AVANTIX

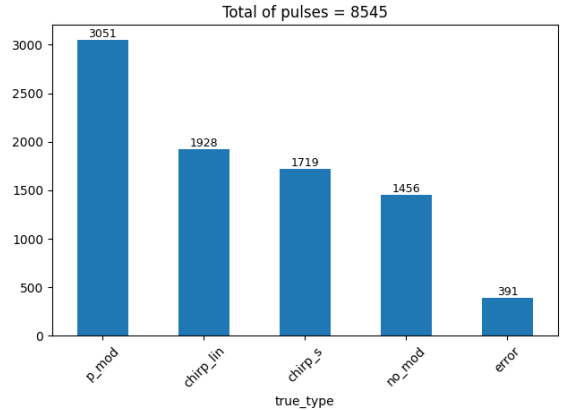
Avancement CBE

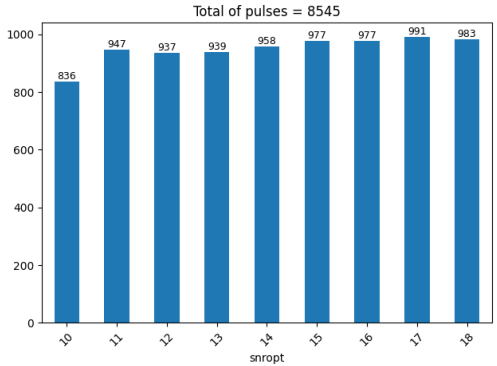
AVANTIX

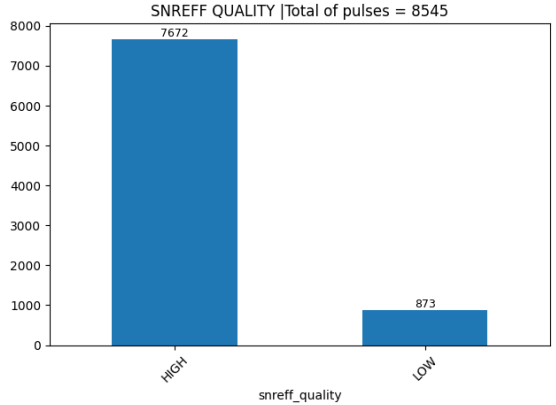
Classification modulation

Jeu de données

9 fichiers /data/SIGINT/FVE/LEGO_FV/long_LEGO_FV_noP124_snropt_{snropt}_iq.pkl
snropt = [10, 18] dB







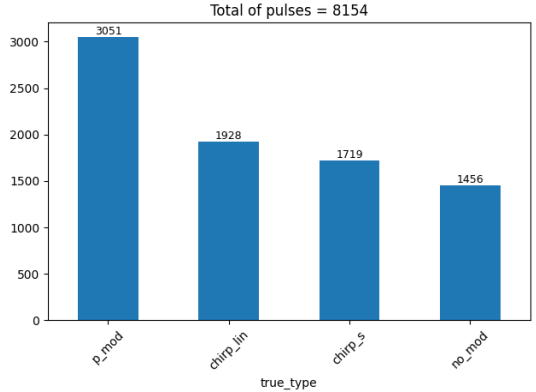
Questions :

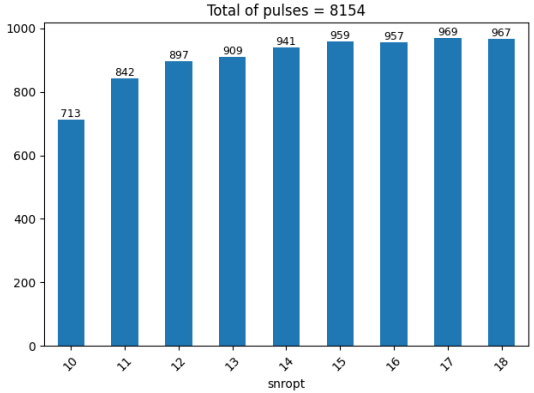
- iq2pdw ne tag jamais en cw ou fmcw → Si di > 1ms alors cw ?
- Gestion des pulses « error » dans phénix : suppression ou « unknown » ?

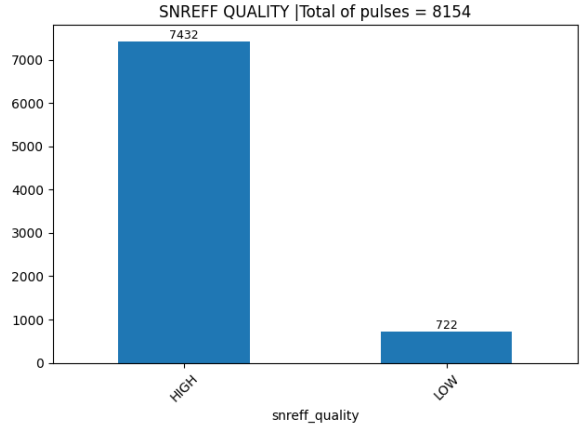
Jeu de données

9 fichiers /data/SIGINT/FVE/LEGO_FV/long_LEGO_FV_noP124_snropt_{snropt}_iq.pkl
snropt = [10, 18] dB

Après suppression des 391 pulses en « error »



