NIR pre-process

Function compare_preprocessing used to compare different pre-processing techniques and their combinations.

Contact info: Giverny Robert, Université de Sherbrooke, giverny.robert@usherbrooke.ca
Ryan Gosselin, Université de Sherbrooke, ryan.gosselin@usherbrooke.ca

Available at: pypi.org
Updated: 15/04/2021

Example 1. Artificial dataset

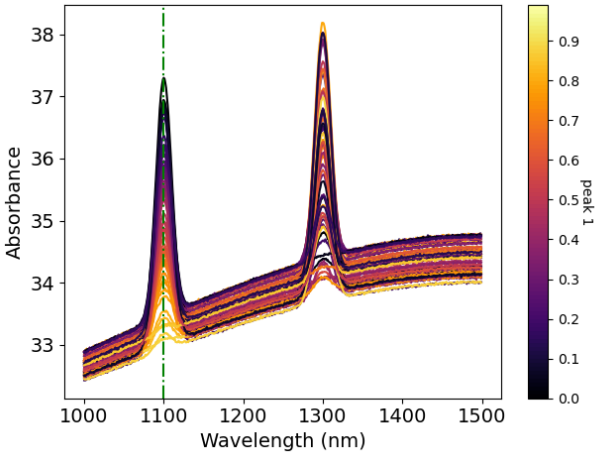
The first dataset contains 60 artificially generated signals. Each signal has 500 variables that simulate 500 wavelengths arbitrarily assigned to the frequency range 1000 to 1500 nm. Signals are made up of two normal distributions at wavelengths 1100 nm and 1300 nm and a curvilinear baseline. Each peak and baseline are multiplied by a randomly selected factor. Each distribution represents a peak associated with a chemical compound, whereas the baseline shift signals the effect of a physical property of the sample. These artificial signals are then corrupted by additive white Gaussian noise with a standard deviation of 0.01. Six pre-processing techniques are evaluated: baseline, de-trending, EMSC, MSC, SNV and SG.

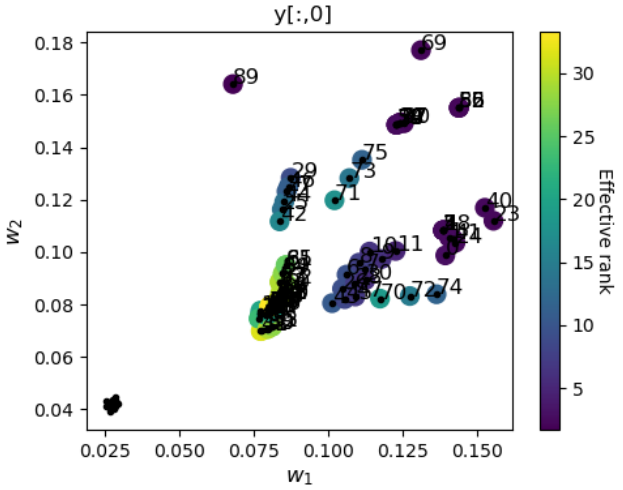
Code described below in folder “tests” of the zip file of this pypi module: ‘S_artificial data.py’ Function ‘colorspectra_y.py’ used to plot spectra.

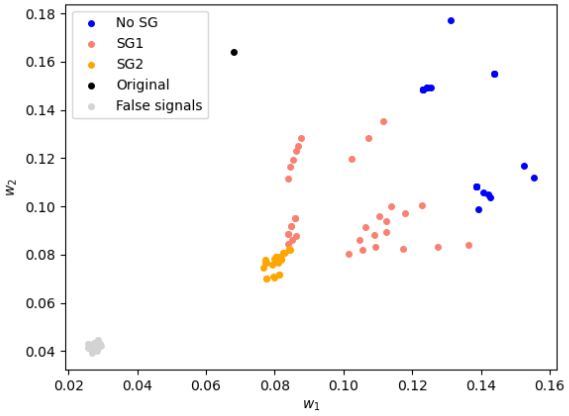
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as st
from colorspectra_y import colorspectra_y
import NIR_preprocess as nir_pre
plt.close('all')

np.random.seed(0)

