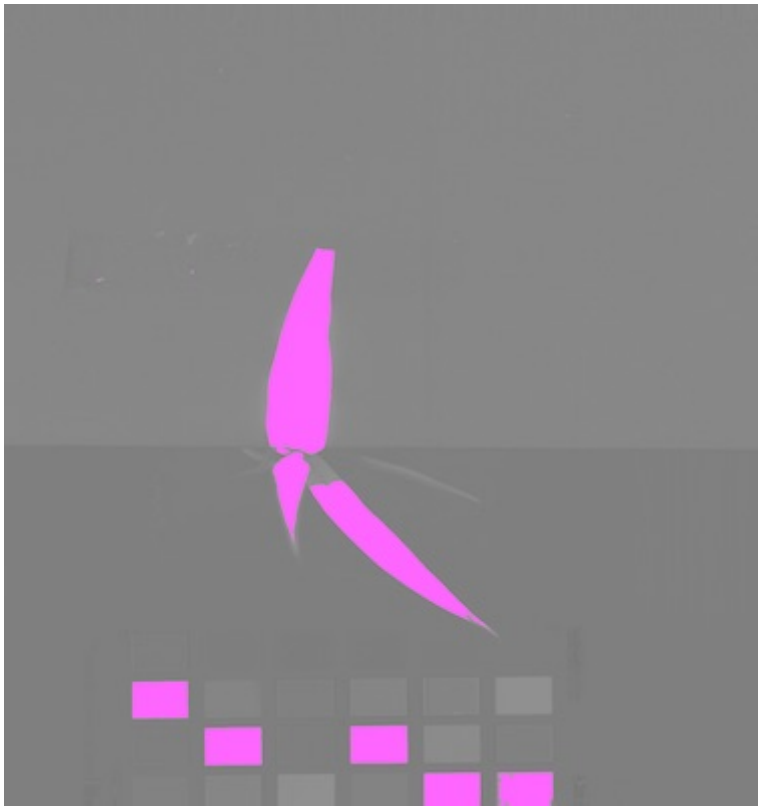


## 針對GDVI影像進行二值化運算

```
# Threshold the grayscale image

# Inputs:
#   gray_img    - Grayscale image data
#   threshold   - Threshold value (between 0-255)
#   max_value   - Value to apply above threshold (255 = white)
#   object_type - 'light' (default) or 'dark'. If the object is lighter than the
#                 background then standard threshold is done. If the object is
#                 darker than the background then inverse thresholding is done.
gdvi_thresh = pcv.threshold.binary(gray_img=index_array_gdvi.array_data,
threshold=150, max_value=255)
```

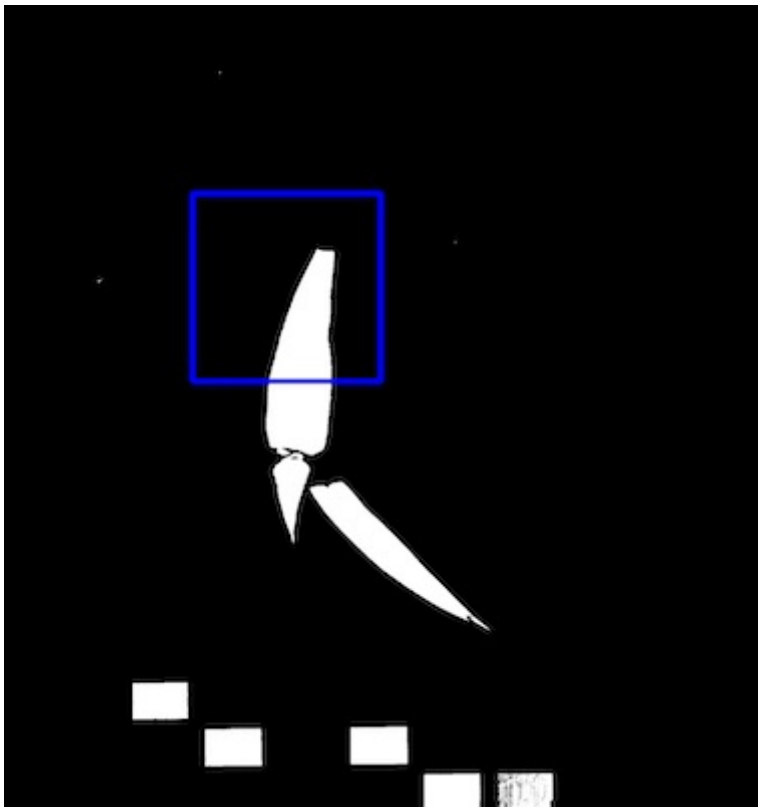


```
# Find Objects

# Inputs:
#   img  - RGB or grayscale image data for plotting
#   mask - Binary mask used for detecting contours
id_objects, obj_hierarchy = pcv.find_objects(img=index_array_gdvi.array_data,
mask=gdvi_thresh)
```