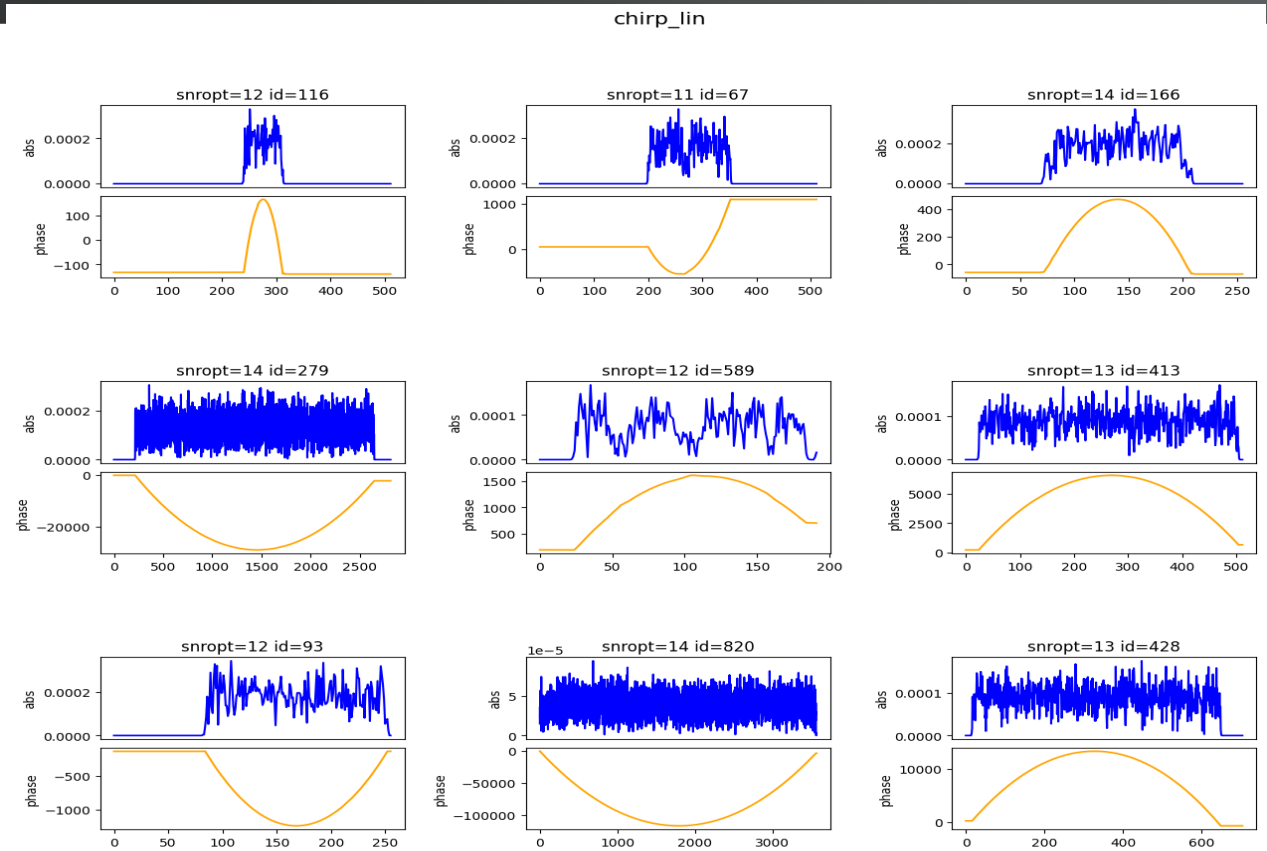




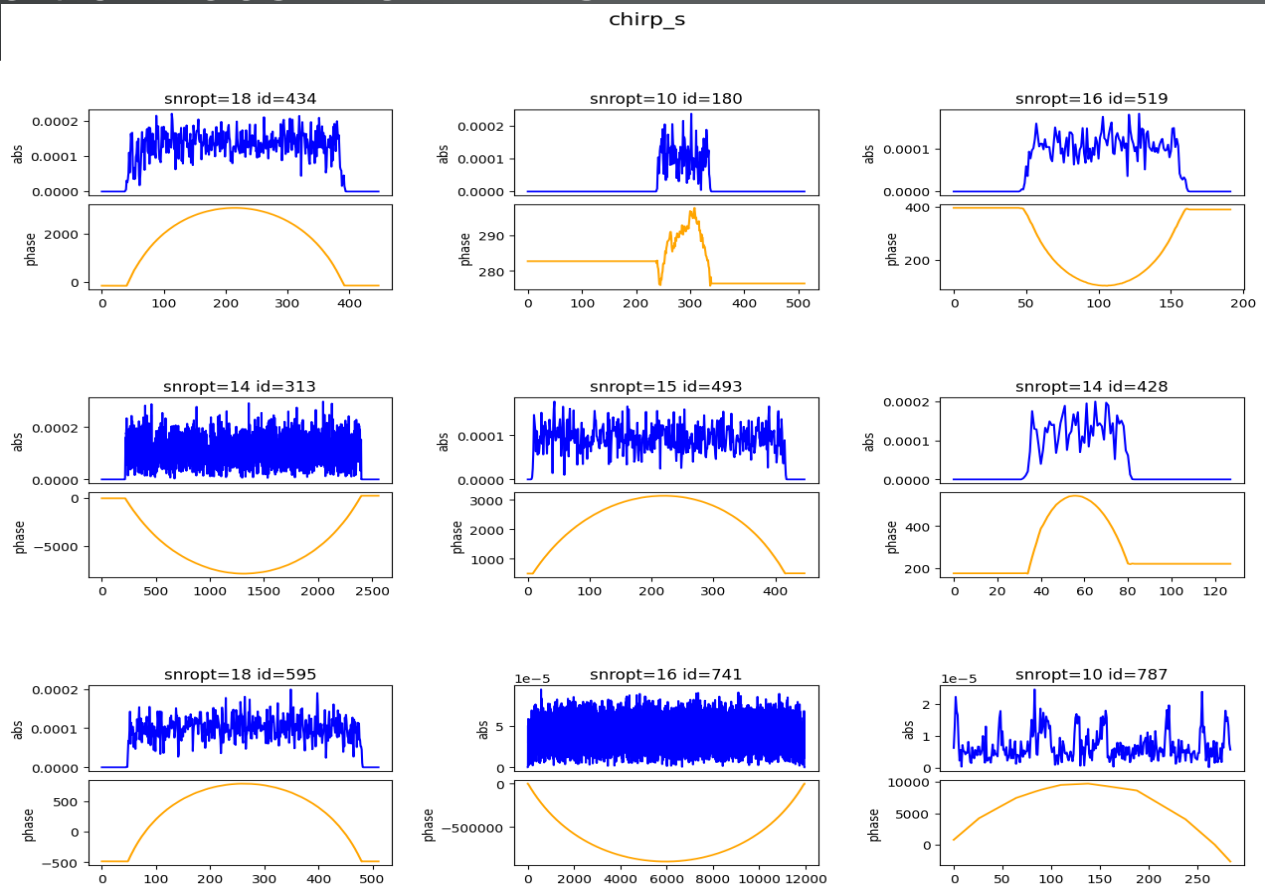
Split Train(80)/Test(20) en fct de snropt et de snreff_quality (via demod vs via tridge)