#%% Choose y value
print('0 - Peak 1')
print('1 - Baseline')
user = int(input('Choose y value: '))

if int(user)==0:
    y_user = 100
else:
    y_user = 200

#%% Create dataset
x = np.arange(-15,15,0.01)

Y = []

for i in range(60):

    r0 = 10*np.random.rand()
    r1 = np.random.rand()
```

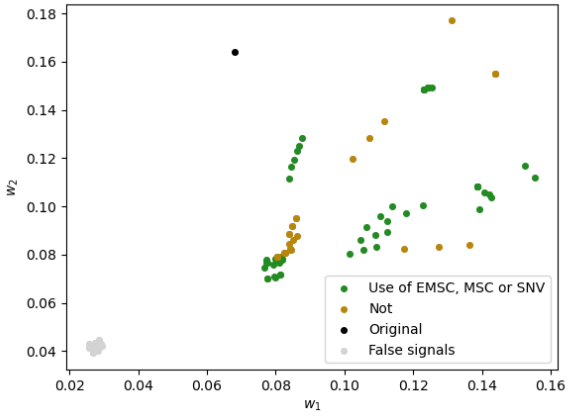
```
    r2 = np.random.rand()
    r3 = np.random.rand()

    y0 = (50+r0)*st.norm.pdf(x,-2,5)
    y1 = 10+r1*st.norm.pdf(x,-6,0.1)
    y2 = 10+r2*st.norm.pdf(x,-4,0.1)
    y3 = 10+r3*st.norm.pdf(x,6,0.1)
    y = y0+y1+y2+y3

    Y.append(y)

Y = np.array(Y)

# Only keep 2 peaks
x = x[800:1300]
Y = Y[:,800:1300]

X0 = Y
n, k = X0.shape

# Create y variable to predict
y = Y[:,y_user]
y = y[np.newaxis].T
y = 1/y

# Add noise to data
X0 = X0 + np.random.randn(n,k)/100

#%% Show original spectra
if int(user)==0:
    c_label = 'peak 1'
else:
    c_label = 'baseline'

colorspectra_y(X0,y, y_label='Absorbance', x_label='Wavelength
(nm)', colorbar_label= c_label)
plt.axvline(x=y_user, linestyle='-.', c='g')
plt.xticks(np.arange(0,k+100,100),[1000,         1100,         1200,
1300,1400,1500],fontsize=14)

