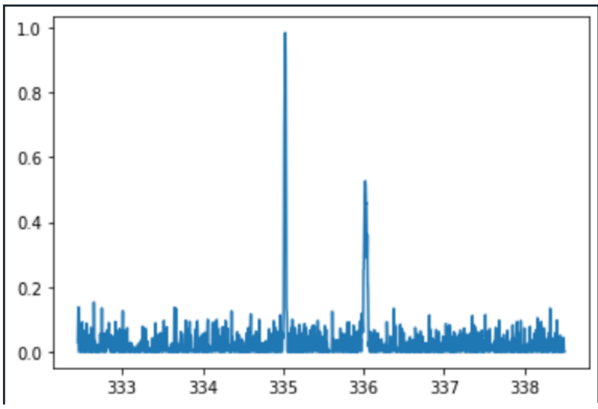
Étant donné une courbe paramétrique définie par (x(t), y(t)), la courbure κ en un point de la courbe est donnée par :

$$k = \frac{|x'(t)y''(t) - y'(t)x''(t)|}{(x'(t)^2 + y'(t)^2)^{3/2}}$$

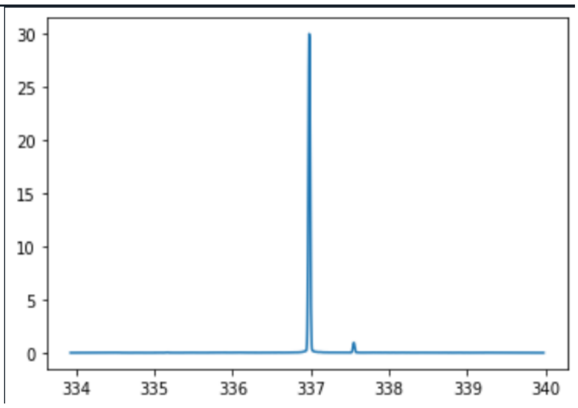
Soit $x(t) = \log(||\ s(t)\ ||_2)$ et $y(t) = \log(||\ m - b*s(t)\ ||_2)$ et $t = \log(\text{itérations})$

Processus de simulation et déconvolution

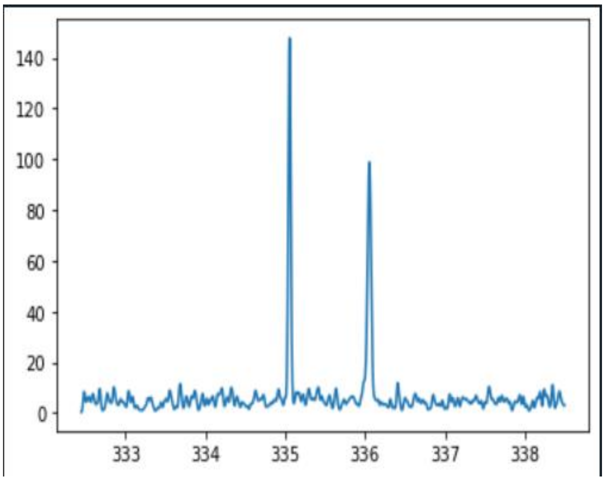
Si l'on prend un signal original simulé, on le convole avec la PSF pour obtenir le signal mesuré simulé, dans le but d'appliquer l'algorithme de Richardson-Lucy afin de récupérer le signal optimal et de le comparer avec le signal original simulé.