Train = 6523 pulses| Test = 1631 pulses

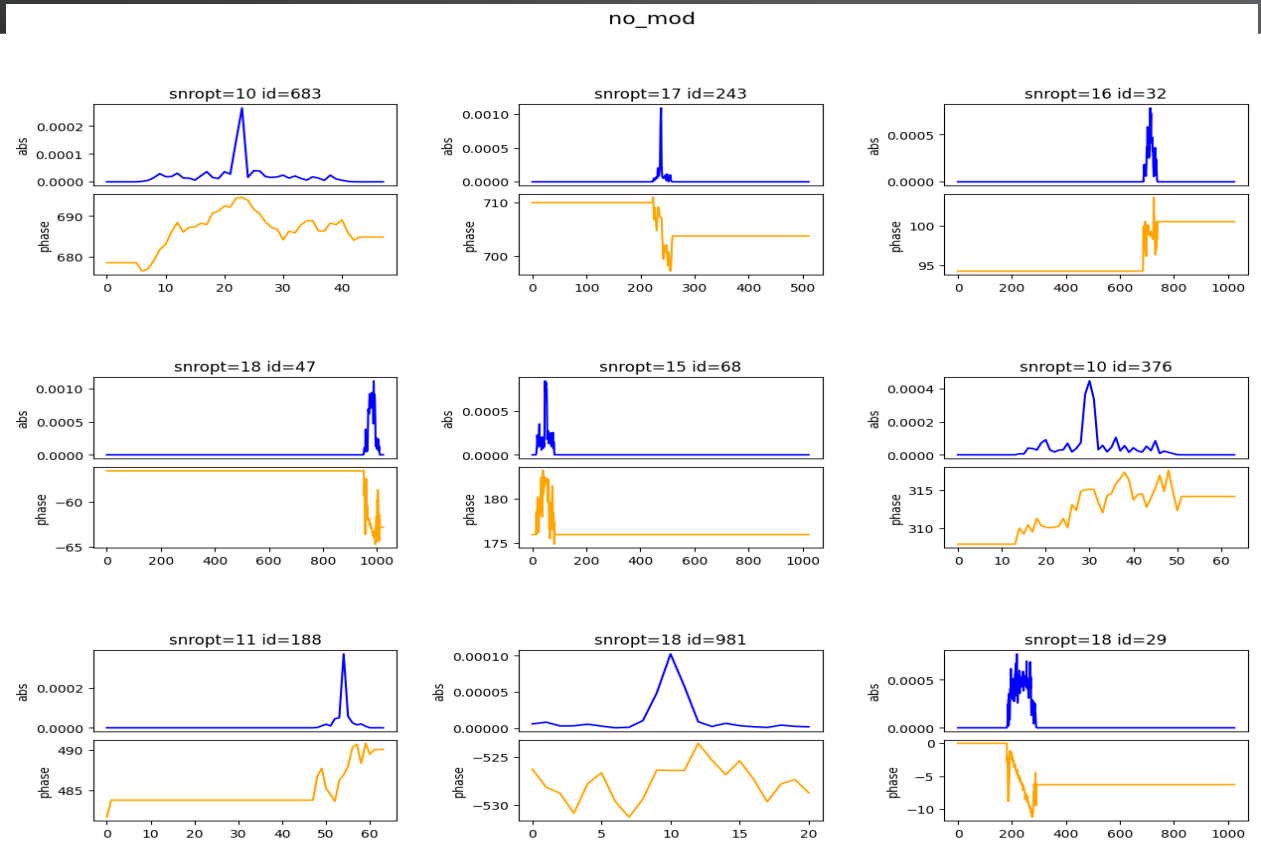
Jeu de données – CHIRP LIN



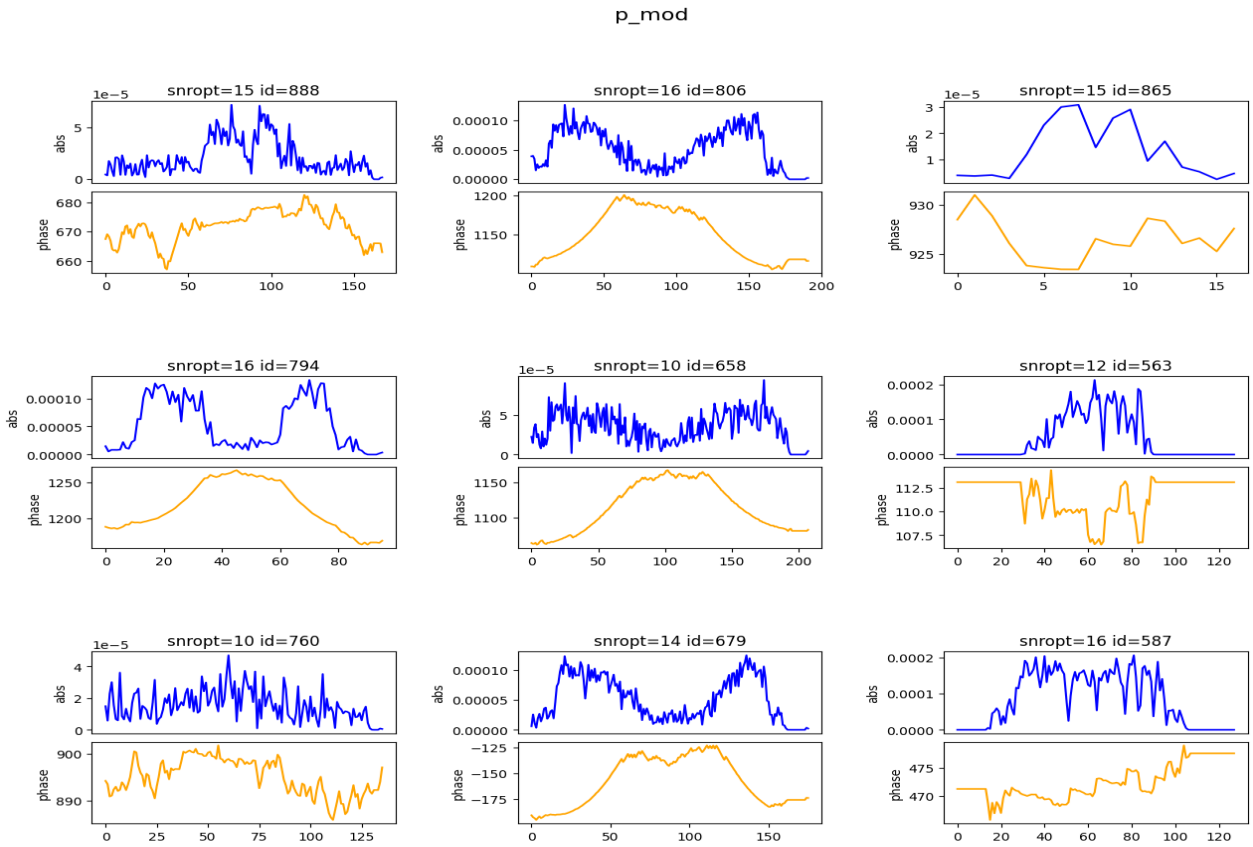
Jeu de données – CHIRP S



Jeu de données – NO MOD



Jeu de données – PMOD



Pré-nettoyage

- ~~Valeur absolue de la phase~~
- Suppression des échantillons où abs==0 en début et fin de signal

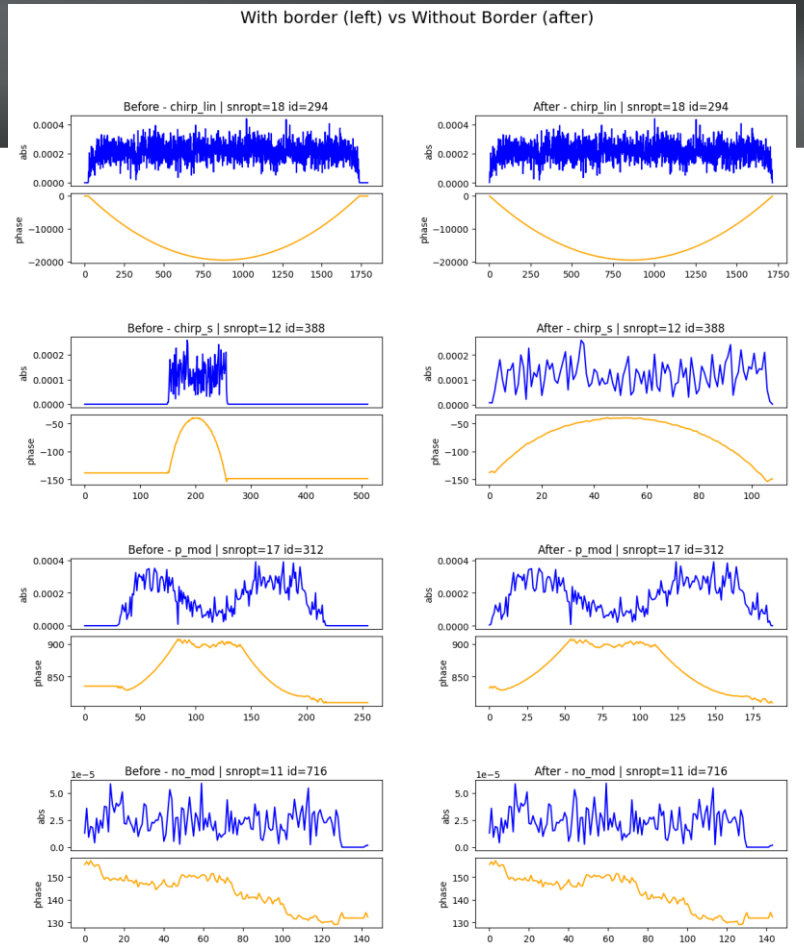
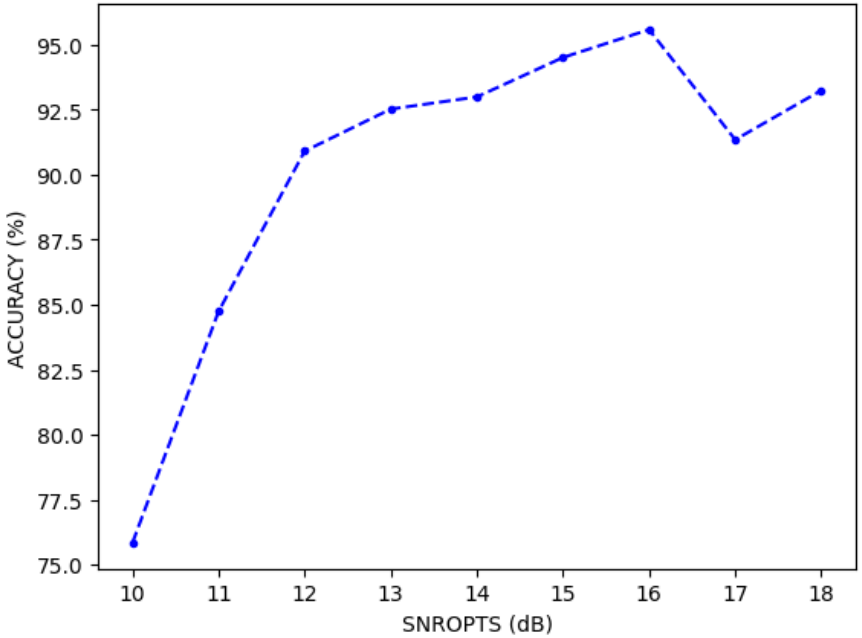
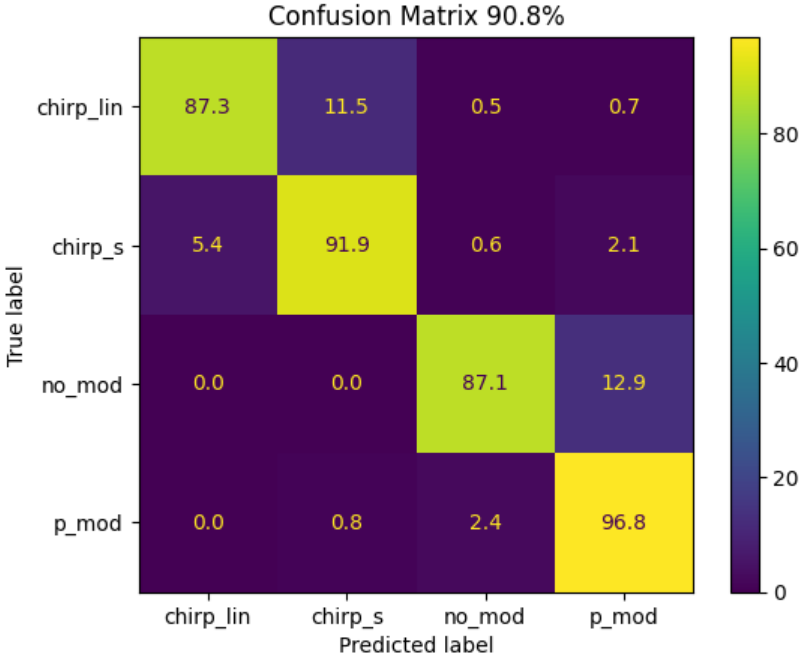