#%% Compare pre-processing
combination,  datasets,  datasets0,  R2_all,  R2adj_all,  RMSECV_all,
VIP_all, Ef_all, Wt = nir_pre.compare_preprocessing(X0, y)
```

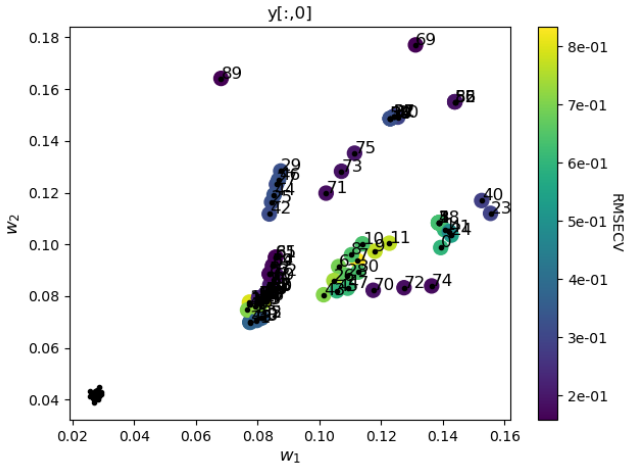
Output plots – peak 1

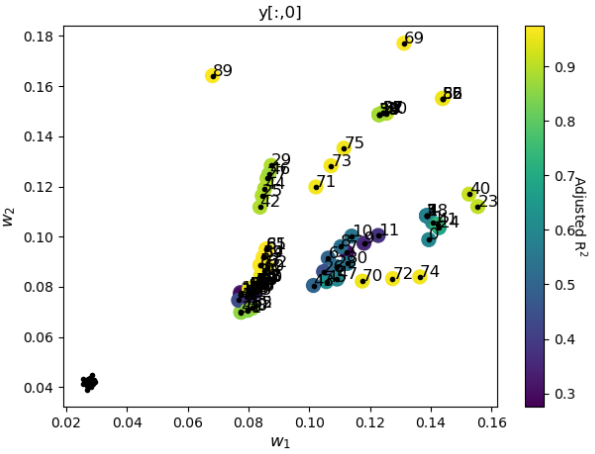




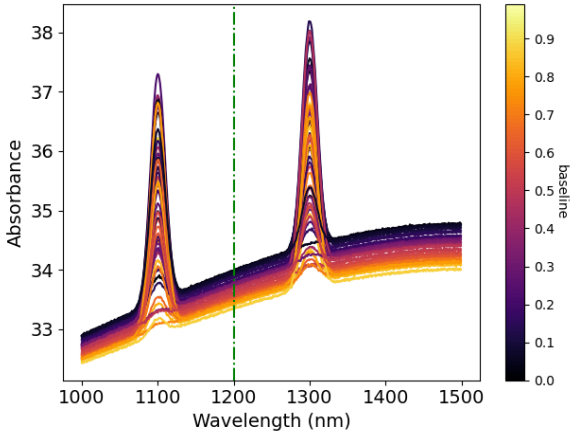


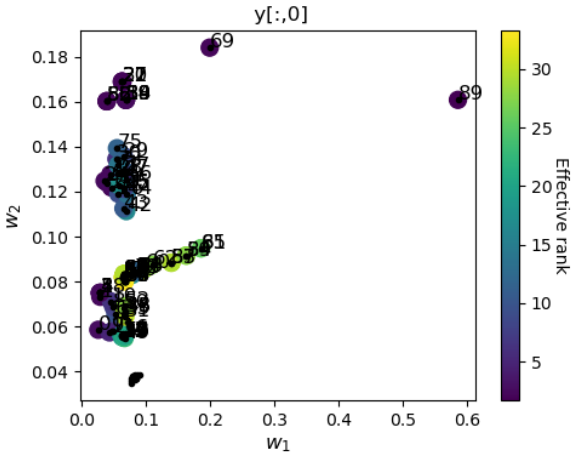


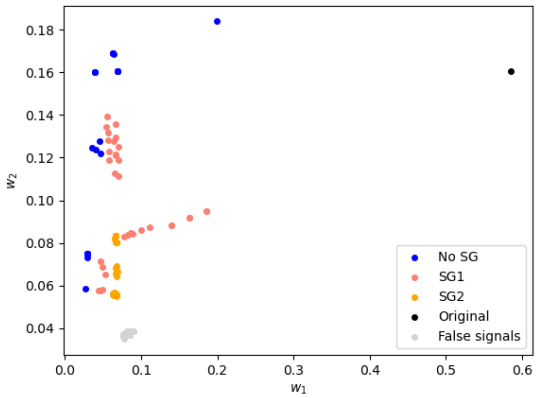


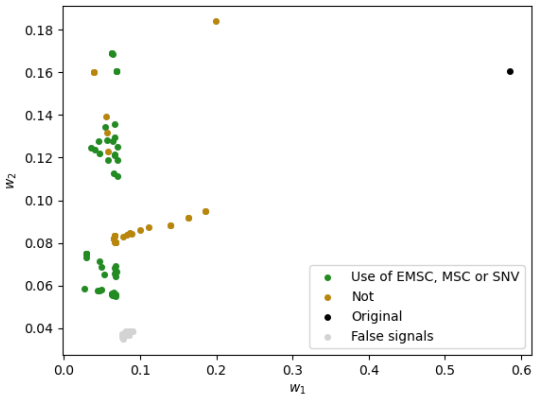