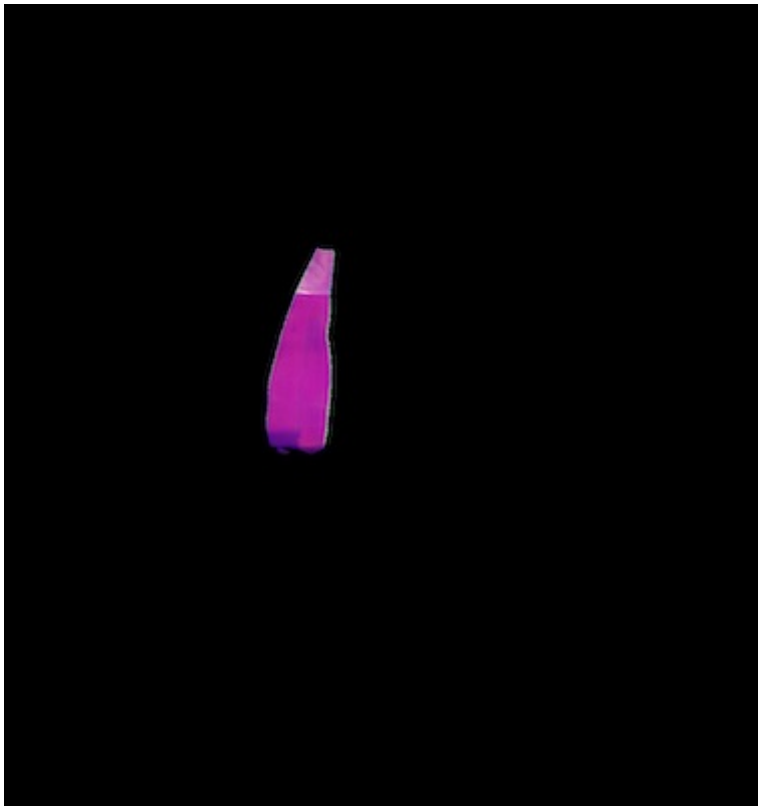


## 針對GDVI影像進行ROI運算

```
# Define ROI

# Inputs:
#   img - RGB or grayscale image to plot the ROI on
#   x   - The x-coordinate of the upper left corner of the rectangle
#   y   - The y-coordinate of the upper left corner of the rectangle
#   h   - The height of the rectangle
#   w   - The width of the rectangle
roi, roi_hierarchy= pcv.roi.rectangle(img=gdvi_thresh, x=500, y=500, h=300, w=300)
```