Tableau des performances

Features	Modèle	Acc % on test	Commentaires
TSFRESH Stratégie greedy	RF	90,8	Borne supérieure
FVE	**DecisionTree FVE max-depth = 5 min_impurity_decrease=0.012**	**84,5**	**Deux DecisionTree entrainés par FVE : 1 pour les low et un pour les high snr**
	DecisionTree	TODBG FVE	DecisionTree entrainé par CBE
	DecisionTree max-depth = 5	TODBG FVE	DecisionTree entrainé par CBE
TSFRESH Top 10 A vérifier si ces feats sont rapides à calculer	RF	83,8	L'objectif était de voir la perte en perf en ne conservant que 10 feats pour optimiser le RTR. Etant donnée que c'est moins bien que FVE+DT(md=5) on n'investigue plus
Custom features 'std', 'skewness', 'kurtosis', 'max_value', 'argmax_index', 'zero_crossings', 'mean_value', 'diff_std', 'symmetry_error', 'duration', 'peak_width_half_max', 'slope_asymmetry', 'derivative_energy', 'quad_fit_error', 'second_deriv_mean', 'cusum_slope_change', 'signal_entropy', 'normalized_auc', 'median_position', 'fft_energy_ratio', 'autocorr_lag1', 'dominant_frequency', 'spectral_centroid'	RF	90,4	Afin d'avoir les features importances
	MLP	89,6	RF est meilleur
	RF TOP 10	86,8	Run pour features importances
	DecisionTree Max-depth = 6 TOP 10	**82,8**	**Equivalent aux deux DT de FVE. A voir ce que ça donne si on ajoute l'indice du nperseg**

EfficientFCParameters (1554 cols)+ RandomForest (borne supérieure)





Performances du MLP

Custom features RAW + SNREFF_QUALITY

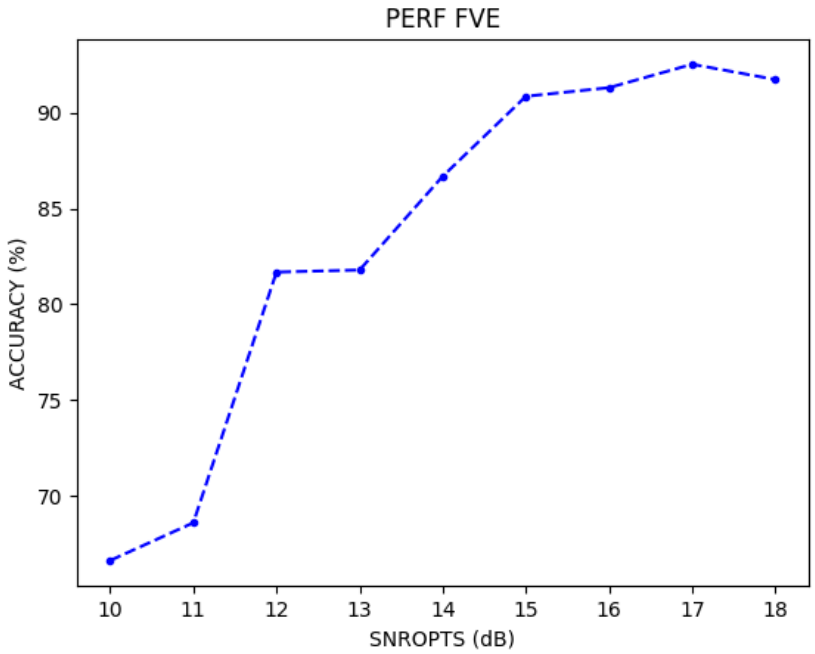
```
Epoch   1 | Train Accuracy: 0.7935 | Test Accuracy: 0.7848
Epoch   2 | Train Accuracy: 0.7811 | Test Accuracy: 0.7781
Epoch   3 | Train Accuracy: 0.8541 | Test Accuracy: 0.8498
Epoch   4 | Train Accuracy: 0.8660 | Test Accuracy: 0.8498
Epoch   5 | Train Accuracy: 0.8767 | Test Accuracy: 0.8645
Epoch   6 | Train Accuracy: 0.8925 | Test Accuracy: 0.8749
Epoch   7 | Train Accuracy: 0.8964 | Test Accuracy: 0.8780
Epoch   8 | Train Accuracy: 0.9002 | Test Accuracy: 0.8670
Epoch   9 | Train Accuracy: 0.9028 | Test Accuracy: 0.8749
Epoch  10 | Train Accuracy: 0.8867 | Test Accuracy: 0.8596
Epoch  11 | Train Accuracy: 0.9085 | Test Accuracy: 0.8853
Epoch  12 | Train Accuracy: 0.9048 | Test Accuracy: 0.8823
Epoch  13 | Train Accuracy: 0.9103 | Test Accuracy: 0.8804
Epoch  14 | Train Accuracy: 0.9112 | Test Accuracy: 0.8811
Epoch  15 | Train Accuracy: 0.9215 | Test Accuracy: 0.8945
Epoch  16 | Train Accuracy: 0.9180 | Test Accuracy: 0.8798
Epoch  17 | Train Accuracy: 0.9246 | Test Accuracy: 0.8841
Epoch  18 | Train Accuracy: 0.9267 | Test Accuracy: 0.8878
Epoch  19 | Train Accuracy: 0.9352 | Test Accuracy: 0.8964
Epoch  20 | Train Accuracy: 0.9131 | Test Accuracy: 0.8792
Epoch  21 | Train Accuracy: 0.9378 | Test Accuracy: 0.8804
Epoch  22 | Train Accuracy: 0.9371 | Test Accuracy: 0.8835
Epoch  23 | Train Accuracy: 0.9365 | Test Accuracy: 0.8884
Epoch  24 | Train Accuracy: 0.9462 | Test Accuracy: 0.8921
Epoch  25 | Train Accuracy: 0.9473 | Test Accuracy: 0.8847
Epoch  26 | Train Accuracy: 0.9503 | Test Accuracy: 0.8890
Epoch  27 | Train Accuracy: 0.9535 | Test Accuracy: 0.8952
Epoch  28 | Train Accuracy: 0.9555 | Test Accuracy: 0.8915
Epoch  29 | Train Accuracy: 0.9578 | Test Accuracy: 0.8903
```

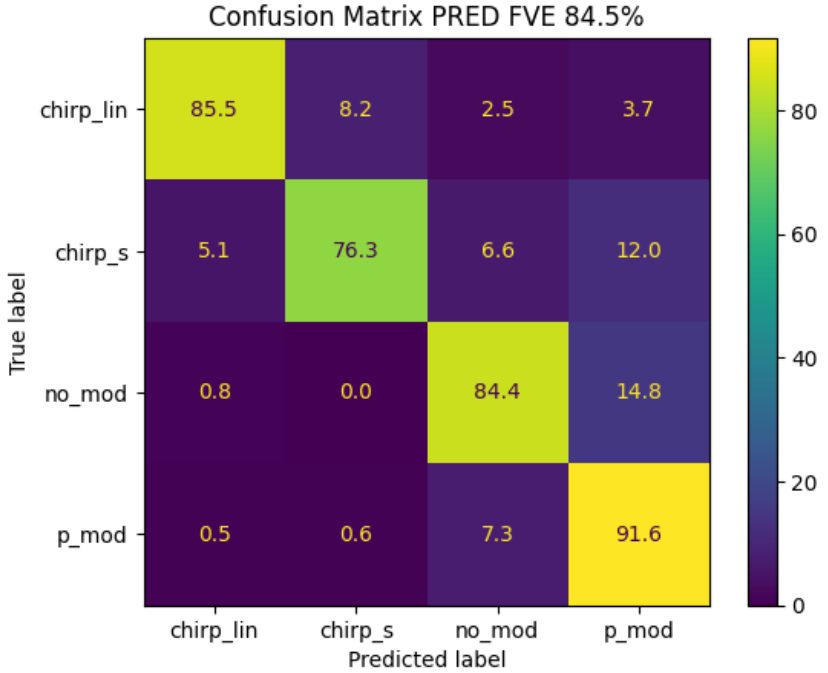
```python
class MLP(nn.Module):
    def __init__(self, input_dim=40, num_classes=4):
        super(MLP, self).__init__()
        self.fc1 = nn.Linear(input_dim, 512)
        self.fc2 = nn.Linear(512, 256)
        self.fc3 = nn.Linear(256, 128)
        self.fc4 = nn.Linear(128, 64)
        self.fc5 = nn.Linear(64, num_classes)

    def forward(self, x):
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = F.relu(self.fc3(x))
        x = F.relu(self.fc4(x))
        x = self.fc5(x)
        return x
```

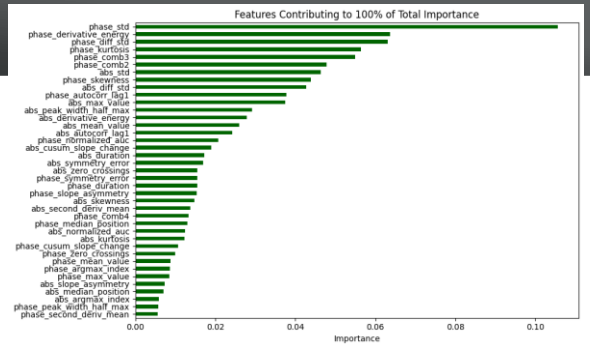
Moins bon que RF (90,4)