(a)



(b)

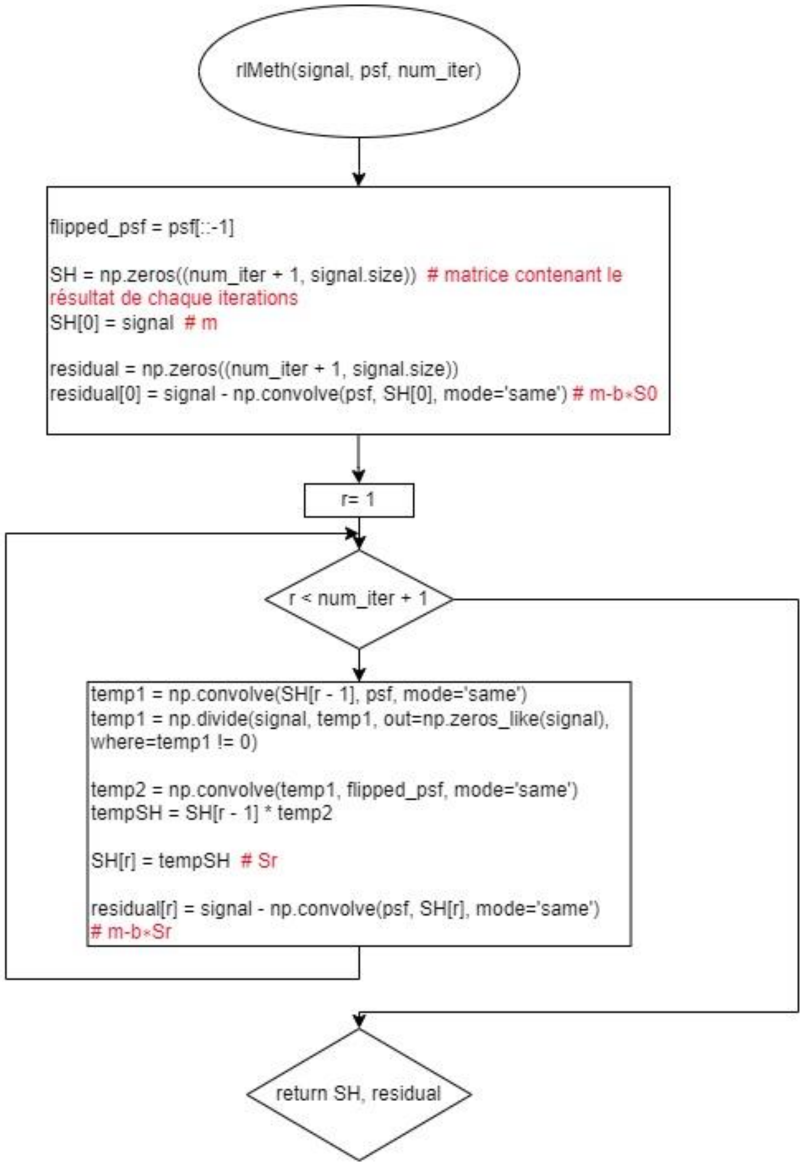


(c)