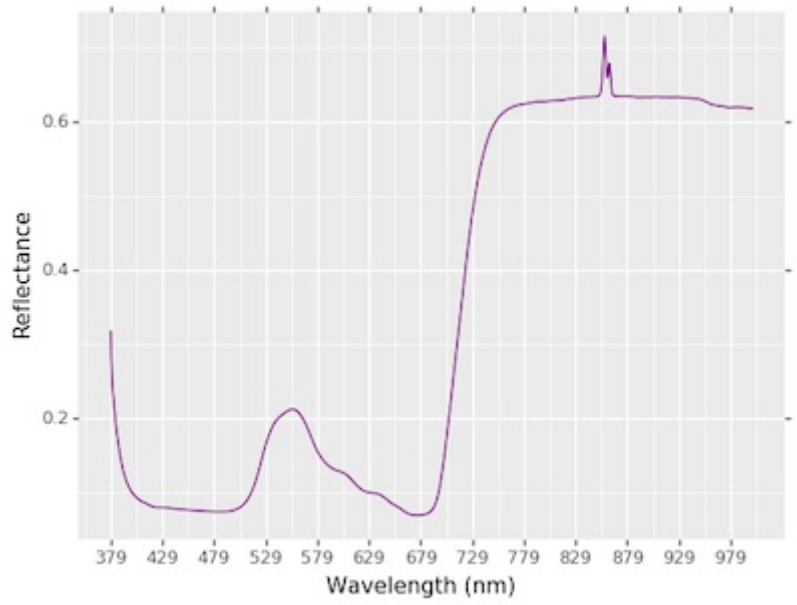


```
# Filter object by a defined region of interest

# Inputs:
#    img            - img to display kept objects
#    roi_contour    - contour of roi, output from any ROI function
#    roi_hierarchy  - contour of roi, output from any ROI function
#    object_contour - contours of objects, output from pcv.find_objects function
#    obj_hierarchy  - hierarchy of objects, output from pcv.find_objects function
#    roi_type       - 'partial' (default, for partially inside), 'cutto', or
#                     'largest' (keep only largest contour)
roi_objects, hierarchy, kept_mask, obj_area =
pcv.roi_objects(img=index_array_gdvi.array_data, roi_contour=roi,

 roi_hierarchy=roi_hierarchy,

 object_contour=id_objects,

 obj_hierarchy=obj_hierarchy,
                                                          roi_type='partial')
```



## 針對GDVI中ROI影像進行波長的直方圖分析

```
# Extract statistics about an index for the leaf region

# Inputs:
#   array         - Hyperspectral index data instance
#   mask          - Binary mask image data
pcv.hyperspectral.analyze_index(array=index_array_gdvi, mask=kept_mask)
```



# PlantCV對Thermal Data的支援（FLIR相機）

> FLIR相機將影像的Thermal data存成一個csv 格式。

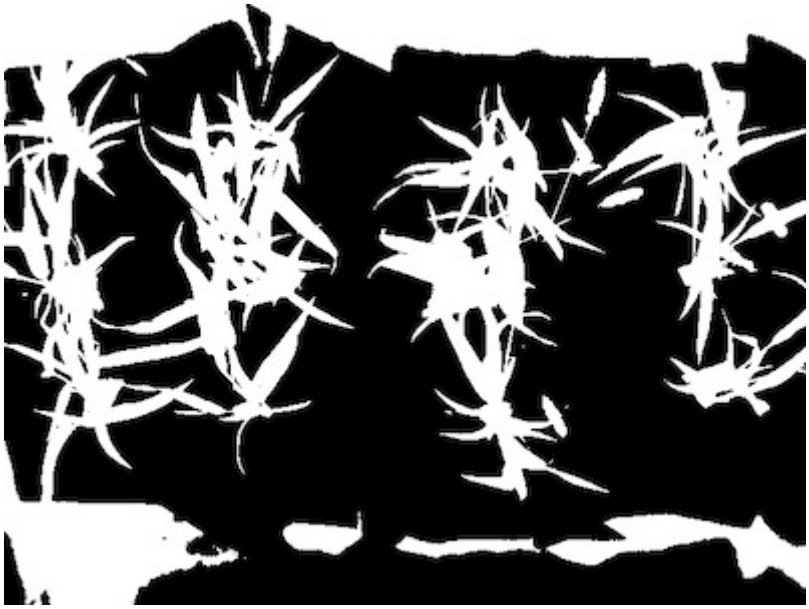
```
# Read raw thermal data

# Inputs:
#   filename - Image file to be read (possibly including a path)
#   mode - Return mode of image; either 'native' (default), 'rgb', 'gray', 'envi', or
'csv'
thermal_data,path,filename = pcv.readimage(filename='FLIR2600.csv', mode="csv")
```