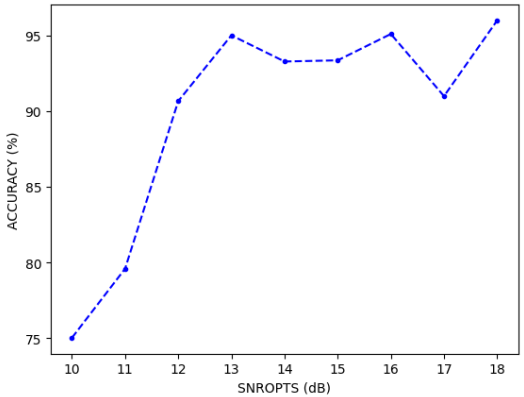
Perf FVE - Baseline

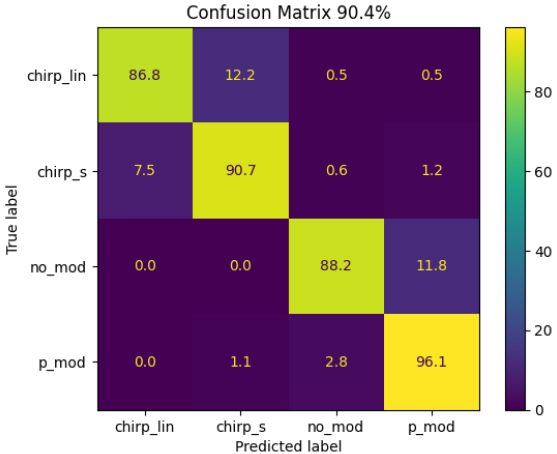




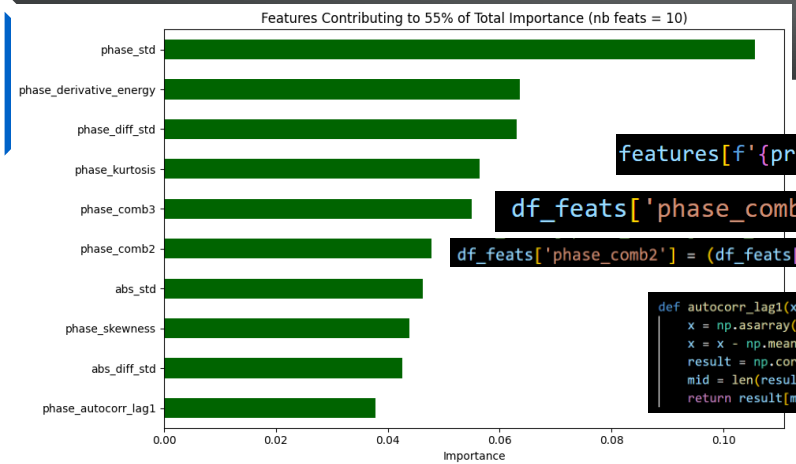
Custom features (39 cols)+ RandomForest







Custom features TOP 10 + RandomForest



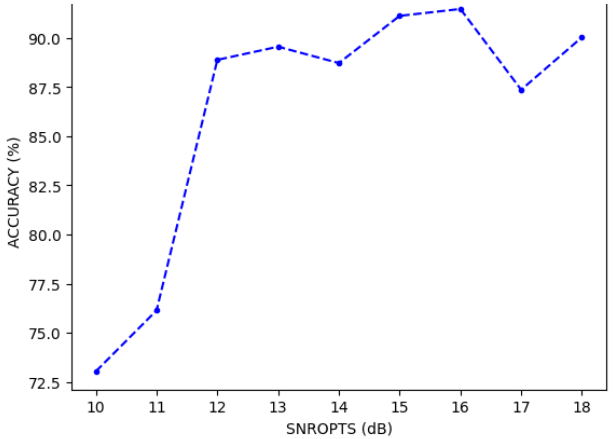

```python
def derivative_energy(x):
    dx = np.diff(x)
    return np.sum(dx**2) / len(dx)
```

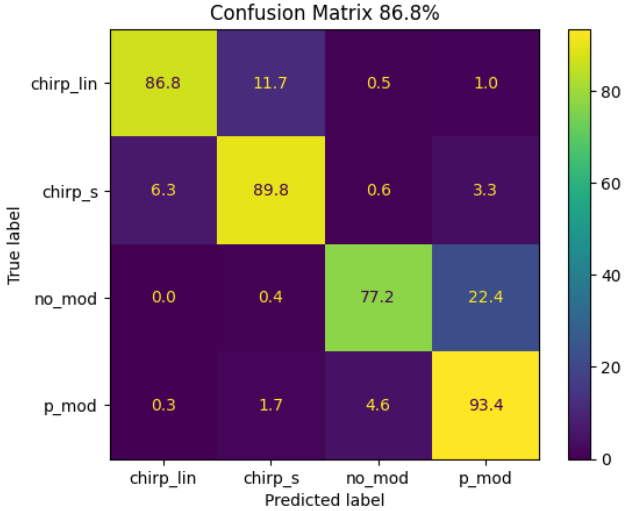
```python
features[f'{prefix}_diff_std'].append(np.std(np.diff(x)))
```

```python
df_feats['phase_comb3'] = df_feats['phase_derivative_energy'] * df_feats['phase_kurtosis']
```

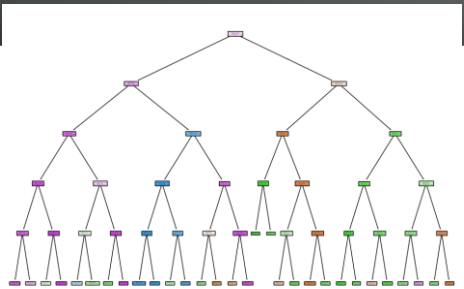
```python
df_feats['phase_comb2'] = (df_feats['phase_max_value'] - df_feats['phase_mean_value']) / (df_feats['phase_std'] + 1e-12)
```

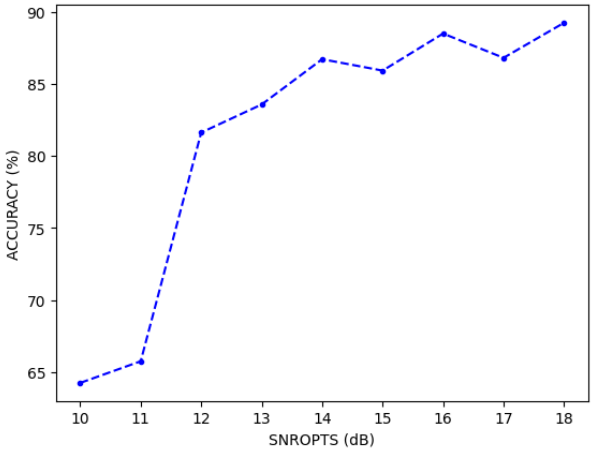
```python
def autocorr_lag1(x):
    x = np.asarray(x)
    x = x - np.mean(x)
    result = np.correlate(x, x, mode='full')
    mid = len(result) // 2
    return result[mid + 1] / result[mid] if result[mid] != 0 else 0
```

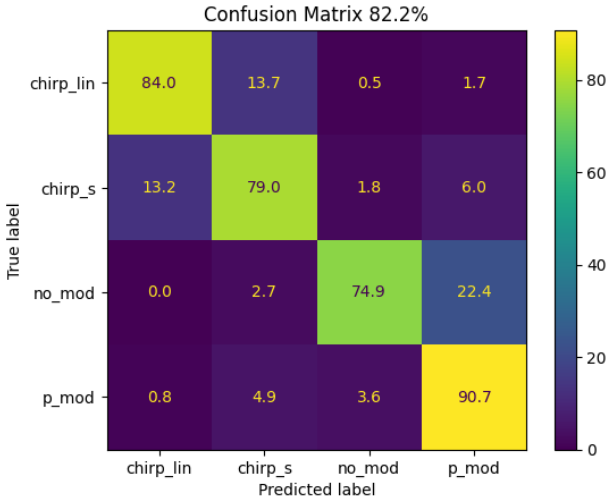




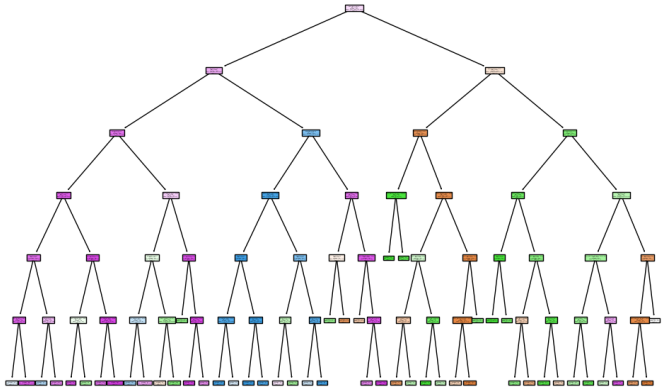
Custom features TOP 10 + DT max-depth = 5