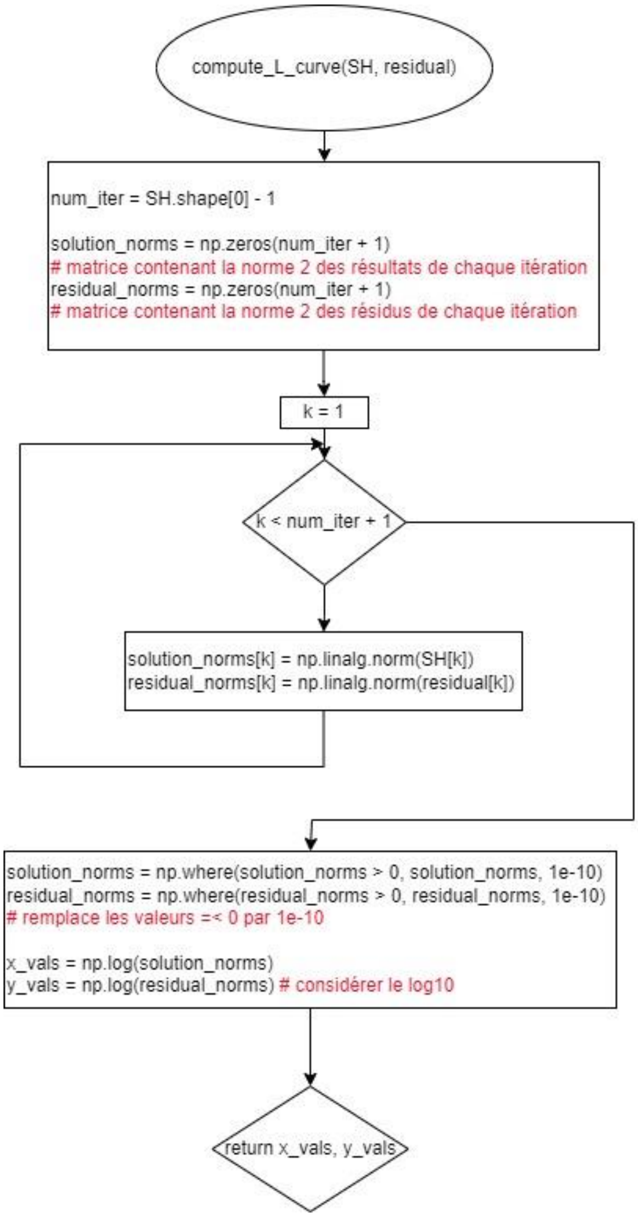
(a) : signal original simulé

(b) : la fonction du psf

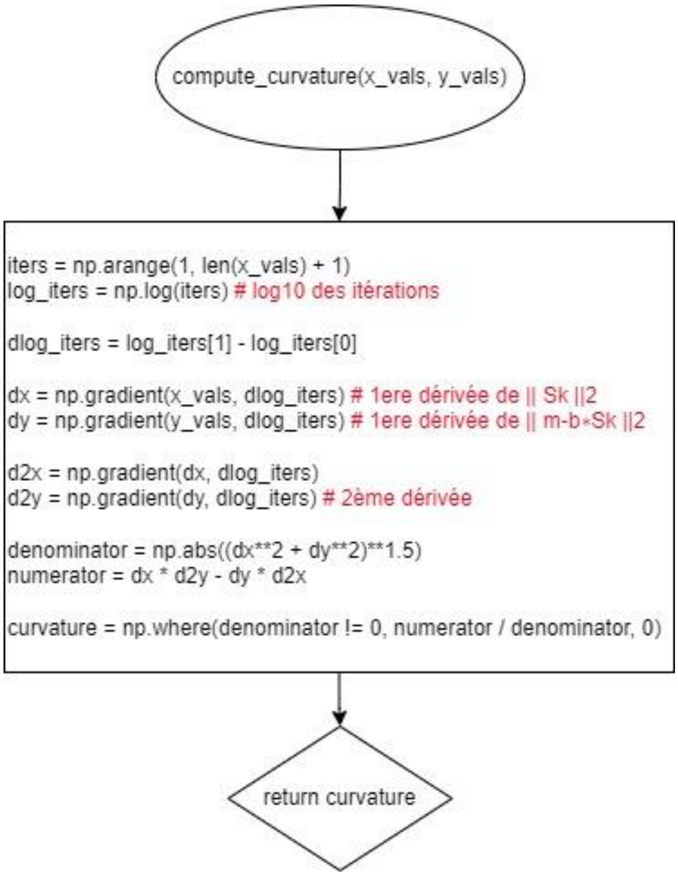
(c) : signal mesuré simulé



La fonction **rlMeth** prend en argument le signal mesuré, la psf et le nombre d'itérations, et retourne un tableau **SH** qui contient le résultat de l'estimation à toutes les itérations, ainsi qu'un tableau **residual** qui contient les résidus m - b∗s(t) à toutes les itérations.



La fonction **compute_L_curve** prend en arguments les tableaux **SH** et **residual**, et retourne les tableaux **x_vals**, **y_vals** qui contiennent le logarithme de la norme $\mathbf{L^2}$ des estimations et des résidus à toutes les itérations.



La fonction **compute_curvature** prend en arguments les tableaux **x_vals** et **y_vals**, et retourne **curvature**, la fonction de courbure présentée précédemment.