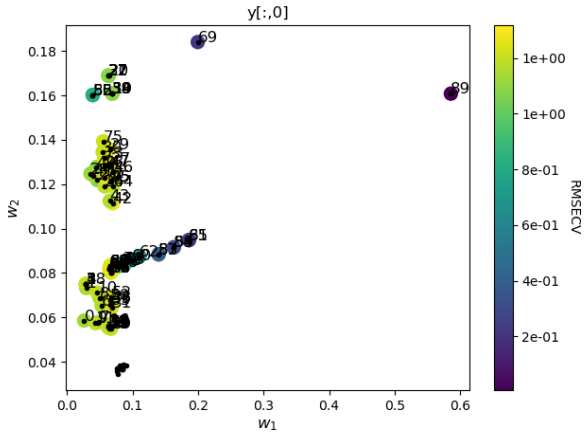
Output plots – baseline

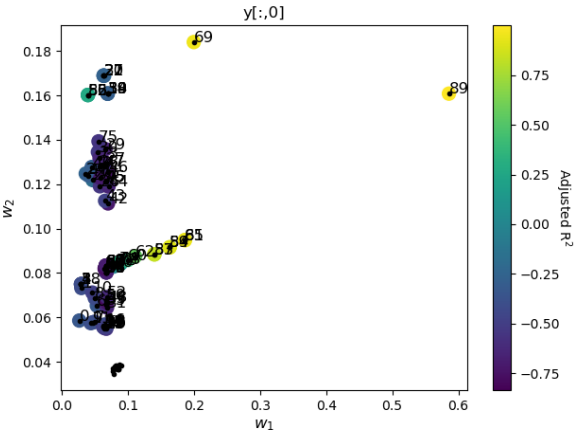












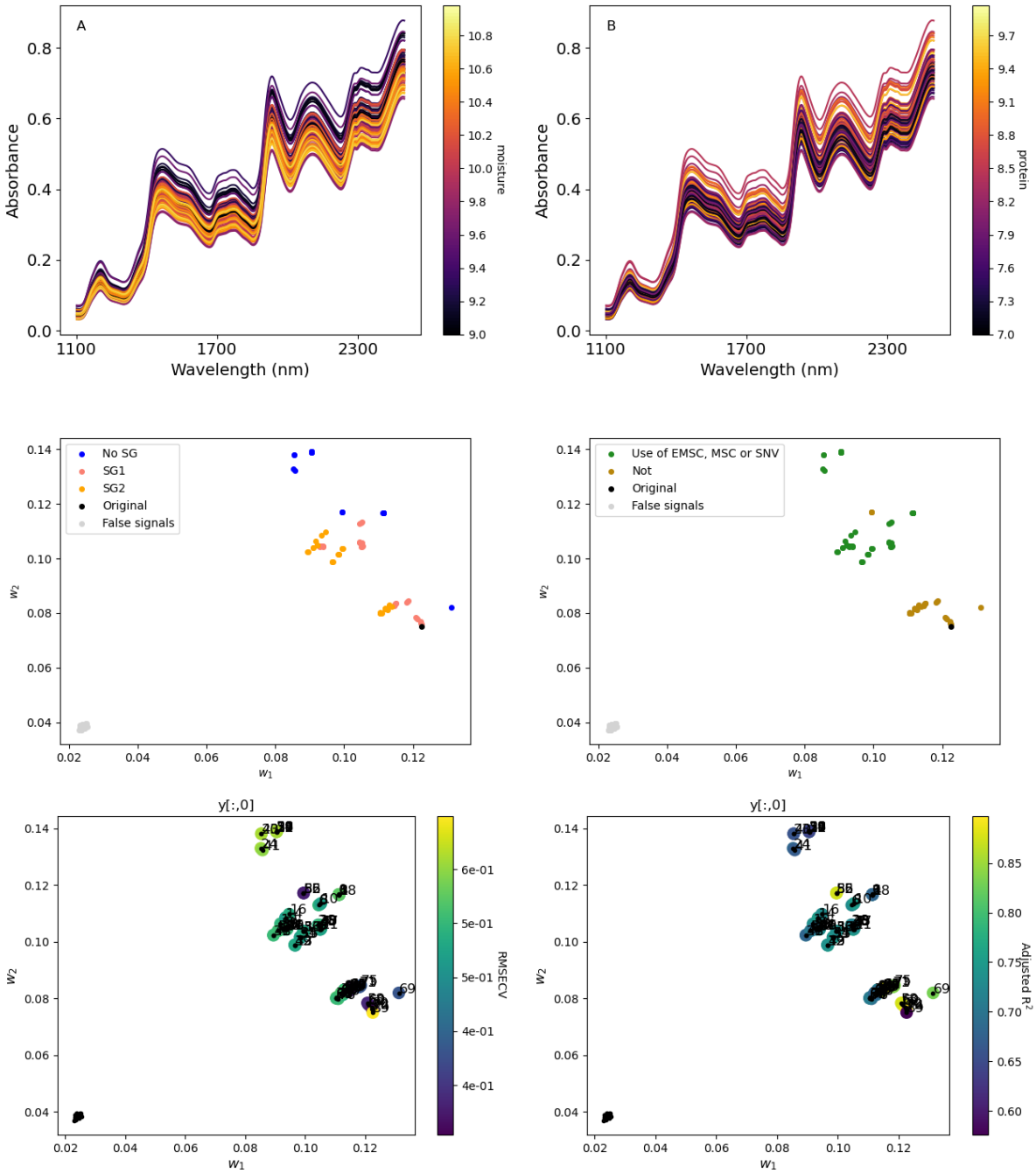
Example 2. Corn dataset

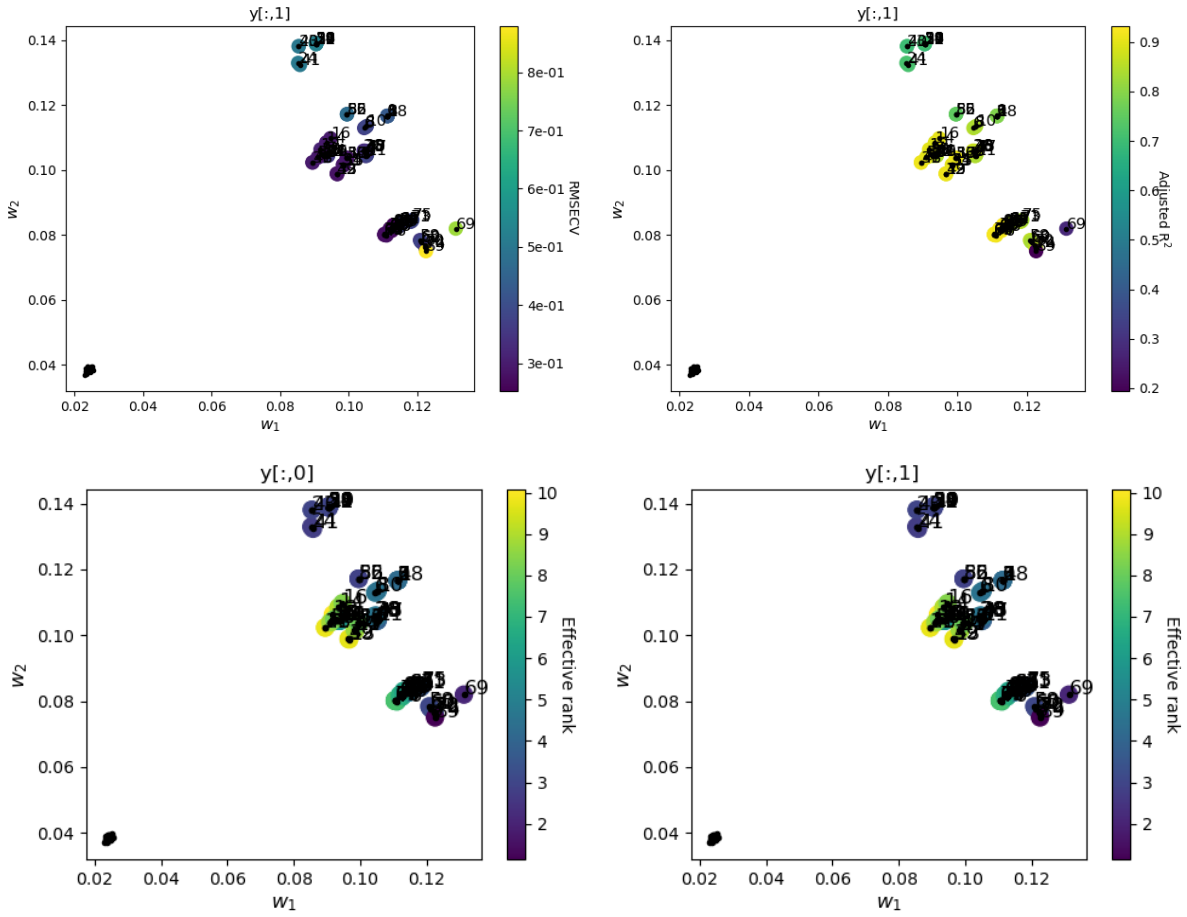
The second dataset used contains spectra of 80 corn samples measured on an NIR spectrometer. The wavelength range is 1100 – 2498 nm, which was acquired at 2 nm intervals, for a total of 700 channels. This dataset is available at *https://www.eigenvector.com/data/Corn/index.html*. We will focus on moisture and protein contents in this study. As previously, six pre-processing techniques are evaluated: baseline, de-trending, EMSC, MSC, SNV and SG.

Spectroscopic data is available in the folder "tests" of the zip file of this pypi module: 'corn.mat'. The code described below is in the file 'S_corn.py'. Function 'fct_loadmat.py' is used to read matlab data.