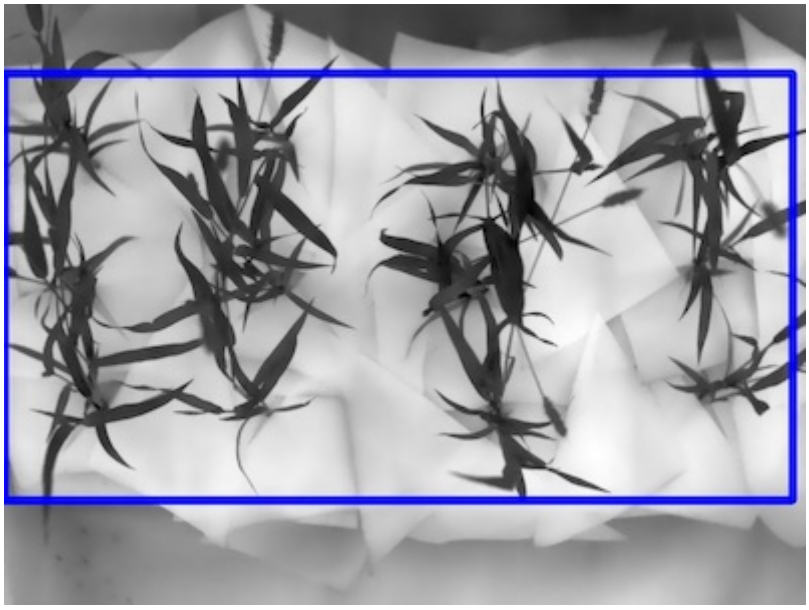
> 原始數據作為圖像。該特定圖像是由FLIR攝像機捕獲的，該攝像機保存了一個.csv熱數據文件。

> 灰度圖像的像素值通常在0-255之間，但是由於原始數據以攝氏度為單位進行測量，因此圖像表示可能看起來很暗。在此示例的熱圖像中，最大值約為38攝氏度，仍然是相當深的灰色陰影。與典型圖像（通常為uint8或8位無符號整數）相比，該圖像還具有不同的數據類型，因為它是float64或Double-precision浮點格式。在將調試模式設置為“繪圖”的Jupyter Notebook中開發工作流程時，後端繪圖將自動縮放圖像，以便於查看。重新縮放原始數據將提供一幅圖像，其像素值範圍從指定的最小值到指定的最大值，並將數據類型更改為與其他可拍攝灰度圖像的PlantCV函數兼容的數據類型。