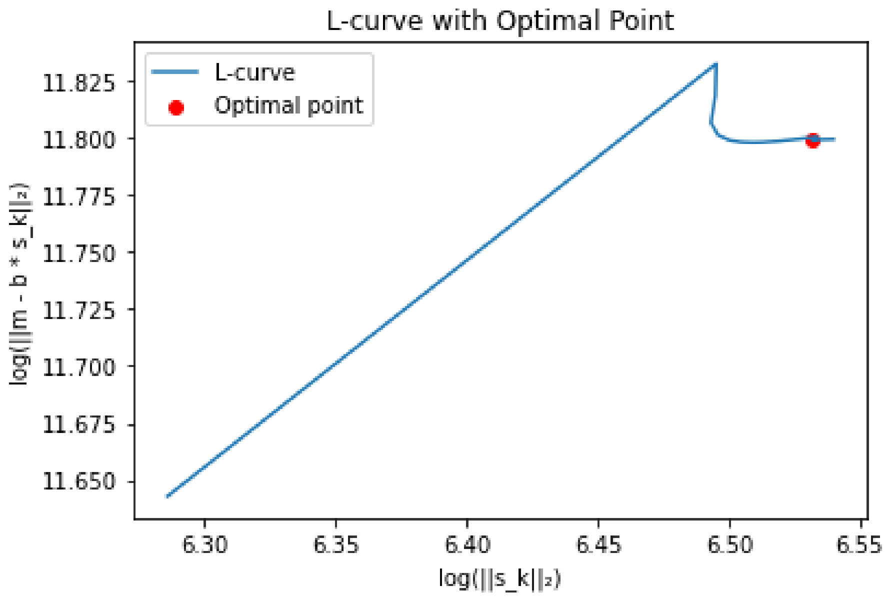
main

```
psf_path = './data/kernel'
psf_files = sorted(glob.glob(psf_path + "/*.csv")) # importation du psf
orig_path = './data/original'
orig_files = glob.glob(orig_path + "/*.csv") # importation d'un signal réel simulé

# Choose which file to use
file_index = 0

# Load kernel and measurement data
psf_data = np.loadtxt(psf_files[file_index], delimiter=';', skiprows=63)
orig_data = np.loadtxt(orig_files[file_index], delimiter=';', skiprows=61)


lb, b, lM, M = fix_inputs(psf_data[0, :], psf_data[1, :], orig_data[0, :], orig_data[1, :])

mes_sim = np.convolve(b, M, mode='same') # convolution du signal réel simulé avec le psf
pour obtenir le signal mesuré

num_iterations = 1500
SH, residual = rlMeth(signal=mes_sim, psf=b, num_iter=num_iterations, autostop=False)

x_vals, y_vals = compute_L_curve(SH, residual) # Calcul L-curve data

curvatures = compute_curvature(x_vals, y_vals) # calcul de la courbure du L-curve

optimal_index = np.argmax(curvatures)
optimal_iteration = optimal_index + 1  # iteration optimal (maximum curvature)


optimal_spectrum = SH[optimal_iteration]  # signal estimé optimal

error_signal_2 = np.linalg.norm(optimal_spectrum - M) # || S - Soptimal ||2

relative_error_signal_2 = np.linalg.norm(optimal_spectrum - M) / np.linalg.norm(M)
# || Soptimal - S ||2 / || S ||2
```

Le reste du code charge les données du signal original simulé et de la psf, effectue la convolution entre le signal original simulé et la psf pour récupérer le signal mesuré simulé, et extrait **optimal_itération**, qui correspond au maximum du **curvature**, ainsi qu'**optimal_spectrum**, le signal à l'itération optimal.