```
import numpy as np
import matplotlib.pyplot as plt
from fct_loadmat import loadmat
from colorspectra_y import colorspectra_y
import NIR_preprocess as nir_pre
plt.close('all')

#%% Load data
data = loadmat('corn.mat')
m5 = data['m5spec']['data']
y = data['propvals']['data']

# Set y values
y1 = y[:,0][np.newaxis].T # moisture
y2 = y[:,2][np.newaxis].T # protein

y = np.concatenate([y1,y2], axis=1)

# Choose which spectromoter to use
X0 = m5
n, k = X0.shape

#%% Show original spectra
colorspectra_y(X0,y1,y_label='Absorbance', x_label='Wavelength (nm)',
colorbar_label='moisture')
plt.text(0.01,0.85,'A', fontsize=12)
plt.xticks([0, 300,600],[1100, 1700, 2300],fontsize=14)

colorspectra_y(X0,y2,y_label='Absorbance', x_label='Wavelength (nm)',
colorbar_label= 'protein')
plt.text(0.01,0.85,'B', fontsize=12)
plt.xticks([0, 300,600],[1100, 1700, 2300],fontsize=14)

#%% Compare pre-processing
combination, datasets, datasets0, R2_all, R2adj_all, RMSECV_all,
VIP_all, Ef_all, Wt = nir_pre.compare_preprocessing(X0, y, nbPC=3,
nb=60)
```

Output plots




y[:,1]
w_1
w_2
RMSECV
y[:,1]
w_1
w_2
Adjusted R^2
y[:,0]
w_1
w_2
Effective rank
y[:,1]
w_1
w_2
Effective rank