Reading Multi Spectral Images

https://nbviewer.jupyter.org/github/thomasaarholt/hyperspy-demos/blob/master/2_SVD_and_BSS.ipynb (https://nbviewer.jupyter.org/github/thomasaarholt/hyperspy-demos/blob/master/2_SVD_and_BSS.ipynb)

Bands and Wavelengths

When talking about spectral data, we talk from both, the electromagnetic spectrum and image bands. Spectral remote sensing data are collected by powerful camera-like instruments known as imaging spectrometers. Imaging spectrometers collect reflected light energy in “bands.”

A band represents a segment of the electromagnetic spectrum. For example, the wavelength values between 800 nanometers (nm) and 850 nm might be one band captured by an imaging spectrometer. The imaging spectrometer collects reflected light energy within a pixel area on the ground. Since an imaging spectrometer collects many different types of light - for each pixel the amount of light energy for each type of light or band will be recorded. So, for example, a camera records the amount of red, green and blue light for each pixel.

Often when we work with a multispectral dataset, the band information is reported as the center wavelength value. This value represents the center point value of the wavelengths represented in that band. Thus in a band spanning 800-850 nm, the center would be 825 nm.

Spectral Resolution

The spectral resolution of a dataset that has more than one band, refers to the spectral width of each band in the dataset. While a general spectral resolution of the sensor is often provided, not all sensors collect information within bands of uniform widths.

Spatial Resolution

The spatial resolution of a raster represents the area on the ground that each pixel covers. If you have smaller pixels in a raster the data will appear more “detailed.” If you have large pixels in a raster, the data will appear more coarse or “fuzzy.”

Multispectral Imagery

Images obtained with a ADC Lite - Tetracam's Lightweight ADC

I made pitures about:

> Aluminum , Copper, Brass, Iron, Stainless Steel, Painted Iron

http://tetracam.com/Products-ADC_Lite.htm (http://tetracam.com/Products-ADC_Lite.htm)

MRobalinho - 25-03-2019

In [1]:

```
# Add libraries
import glob, os
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from PIL import Image
from openpyxl import load_workbook
```

In [2]:

```
# Verify my current folder
currDir = os.path.dirname(os.path.realpath("__file__"))
mypath = currDir
print(currDir)
```

```
C:\Users\manuel.robalinho\Google Drive\UPT_Portucalense\Trabalho final\Class
ificacao_Sucata\Jupyter_Notebook
```

In [3]:

```
# Path to the image files
folder = "imagedata03"
path = currDir + "/" + folder + "/"

# Part name of file to filter files
end_file = "_1.jpg"
```

In [4]:

```
# Read files from folder
print(path)
print(' ---- IMAGES ON THE FOLDER -------')

for file in os.listdir(path):
    if file.endswith(end_file):
        print(os.path.join(file))
```

```
C:\Users\manuel.robalinho\Google Drive\UPT_Portucalense\Trabalho final\Class
ificacao_Sucata\Jupyter_Notebook/imagedata03/
 ---- IMAGES ON THE FOLDER -------
Aluminum_1.jpg
Brass_1.jpg
CopperWire_1.jpg
Copper_1.jpg
Iron_1.jpg
PaintedIron_1.jpg
StainlessSteel_1.jpg
```

In [5]:

```
# Create Data Frame with image information
df_image = []
```

In [6]:

```
# Look from an chanel from then image

def channel(img, n):
    """Isolate the nth channel from the image.

       n = 0: red, 1: green, 2: blue
    """
    a = np.array(img)
    a[:,:,(n!=0, n!=1, n!=2)] *= 0
#   a[:,:,n] *= 0
#   print(Image.fromarray(a), 'Get Channel n: ', n)

    print('Get Channel n: ', n)
    return Image.fromarray(a)

# def to resize
# Given parameters : image , number to divide (resize)
def imageResize(img, n):
    width, height = img.size

    print('Original size:', width, '/', height, 'Resize:',n)

    newWidth = int(width / n)
    newHeight = int(height / n)
    img.resize((newWidth, newHeight), Image.ANTIALIAS)
    print('New size:', newWidth, '/', newHeight)
    return img
```

In [7]:

```
# Obtain main color from image
# https://convertingcolors.com/rgb-color-169_171_170.html

def get_main_color(path, file):
    img = Image.open(path+file)
    colors = img.getcolors( 1024*1024) #put a higher value if there are many colors in your
    print('Get main Color file:', file)
    max_occurence, most_present = 0, 0
    try:
        for c in colors:
            if c[0] > max_occurence:
                (max_occurence, most_present) = c
        return most_present
    except TypeError:
        raise Exception("Too many colors in the image")
```