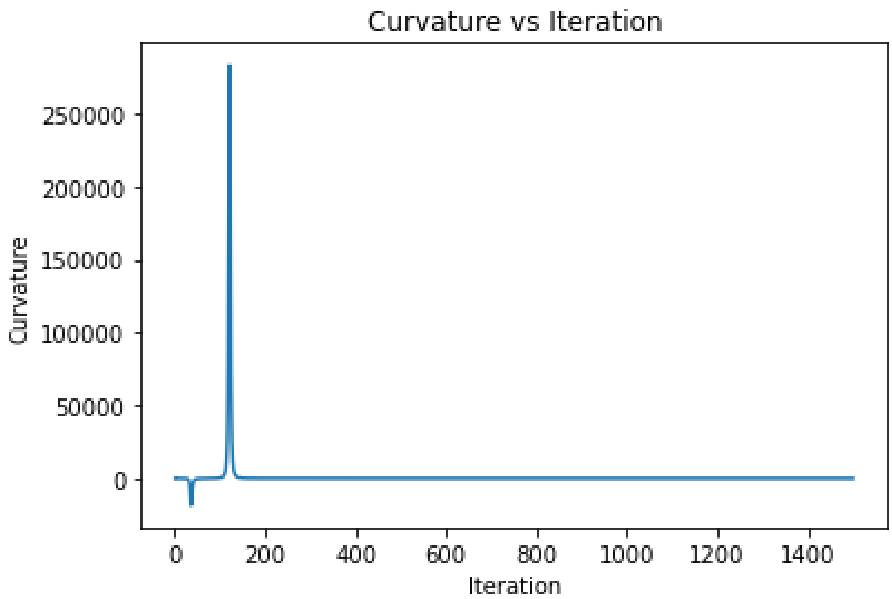
Le code calcule également **error_signal_2**, la norme **L^2** de la différence entre le signal optimal et le signal original simulé, ainsi que **relative_error_signal_2**.

Tracé du L-curve

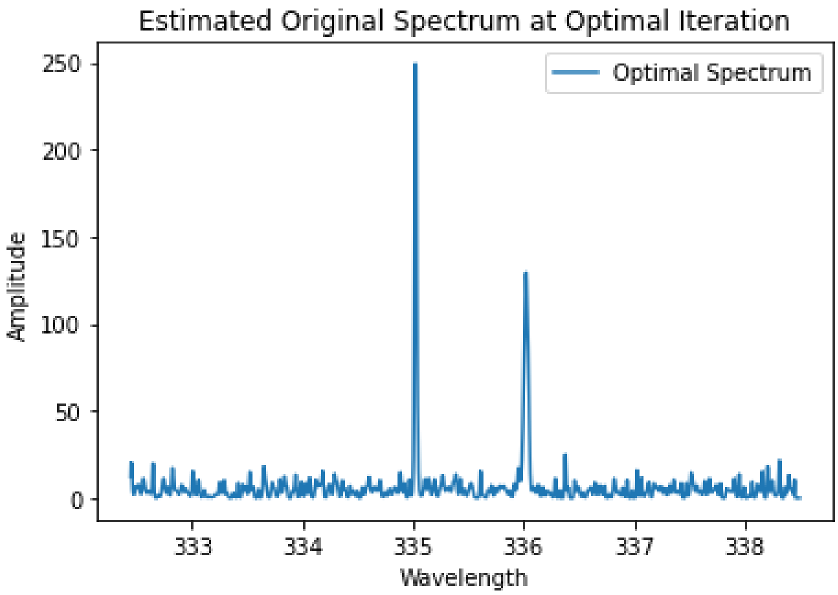


Pourquoi elle a cette allure ?