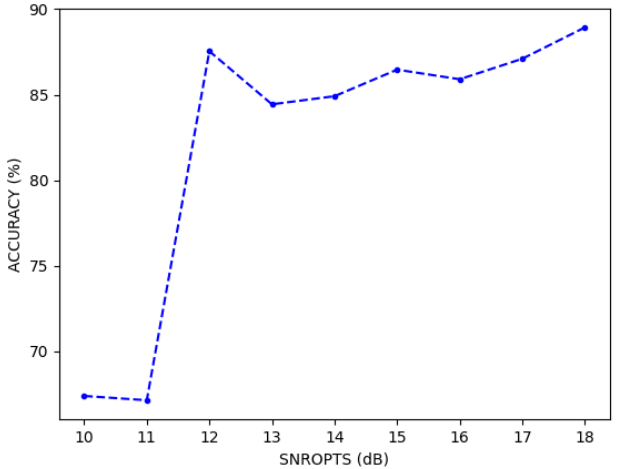






Custom features TOP 10 + DT max-depth = 6





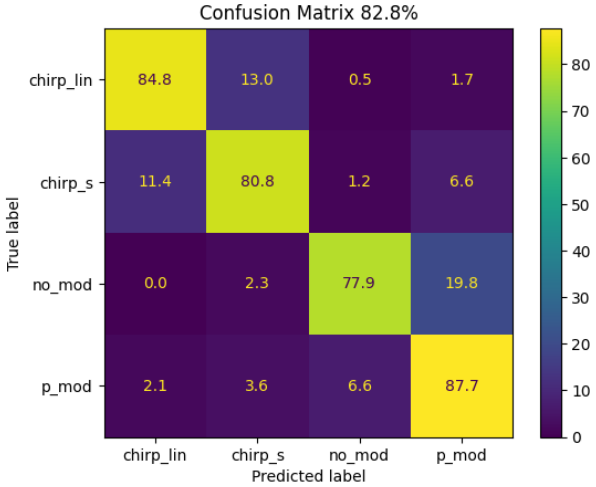
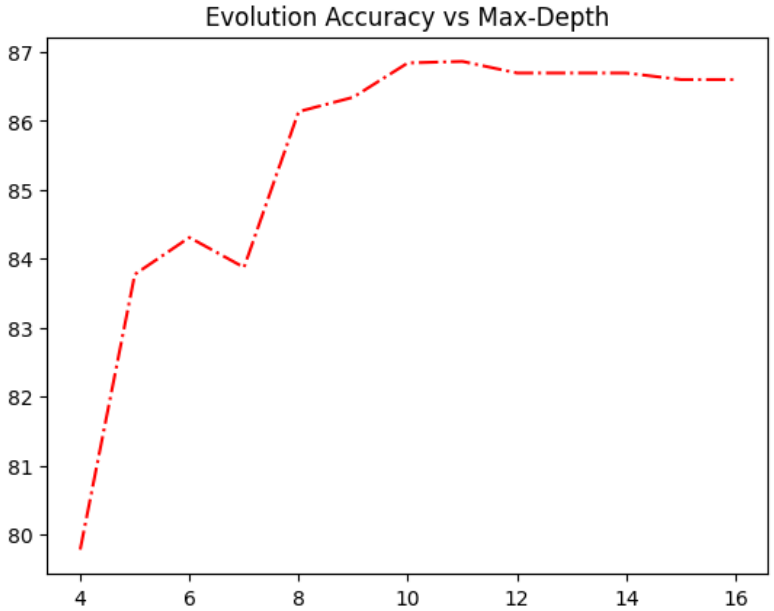


Tableau des performances

40 Features	Modèle	Acc % on test
FVE	DecisionTree FVE max-depth = 5 min_impurity_decrease=0.012	84,5
	DecisionTree Max-depth = 10	TODO
Custom features RAW + SNREFF_QUALITY 'std', 'skewness', 'kurtosis', 'max_value', 'argmax_index', 'zero_crossings', 'mean_value', 'diff_std', 'symmetry_error', 'duration', 'peak_width_half_max', 'slope_asymmetry', 'derivative_energy', 'quad_fit_error', 'second_deriv_mean', 'cusum_slope_change', 'signal_entropy', 'normalized_auc', 'median_position', 'fft_energy_ratio', 'autocorr_lag1', 'dominant_frequency', 'spectral_centroid'	DecisionTree Max-depth = 6	84,2
	DecisionTree Max-depth = 10	86,8

Performances en fct du MAX-DEPTH

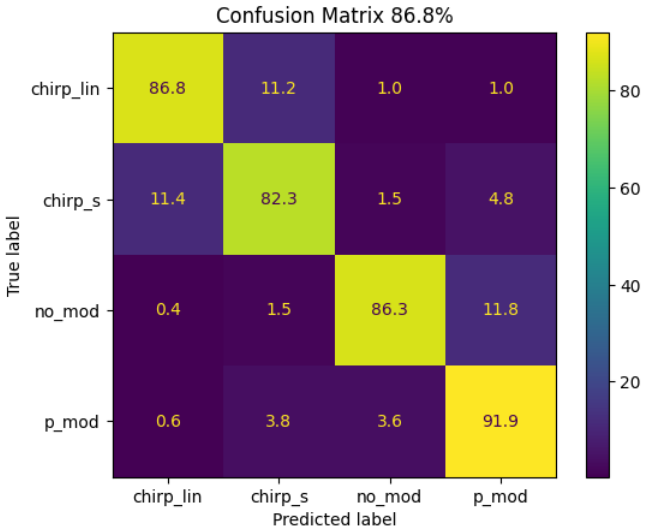
Custom features RAW + SNREFF_QUALITY en fct du max-depth

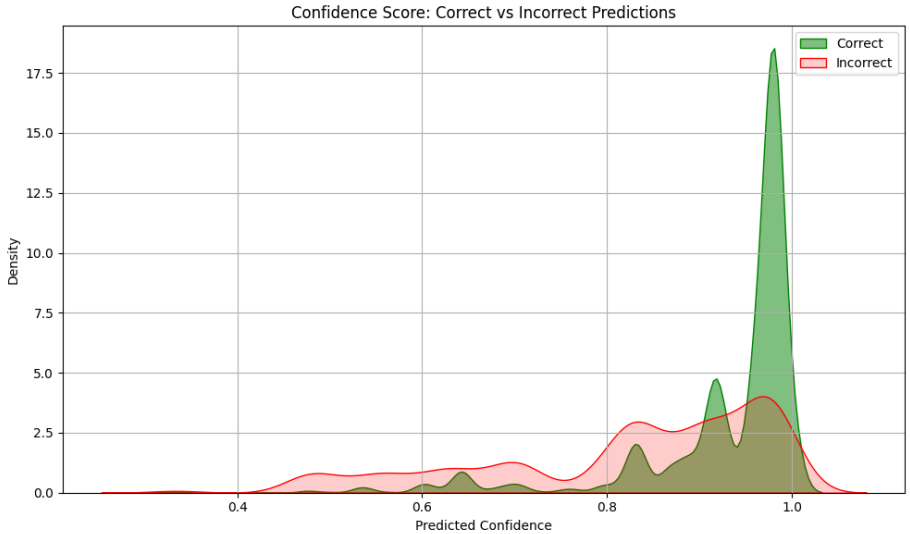


On gagne à passer à max-depth = 10 sans surcout en temps d'inférence car la complexité est O(max-depth)

Analyse des erreurs

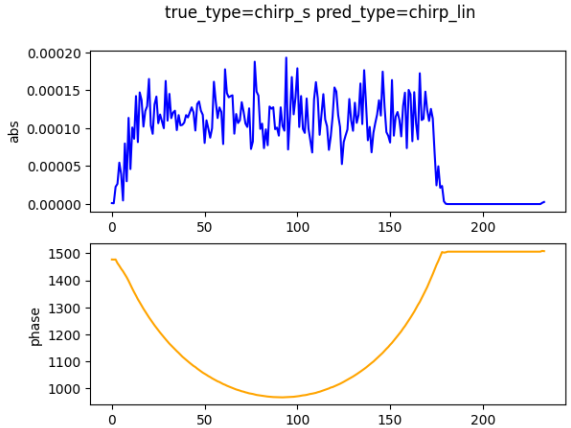
Custom features RAW + SNREFF_QUALITY à max-depth = 10
Accuracy = 86,8%