In [8]:

```
def print_file(path, xfile):
  print('------------------------------------------------------------------------')
  tif_f1 = Image.open(path+xfile)

  print('Inf.File:',xfile)

  # Transform Image to array
  aArray = np.array(tif_f1)
  # Array sum
  xsum = aArray.sum() / 1000000

  # Get channel 0
  x0_channel = channel(tif_f1, 0)
  aArray = np.array(x0_channel)
  xsum_0 = aArray.sum() / 1000000

  # Get channel 1
  x1_channel = channel(tif_f1, 1)
  aArray = np.array(x1_channel)
  xsum_1 = aArray.sum() / 1000000

  # Get channel 2
  x2_channel = channel(tif_f1, 2)
  aArray = np.array(x2_channel)
  xsum_2 = aArray.sum() / 1000000

  # Histogram from image
  aHist = tif_f1.histogram()
  hsum = sum(aHist) / 100000

  # Histogram channel 0
  aHist_0 = x0_channel.histogram()
  hsum_0 = sum(aHist_0) / 100000

  # Histogram channel 1
  aHist_1 = x1_channel.histogram()
  hsum_1 = sum(aHist_1) / 100000

  # Histogram chanel 0
  aHist_2 = x2_channel.histogram()
  hsum_2 = sum(aHist_2) / 100000

  # number elements on list
  nlist = len(aHist)

  # Get color
  pix_val = list(tif_f1.getdata())

  main_color = get_main_color(path, xfile)
  print('Main color from image:',xfile, main_color)

  # Transform tuple in a list
  pix_val_flat = [x for sets in pix_val for x in sets]
  # Sum the list and medium list pixel
  sum_pix = sum(pix_val_flat)
  med_pix = sum_pix / len(pix_val_flat)

  # Obtain name file without extension
  sample_name = os.path.basename(xfile).split('_')[0]
```

```
# Print information
print(sample_name,' Size:',tif_f1.size, ' Format:',tif_f1.format, ' Mode:', tif_f1.mode,'
# More information image
print('         ',' Sum array:',xsum, ' Sum Ch 0:', xsum_0, ' Sum Ch 1:', xsum_1, ' Sum C
# More information image
print('         ',' Histog:', hsum ,'  N.List elem:', nlist ,' Color:', med_pix)
 # insert information in a Pandas Data Frame
df_image.append((folder, xfile, sample_name, tif_f1.size, tif_f1.format, tif_f1.mode ,tif
                 xsum, xsum_0, xsum_1, xsum_2, hsum, nlist, med_pix))
```