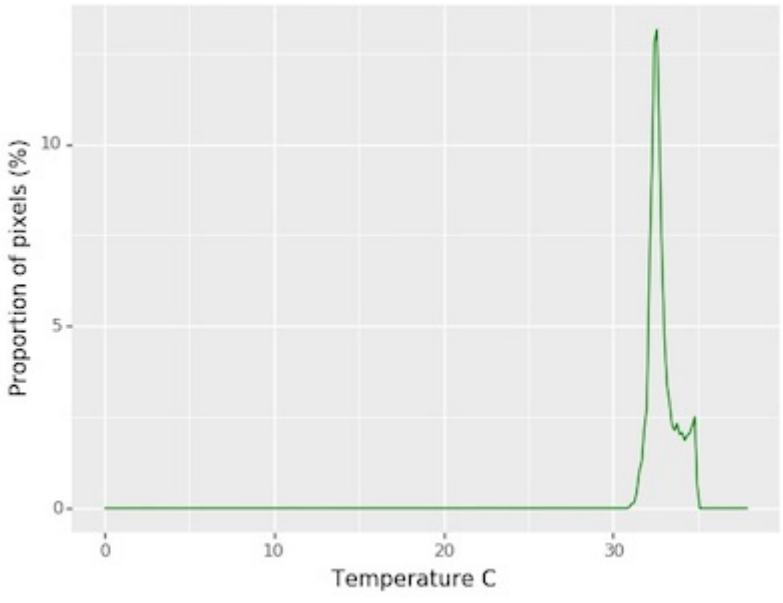
```
# Rescale the thermal data to a colorspace with range 0-255

# Inputs:
#   gray_img - Grayscale image data
#   min_value - New minimum value for range of interest. default = 0
#   max_value - New maximum value for range of interest. default = 255
scaled_thermal_img = pcv.transform.rescale(gray_img=thermal_data)
```

```
# Threshold the thermal data to make a binary mask

# Inputs:
#   gray_img - Grayscale image data
#   threshold- Threshold value (between 0-255)
#   max_value - Value to apply above threshold (255 = white)
#   object_type - 'light' (default) or 'dark'. If the object is lighter than the
background then standard
#                 threshold is done. If the object is darker than the background then
inverse thresholding is done.
bin_mask = pcv.threshold.binary(gray_img=thermal_data, threshold=35, max_value=255,
object_type='dark')
```





```
##### Analysis #####

# Analyze thermal data

# Inputs:
#   img - Array of thermal values
#   mask - Binary mask made from selected contours
#   histplot - If True plots histogram of intensity values (default histplot = False)
analysis_img = pcv.analyze_thermal_values(thermal_array=thermal_data, mask=kept_mask,
histplot=True)
```