Tracé de la Courbure vs itération



Signal estimé à l'itération optimal



Pics irréalistes

```
In [2]: optimal_iteration
Out[2]: 121

In [3]: error_signal_2
Out[3]: 683.7509544790001

In [4]: relative_error_signal_2
Out[4]: 235.41837018069168
```

Différence avec l'approche d'eduarda

Dans la formule de la courbure, il a été considéré que y = f(x)

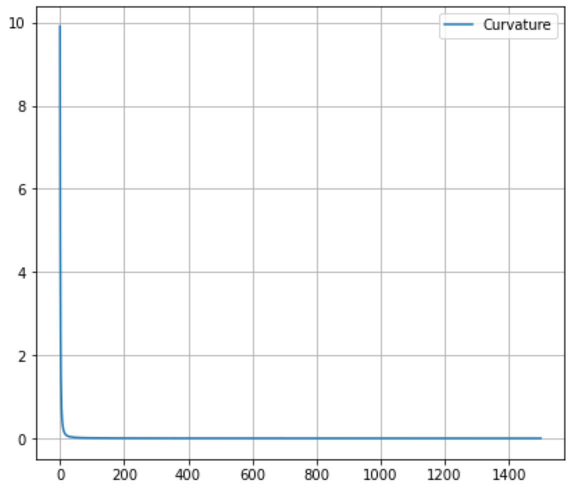
$$k = \frac{|x'y''(x) - y'(x)x''|}{\left(x'^2 + y'(x)^2\right)^{\frac{3}{2}}}$$

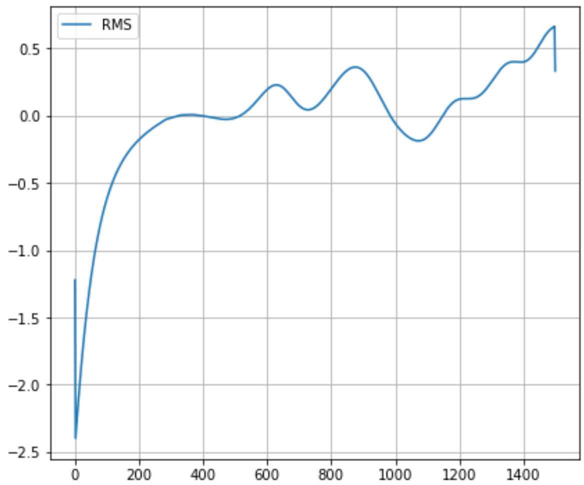
Avec y = log10(RMS), $RMS(k) = \sqrt{\frac{1}{L}\sum_{i=1}^{L}\left(SH_{k+1,i} - SH_{k,i}\right)^2}$ k = itération

Qui se simplifie en

$$k = \frac{|y''(x)|}{(1 + y'(x)^2)^{\frac{3}{2}}}$$

x : itération

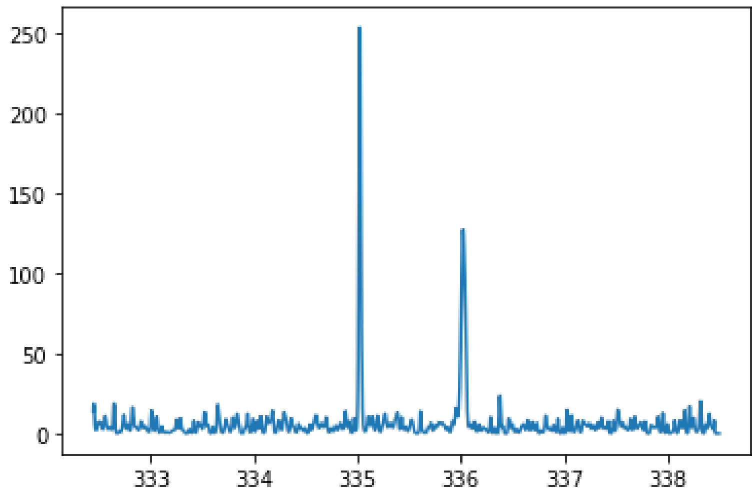




RMS vs iterations

courbure du RMS vs iterations

Tracé du signal estimé à l'itération optimal



pics irréalistes

```
Optimal number of iterations = 71

In [6]: error_signal_1
Out[6]: 684.3994306597986

In [7]: relative_error_signal_1
Out[7]: 235.64164329582886
```