In [9]:

```
# Create Data Frame with image information
df_image = []

xend_file = "*" + end_file
os.chdir(path)
for file in glob.glob(xend_file):
 #   print(file)
    print_file(path,file)
```

```
-----------------------------------------------------------------------
Inf.File: Aluminum_1.jpg
Get Channel n:  0
Get Channel n:  1
Get Channel n:  2
Get main Color file: Aluminum_1.jpg
Main color from image: Aluminum_1.jpg (189, 185, 176)
Aluminum  Size: (5312, 2988)  Format: JPEG  Mode: RGB  Bands Extrems: ((0, 2
55), (0, 255), (9, 255))
           Sum array: 3487.099309  Sum Ch 0: 2614.556834  Sum Ch 1: 2600.402
524  Sum Ch 2: 2567.107247
           Histog: 476.16768   N.List elem: 768  Color: 163.4312224844828
-----------------------------------------------------------------------
Inf.File: Brass_1.jpg
Get Channel n:  0
Get Channel n:  1
Get Channel n:  2
Get main Color file: Brass_1.jpg
Main color from image: Brass_1.jpg (179, 180, 182)
Brass  Size: (5312, 2988)  Format: JPEG  Mode: RGB  Bands Extrems: ((14, 25
5), (6, 255), (0, 255))
           Sum array: 3351.352573  Sum Ch 0: 2626.345572  Sum Ch 1: 2553.685
796  Sum Ch 2: 2466.288501
           Histog: 476.16768   N.List elem: 768  Color: 160.5804045541268
-----------------------------------------------------------------------
Inf.File: CopperWire_1.jpg
Get Channel n:  0
Get Channel n:  1
Get Channel n:  2
Get main Color file: CopperWire_1.jpg
Main color from image: CopperWire_1.jpg (219, 218, 223)
CopperWire  Size: (5312, 2988)  Format: JPEG  Mode: RGB  Bands Extrems: ((0,
255), (0, 255), (0, 255))
           Sum array: 2595.753222  Sum Ch 0: 2314.445608  Sum Ch 1: 2313.038
337  Sum Ch 2: 2263.236573
           Histog: 476.16768   N.List elem: 768  Color: 144.71205853366612
-----------------------------------------------------------------------
Inf.File: Copper_1.jpg
Get Channel n:  0
Get Channel n:  1
Get Channel n:  2
Get main Color file: Copper_1.jpg
Main color from image: Copper_1.jpg (201, 205, 204)
Copper  Size: (5312, 2988)  Format: JPEG  Mode: RGB  Bands Extrems: ((35, 25
5), (17, 255), (6, 255))
           Sum array: 3400.502407  Sum Ch 0: 2731.080699  Sum Ch 1: 2549.470
157  Sum Ch 2: 2414.918847
           Histog: 476.16768   N.List elem: 768  Color: 161.61260048141025
-----------------------------------------------------------------------
Inf.File: Iron_1.jpg
```

```
Get Channel n:  0
Get Channel n:  1
Get Channel n:  2
Get main Color file: Iron_1.jpg
Main color from image: Iron_1.jpg (174, 174, 176)
Iron  Size: (5312, 2988)  Format: JPEG  Mode: RGB  Bands Extrems: ((12, 25
5), (23, 255), (21, 255))
           Sum array: 2829.021121  Sum Ch 0: 2388.854306  Sum Ch 1: 2389.478
945  Sum Ch 2: 2345.655166
           Histog: 476.16768   N.List elem: 768  Color: 149.61091893091105
--------------------------------------------------------------------------
Inf.File: PaintedIron_1.jpg
Get Channel n:  0
Get Channel n:  1
Get Channel n:  2
Get main Color file: PaintedIron_1.jpg
Main color from image: PaintedIron_1.jpg (185, 185, 183)
PaintedIron  Size: (5312, 2988)  Format: JPEG  Mode: RGB  Bands Extrems:
((1, 238), (0, 242), (0, 245))
           Sum array: 168.784754  Sum Ch 0: 2928.383094  Sum Ch 1: 2929.9995
42  Sum Ch 2: 2900.33671
           Histog: 476.16768   N.List elem: 768  Color: 183.94191193320808
--------------------------------------------------------------------------
Inf.File: StainlessSteel_1.jpg
Get Channel n:  0
Get Channel n:  1
Get Channel n:  2
Get main Color file: StainlessSteel_1.jpg
Main color from image: StainlessSteel_1.jpg (166, 166, 166)
StainlessSteel  Size: (5312, 2988)  Format: JPEG  Mode: RGB  Bands Extrems:
((3, 255), (0, 255), (1, 255))
           Sum array: 3822.686116  Sum Ch 0: 2716.61188  Sum Ch 1: 2710.0994
22  Sum Ch 2: 2690.94211
           Histog: 476.16768   N.List elem: 768  Color: 170.47888281707822
```

In [10]:

```
df = pd.DataFrame(df_image,columns=['Folder','File','Material','Size','Format','Mode','Banc
                                    'Array_sum', 'Sum_Ch0','Sum_Ch1','Sum_Ch2',
                                    'Histogram','Number_list_elements','Color'])
df.head()
```

Out[10]:

	Folder	File	Material	Size	Format	Mode	Bands Extrems	Array_sum	S
0	imagedata03	Aluminum_1.jpg	Aluminum	(5312, 2988)	JPEG	RGB	((0, 255), (0, 255), (9, 255))	3487.099309	261
1	imagedata03	Brass_1.jpg	Brass	(5312, 2988)	JPEG	RGB	((14, 255), (6, 255), (0, 255))	3351.352573	262
2	imagedata03	CopperWire_1.jpg	CopperWire	(5312, 2988)	JPEG	RGB	((0, 255), (0, 255), (0, 255))	2595.753222	231
3	imagedata03	Copper_1.jpg	Copper	(5312, 2988)	JPEG	RGB	((35, 255), (17, 255), (6, 255))	3400.502407	273
4	imagedata03	Iron_1.jpg	Iron	(5312, 2988)	JPEG	RGB	((12, 255), (23, 255), (21, 255))	2829.021121	238