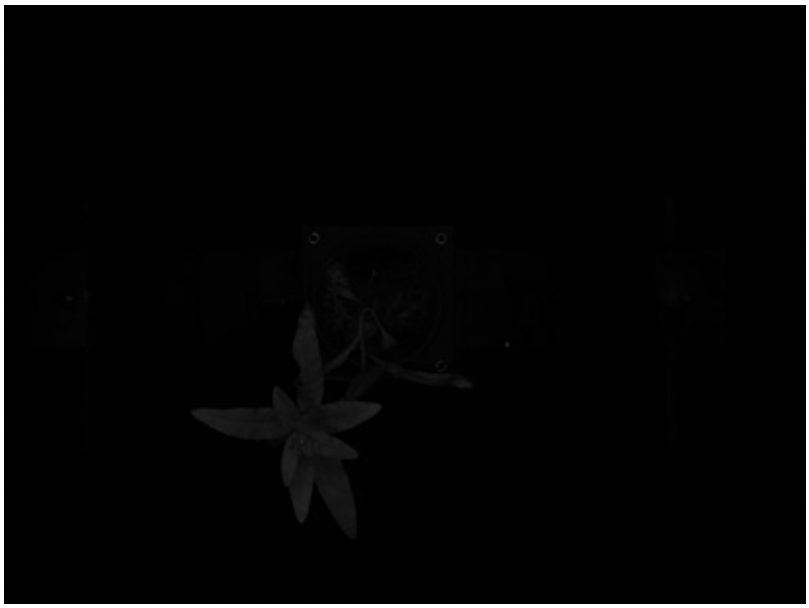



# PlantCV 對飽和脈衝葉綠素螢光影像(PSII)的支援

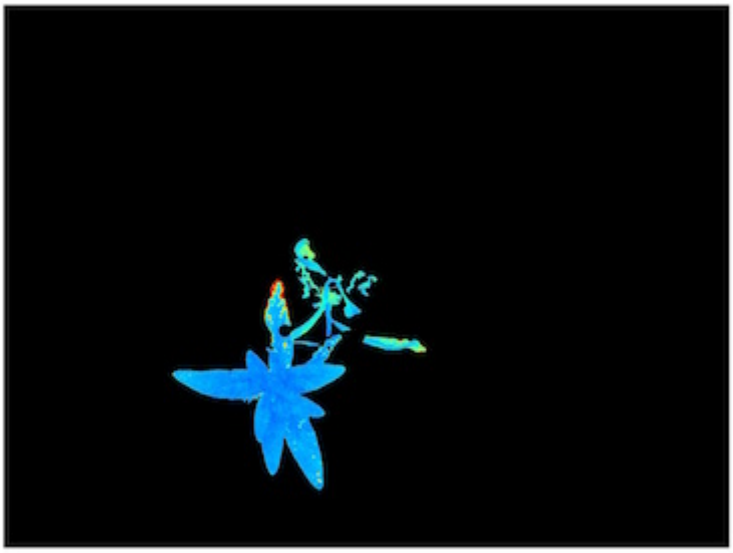
> PlantCV提供計算最大螢光有效光量子之螢光值, $F_o/F_m = (F_m - F_o)/F_m$

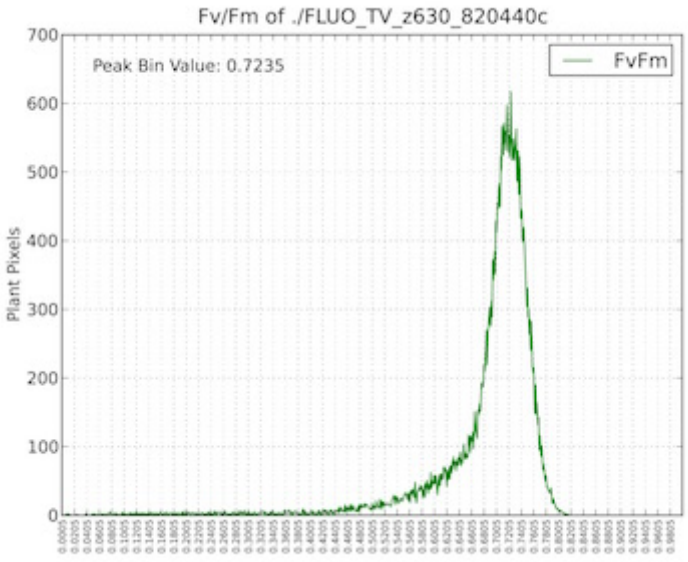
> 需要3個必需的輸入：
>
> 1. 圖片1：F0（又名Fdark / null）圖片。
> 2. 圖片2：Fmin圖片。
> 3. 圖片3：Fmax圖片。

> 從上到下輸入圖像：F0（空圖像，也稱為Fdark）； Fmin圖像； Fmax圖像。



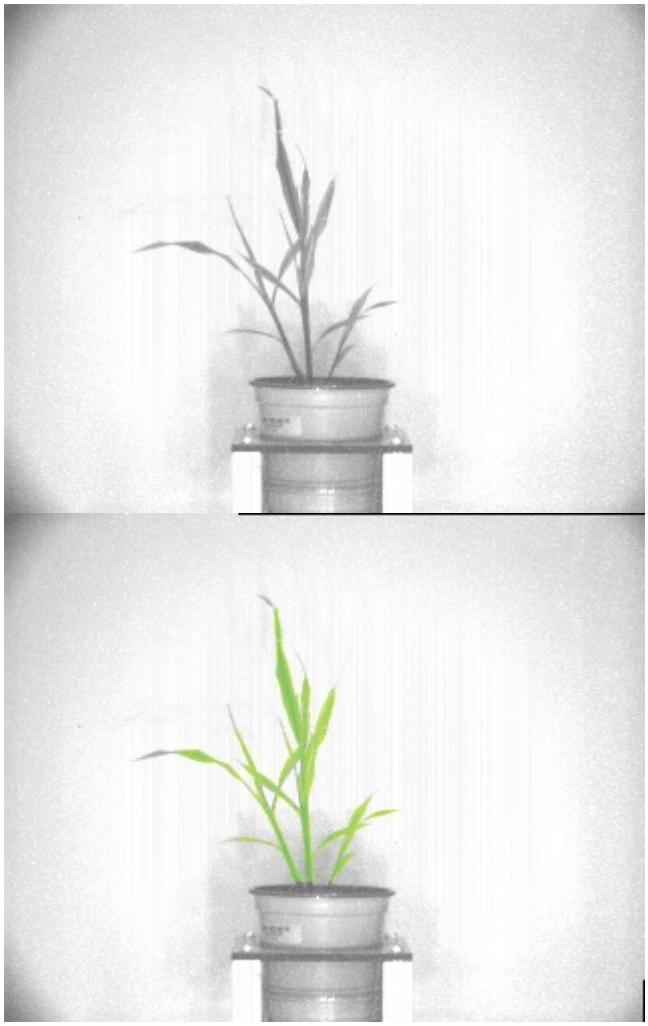
（頂部）通過Fv / Fm值偽彩色的圖像。（底部）原始Fv / Fm值的直方圖。