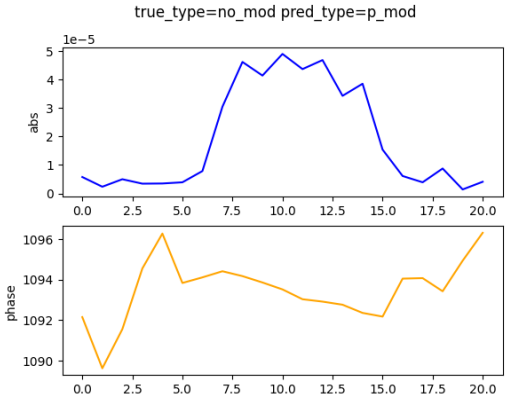


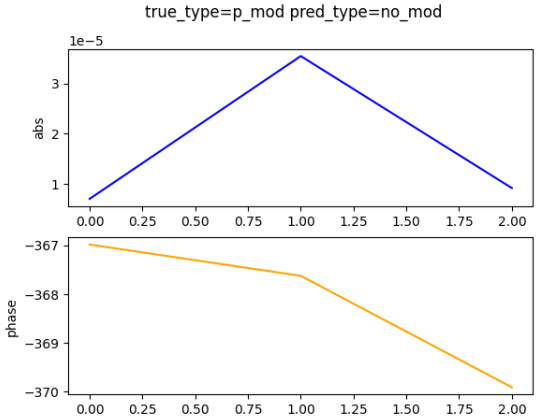


Analyse des erreurs : exemples

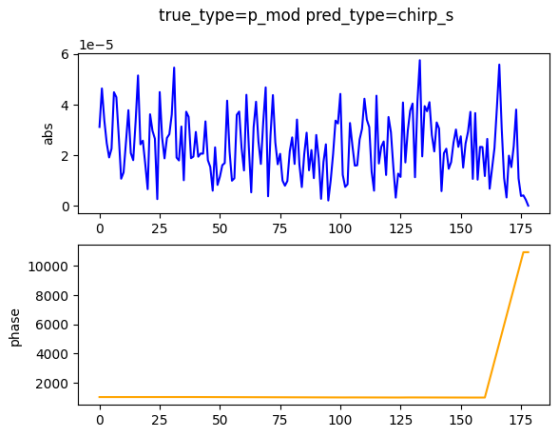
Custom features RAW + SNREFF_QUALITY à max-depth = 10
Accuracy = 86,8%
199 impulsions mal classées / 1631