In [11]:

```
# Verify my current folder
path = mypath + r"/upt_data.xlsx"
print('Write statistics into file :', path)

# Block to Read excel old excel file
book = load_workbook(path)
writer = pd.ExcelWriter(path, engine = 'openpyxl')
writer.book = book
# ------------------------

# Write statistics into excel file
#writer = pd.ExcelWriter(path, engine = 'xlsxwriter') # only for new excelfile
df.to_excel(writer, sheet_name = folder)
writer.save()
writer.close()
```

```
Write statistics into file : C:\Users\manuel.robalinho\Google Drive\UPT_Port
ucalense\Trabalho final\Classificacao_Sucata\Jupyter_Notebook/upt_data.xlsx
```

In [12]:

```
df_plot = pd.DataFrame(df, columns=["Material", "Array_sum", "Sum_Ch0", "Sum_Ch1", "Sum_Ch2
df_plot
```

Out[12]:

	Material	Array_sum	Sum_Ch0	Sum_Ch1	Sum_Ch2	Color
0	Aluminum	3487.099309	2614.556834	2600.402524	2567.107247	163.431222
1	Brass	3351.352573	2626.345572	2553.685796	2466.288501	160.580405
2	CopperWire	2595.753222	2314.445608	2313.038337	2263.236573	144.712059
3	Copper	3400.502407	2731.080699	2549.470157	2414.918847	161.612600
4	Iron	2829.021121	2388.854306	2389.478945	2345.655166	149.610919
5	PaintedIron	168.784754	2928.383094	2929.999542	2900.336710	183.941912
6	StainlessSteel	3822.686116	2716.611880	2710.099422	2690.942110	170.478883

In [13]:

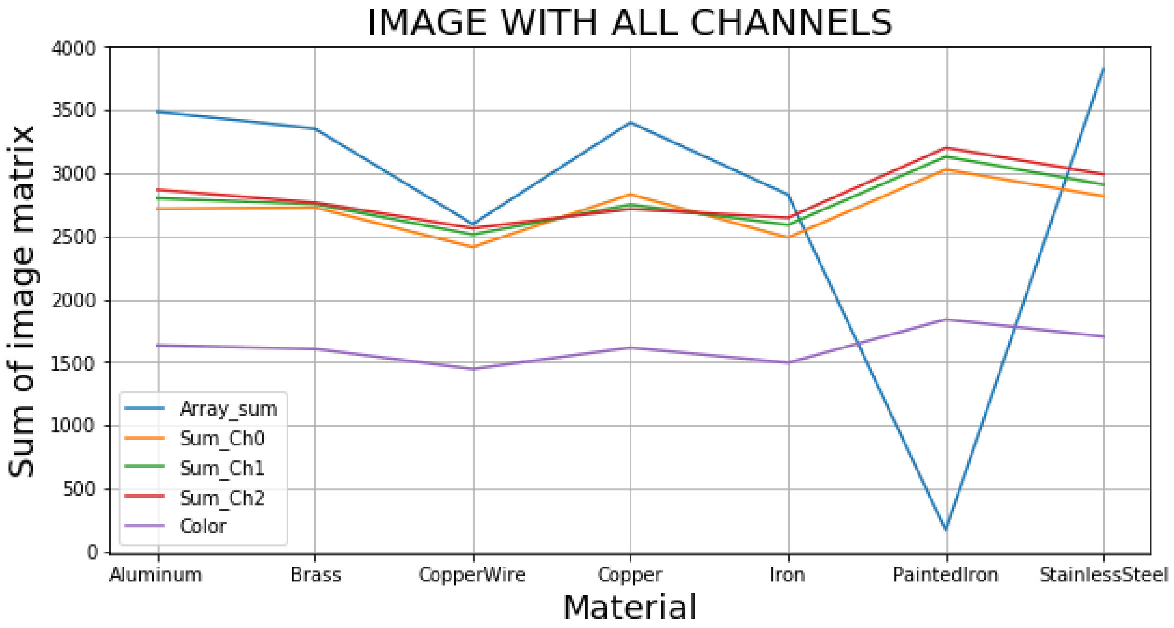
```
df_plot.Sum_Ch0 = df_plot.Sum_Ch0 + 100 # to have diference lines during plot
df_plot.Sum_Ch1 = df_plot.Sum_Ch1 + 200
df_plot.Sum_Ch2 = df_plot.Sum_Ch2 + 300
df_plot.Color  = df_plot.Color * 10
df_plot
```

Out[13]:

	Material	Array_sum	Sum_Ch0	Sum_Ch1	Sum_Ch2	Color
0	Aluminum	3487.099309	2714.556834	2800.402524	2867.107247	1634.312225
1	Brass	3351.352573	2726.345572	2753.685796	2766.288501	1605.804046
2	CopperWire	2595.753222	2414.445608	2513.038337	2563.236573	1447.120585
3	Copper	3400.502407	2831.080699	2749.470157	2714.918847	1616.126005
4	Iron	2829.021121	2488.854306	2589.478945	2645.655166	1496.109189
5	PaintedIron	168.784754	3028.383094	3129.999542	3200.336710	1839.419119
6	StainlessSteel	3822.686116	2816.611880	2910.099422	2990.942110	1704.788828