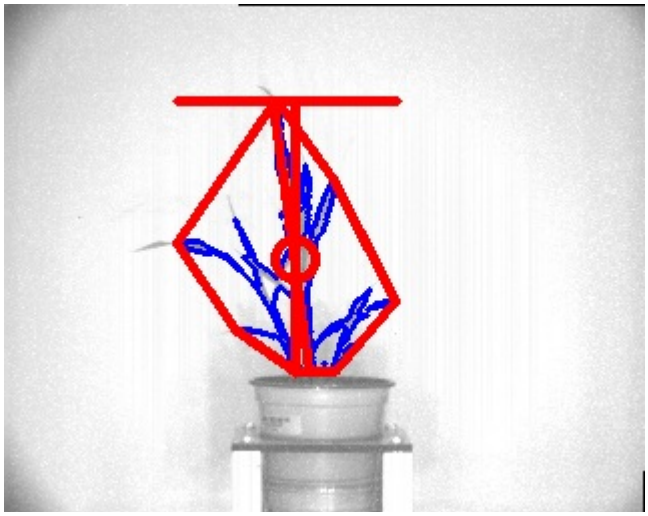


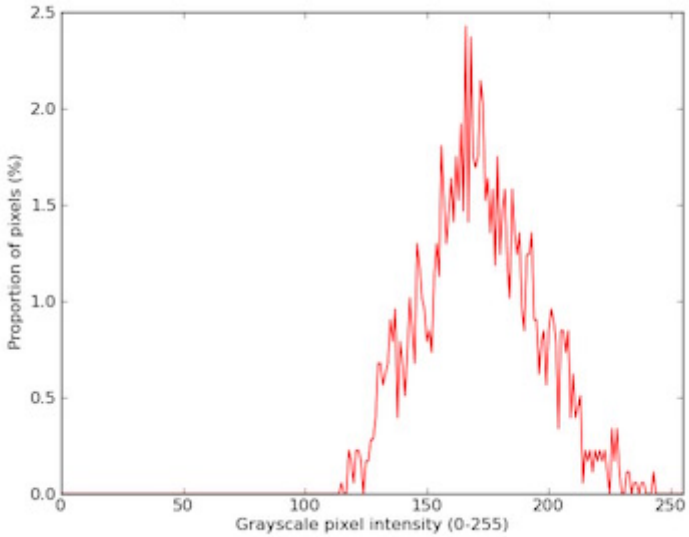


# PlantCV對近紅外光影像（NIR Image）的支援

從上到下：通過信號強度偽彩色的對像圖像（較深的綠色更黑，黃色更白）；在原始圖像上打印的植物的形狀屬性；信號強度值的直方圖。








# PlantCV對Machine Learning的支援

樸素貝葉斯分類器(Naive Bayes Classifier)

用途：用於幫助區分植物和背景