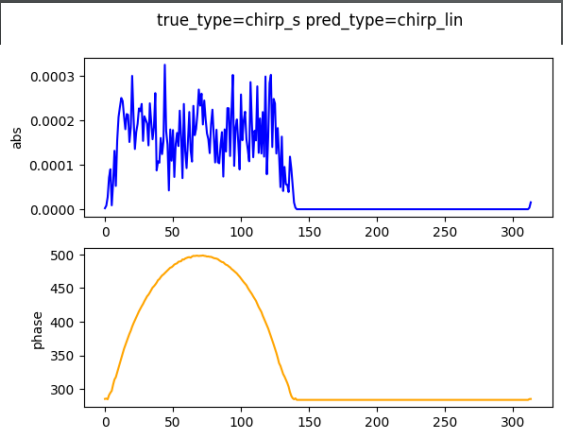


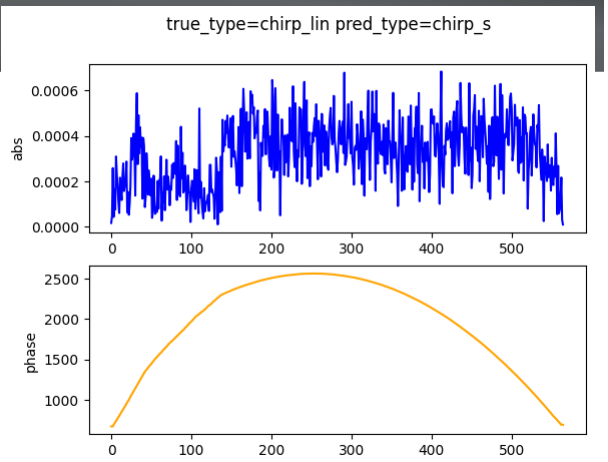


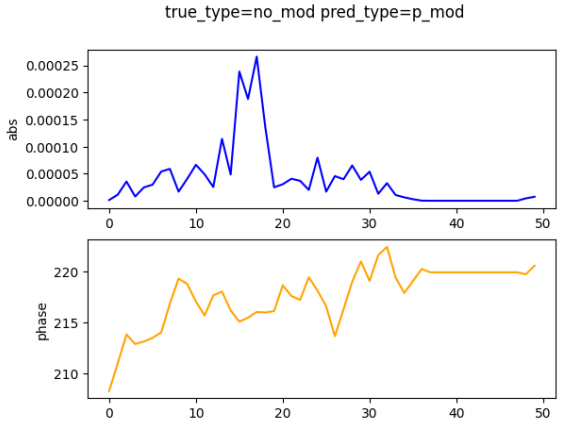


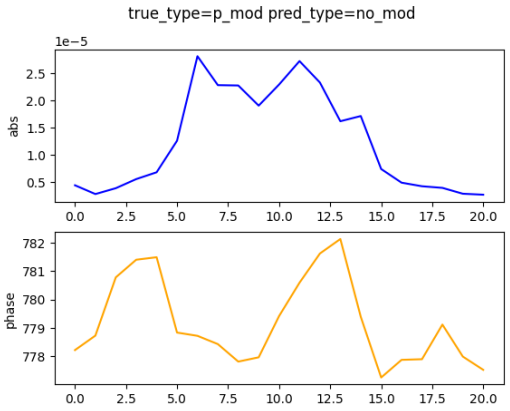
Analyse des erreurs : exemples

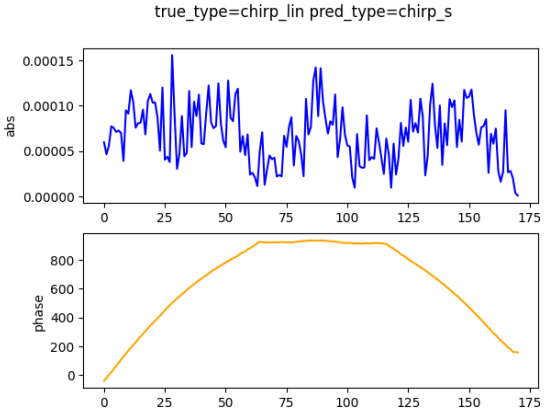




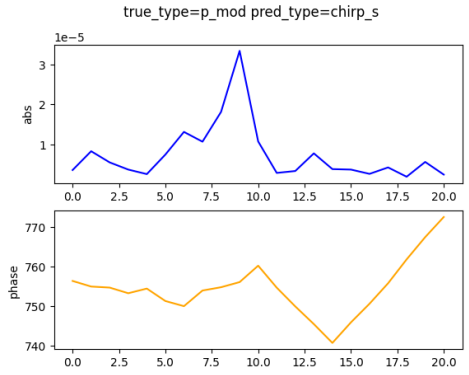


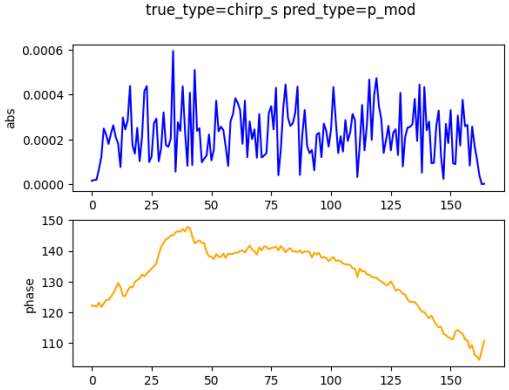


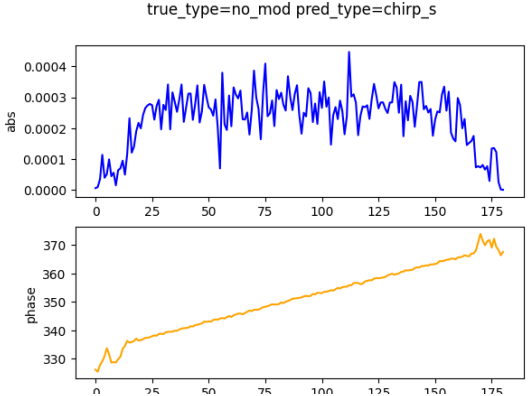


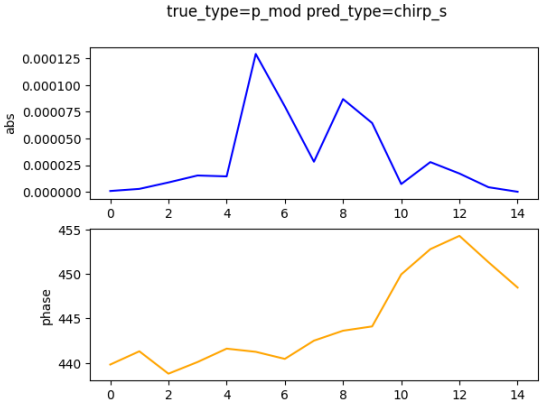


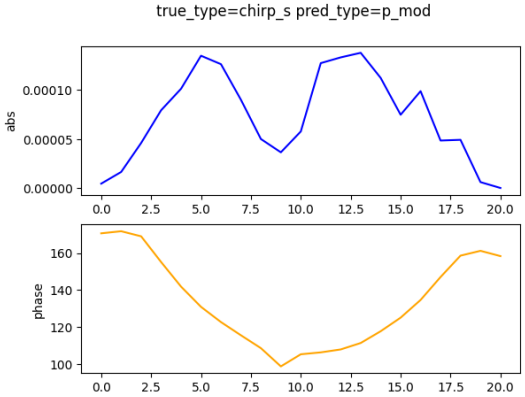
Analyse des erreurs : exemples

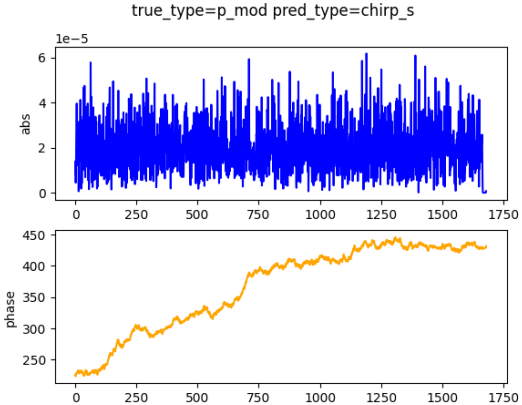




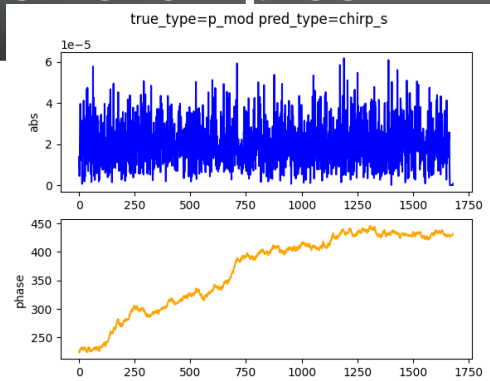




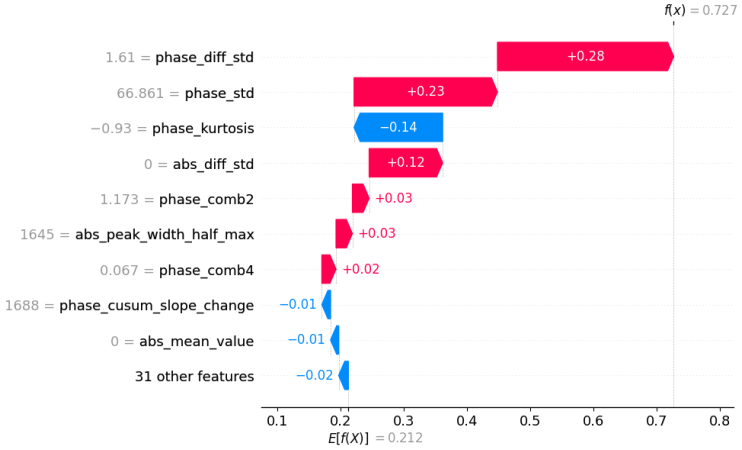




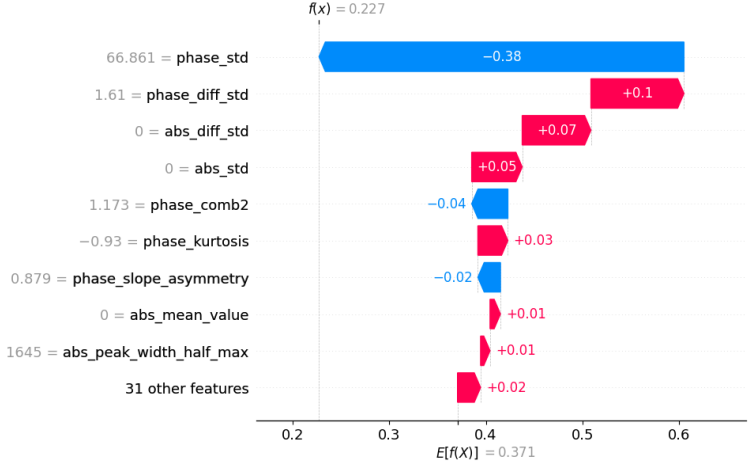
Analyse des erreurs : exemples



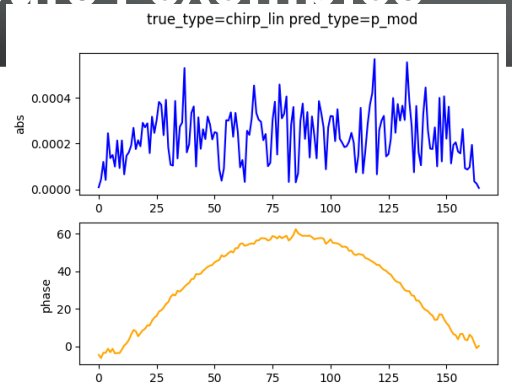
Explaination on pred_type choosed by model



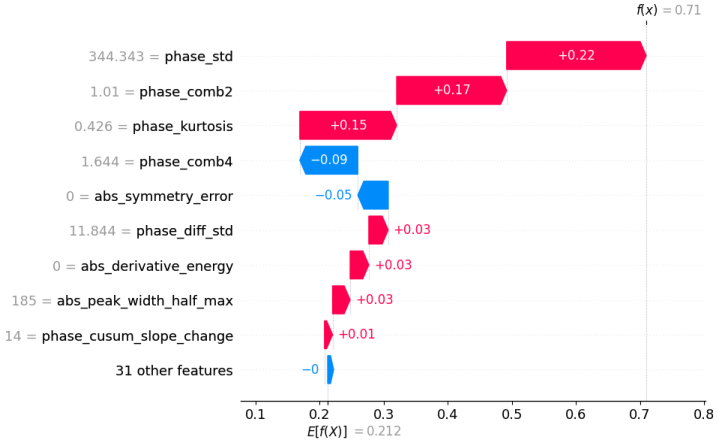
Explaination on true_type not choosed by model



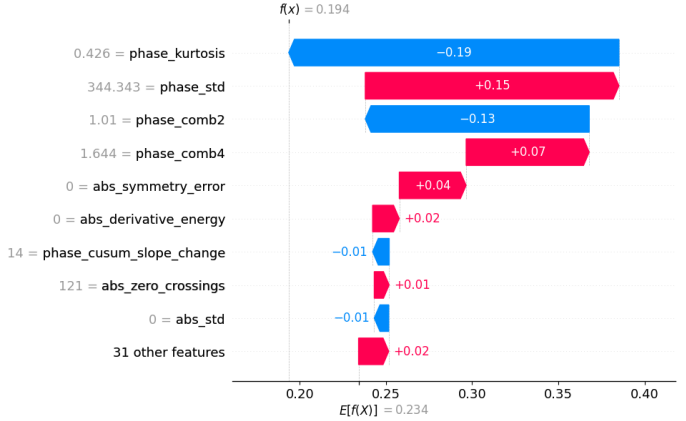
Analyse des erreurs : exemples



Explaination on pred_type choosed by model



Explaination on true_type not choosed by model



AVANTIX

IQ2PDW CPU plugin PHX2

Travaux

- Gestion des fs/NPERSEGS
- Plugin PHX

List des NPERSEGS

```python
def get_NPERSEGS(fs):
    NPERSEGS_4GS = [64, 128, 256, 512, 1024, 4096, 16384, 131072]
    nperseg_min = 8
    def nearest_power_of_2(x):
        return  2 ** round(torch.math.log2(x)) if x > 0 else 0

    NPERSEGS = [nearest_power_of_2(int(nperseg * fs / 4e9)) for nperseg in NPERSEGS_4GS]
    NPERSEGS = [x for x in NPERSEGS if x >= nperseg_min]

    return NPERSEGS
```

fs	NPERSEGS
100 MSps	[8, 16, 32, 128, 512, 4096]
500 MSps	[8, 16, 32, 64, 128, 512, 2048, 16384]
1 GSps	[16, 32, 64, 128, 256, 1024, 4096, 32768]
2 GSps	[32, 64, 128, 256, 512, 2048, 8192, 65536]
4 GSps	[64, 128, 256, 512, 1024, 4096, 16384, 131072]
244 MSps	[8, 16, 32, 64, 256, 1024, 8192]
4 MHz	[16, 128]

AVANTIX

MERCI