In [14]:

```
df_plot.plot(y=["Array_sum","Sum_Ch0","Sum_Ch1", "Sum_Ch2","Color"],figsize=(10,5), grid=Tr

# Obtain legend (xticks) for X axis
loc_Array_sum = np.arange(len(df_plot.index))
# Position of X labels
xtick_loc = list(loc_Array_sum)
# Name of x labels
xticks = list(df_plot.Material)
#-------

#plt.plot(df_plot.Array_sum)
plt.title('IMAGE WITH ALL CHANNELS',fontsize=20)
plt.ylabel('Sum of image matrix',fontsize=18)
plt.xticks(xtick_loc, df_plot.Material, rotation=0)
plt.xlabel('Material',fontsize=18)
plt.show()
```



In [15]:

```
# Create pivot table
df_plot1 = df_plot.groupby('Material')['Array_sum', 'Sum_Ch0','Sum_Ch1','Sum_Ch2','Color'].
df_plot1
```

Out[15]:

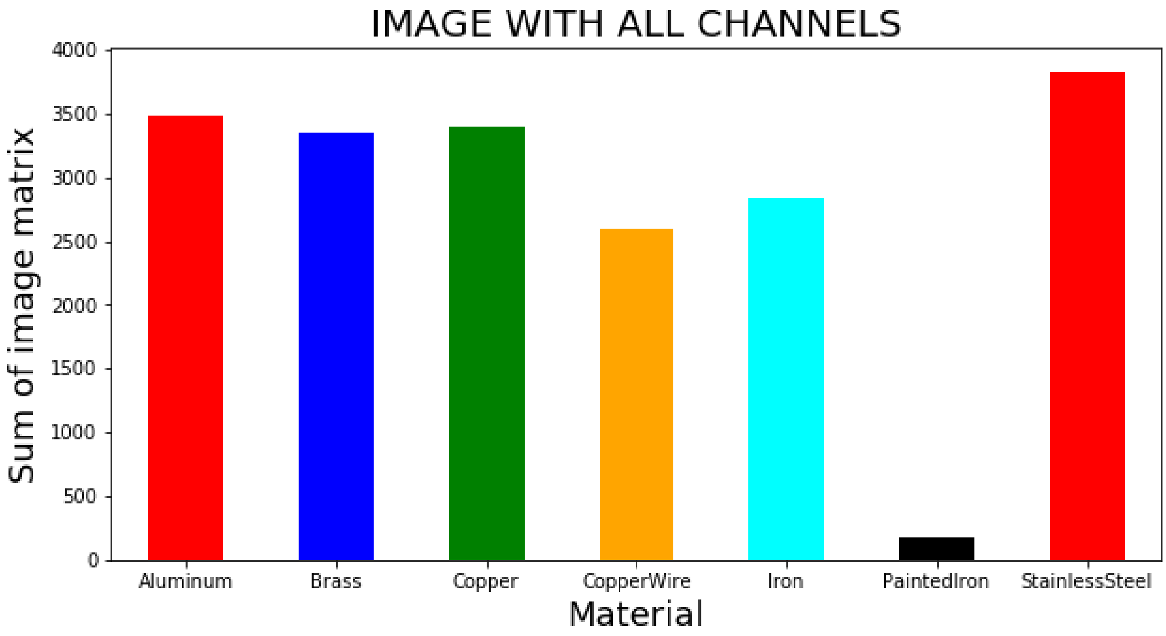
	Array_sum	Sum_Ch0	Sum_Ch1	Sum_Ch2	Color
Material					
Aluminum	3487.099309	2714.556834	2800.402524	2867.107247	1634.312225
Brass	3351.352573	2726.345572	2753.685796	2766.288501	1605.804046
Copper	3400.502407	2831.080699	2749.470157	2714.918847	1616.126005
CopperWire	2595.753222	2414.445608	2513.038337	2563.236573	1447.120585
Iron	2829.021121	2488.854306	2589.478945	2645.655166	1496.109189
PaintedIron	168.784754	3028.383094	3129.999542	3200.336710	1839.419119
StainlessSteel	3822.686116	2816.611880	2910.099422	2990.942110	1704.788828

In [16]:

```
df = pd.DataFrame(df_plot1.Array_sum)
color = ['red','blue','green','orange','cyan','black']
```

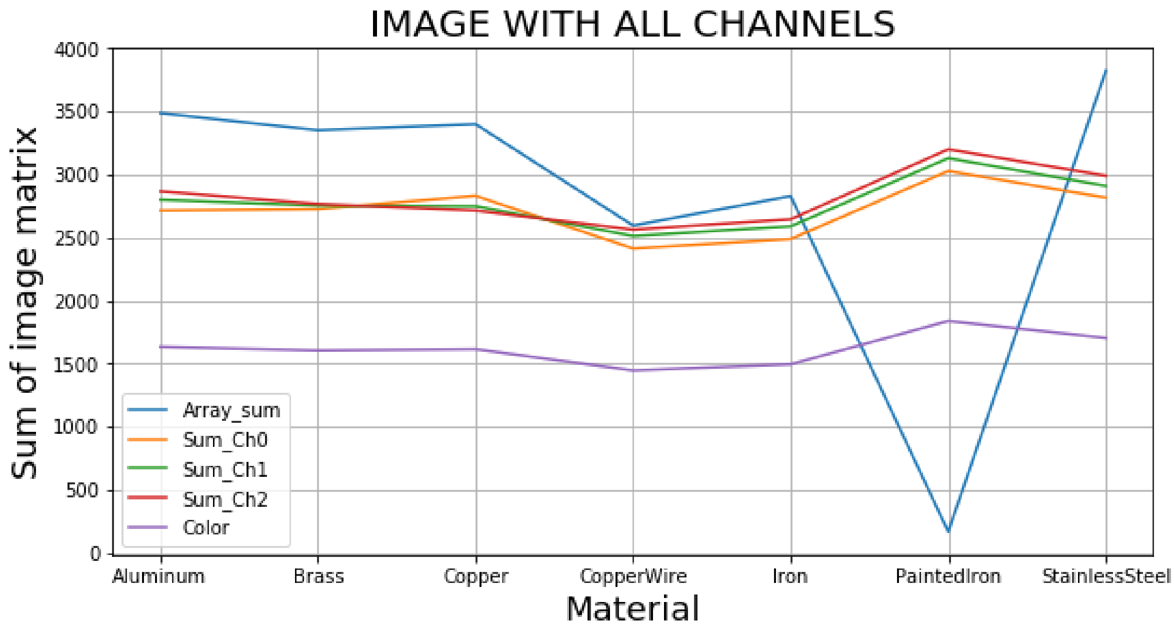
In [17]:

```
df.plot(kind='bar', y=0, color=color, legend=False, rot=0, figsize=(10,5))
plt.title('IMAGE WITH ALL CHANNELS',fontsize=20)
plt.xlabel('Material',fontsize=18)
plt.ylabel('Sum of image matrix',fontsize=18)
plt.show()
```



In [18]:

```
loc_Array_sum = np.arange(len(df_plot1.index))
xtick_loc = list(loc_Array_sum)
xticks = list(df_plot1.index)

df_plot1.plot( y=["Array_sum","Sum_Ch0","Sum_Ch1", "Sum_Ch2","Color"],figsize=(10,5), grid=
plt.xticks(xtick_loc, df_plot1.index, rotation=0)
plt.title('IMAGE WITH ALL CHANNELS',fontsize=20)
plt.xlabel('Material',fontsize=18)
plt.ylabel('Sum of image matrix',fontsize=18)
plt.show()
```



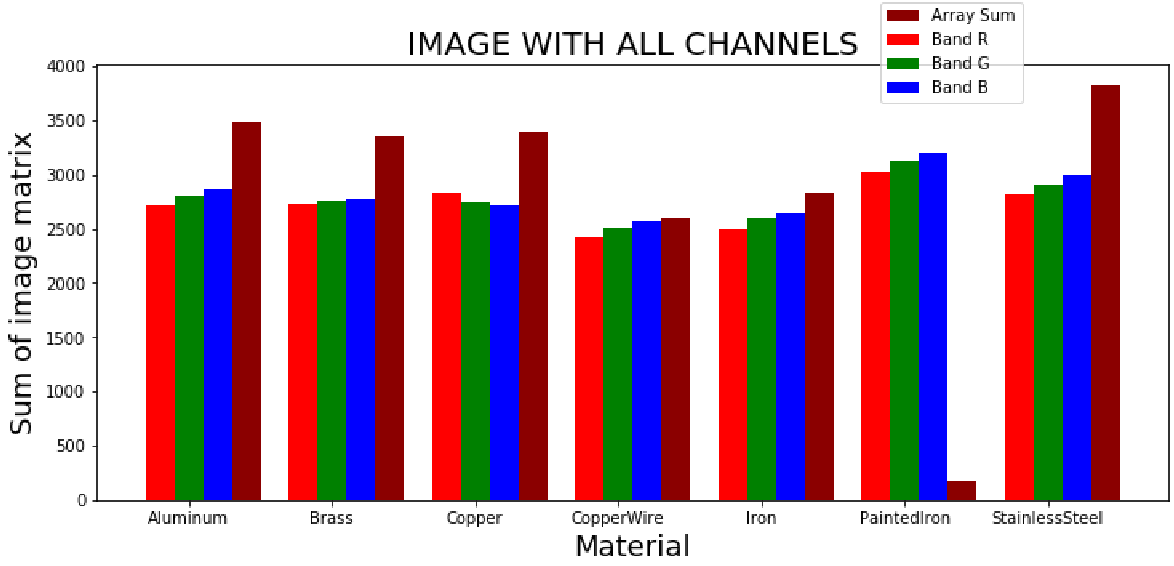
In [19]:

```
loc_Array_sum = np.arange(len(df_plot1.index))+0.1 # Offsetting the tick-label location
loc_r = np.arange(len(df_plot1.index))-0.1 # Offsetting the tick-label location
loc_g = np.arange(len(df_plot1.index))-0.3 # Offsetting the tick-label location
loc_b = np.arange(len(df_plot1.index))-0.5 # Offsetting the tick-label location

xtick_loc = list(loc_g)
xticks = list(df_plot1.index)
```

In [ ]:

In [20]:

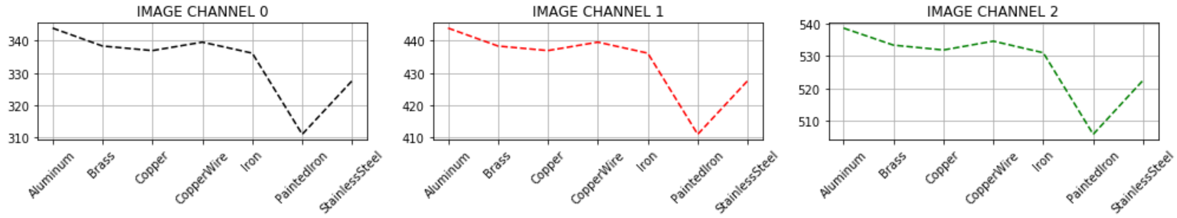
```
#Plot  Bar Graph
#df_plot1.plot(kind='bar', figsize=(12,5), grid=True, color='darkred',fontsize=18)
loc_Array_sum = np.arange(len(df_plot1.index))+0.1 # Offsetting the tick-label location
loc_b = np.arange(len(df_plot1.index))-0.1 # Offsetting the tick-label location
loc_g = np.arange(len(df_plot1.index))-0.3 # Offsetting the tick-label location
loc_r = np.arange(len(df_plot1.index))-0.5 # Offsetting the tick-label location

#xtick_loc = list(loc_Array_sum) + list(loc_r) + list(loc_g) + list(loc_b)
#xticks = list(selected.keys())+ list(rejected.keys())
colors = ['darkred','red','green','blue','orange','cyan','black']
plt.figure(figsize=(12,5))

plt.bar(loc_Array_sum, df_plot1.Array_sum, color=colors[0], width=0.2, label='Array Sum')
plt.bar(loc_r, df_plot1.Sum_Ch0, color=colors[1], width=0.2,label='Band R')
plt.bar(loc_g, df_plot1.Sum_Ch1, color=colors[2], width=0.2,label='Band G')
plt.bar(loc_b, df_plot1.Sum_Ch2, color=colors[3], width=0.2,label='Band B')

plt.title('IMAGE WITH ALL CHANNELS',fontsize=20)
plt.xlabel('Material',fontsize=18)
plt.ylabel('Sum of image matrix',fontsize=18)
plt.xticks(xtick_loc, xticks, rotation=0)
plt.legend(bbox_to_anchor=(.8,0.8),\
    bbox_transform=plt.gcf().transFigure)

plt.show()
```



In [22]:

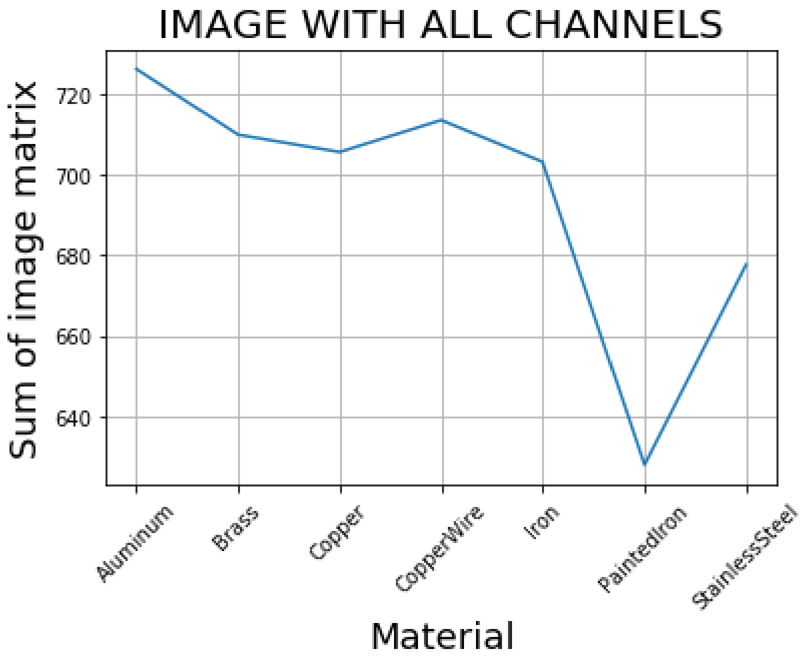
```
plt.figure(1)
plt.figure(figsize=(17, 4))
plt.tight_layout()
plt.subplot(231)
plt.title('IMAGE CHANNEL 0')
plt.xticks(rotation=45)
plt.grid(True)
plt.plot(df_plot1.Sum_Ch0, 'k--')

plt.subplot(232)
plt.title('IMAGE CHANNEL 1')
plt.xticks(rotation=45)
plt.grid(True)
plt.plot(df_plot1.Sum_Ch1,  'r--')

plt.subplot(233)
plt.title('IMAGE CHANNEL 2')
plt.xticks(rotation=45)
plt.plot(df_plot1.Sum_Ch2,  'g--')
plt.grid(True)
plt.show()
```

```
<Figure size 432x288 with 0 Axes>
```



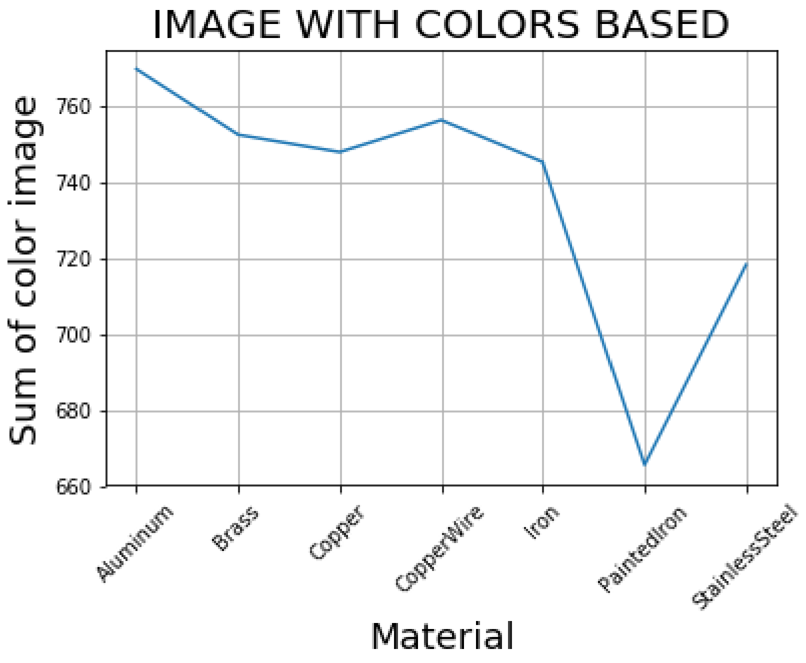
In [23]:

```
# Plot channel based
plt.plot(df_plot1.Array_sum)
plt.title('IMAGE WITH ALL CHANNELS',fontsize=20)
plt.xlabel('Material',fontsize=18)
plt.ylabel('Sum of image matrix',fontsize=18)
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```



In [24]:

```
# Plot based on color
plt.plot(df_plot1.Color)
plt.title('IMAGE WITH COLORS BASED',fontsize=20)
plt.xlabel('Material',fontsize=18)
plt.ylabel('Sum of color image',fontsize=18)
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```



In [ ]: