OF_NxMx2:

$$\chi^2 = \left(V_i - A_\alpha \cdot T_{\alpha\,.i} \cdot e^{-i \cdot \omega \cdot t_\alpha}\right) \cdot \left(J_{i\,j}\right) \cdot \left(V_j - A_\beta \cdot T_{\beta\,.j} \cdot e^{i \cdot \omega \cdot t_\beta}\right)$$

Approximate signal in “ith” channel as a combination of “alpha” different templates all moving separately in time

Noise CSD

Conjugate of First term

i, j-> channnel_index-> 1, 2…N
alpha, beta -> Template index-> 1, 2,…M

The above equation is written in Fourier space, V’s, T’s are FFT of signals, templates respectively and J’s is current noise_CSD, thus I have ignored the summation over frequency notation.

Goal is to minimize “Chi2”: $\dfrac{\partial \chi^2}{\partial A_\gamma} = 0$

Upon doing this you end up with a matrix equation of

$[P]_{mxm}[A]_m=[Q]_m$ for each frequencies

Total dimension:

Dimension of P: [frequency_bin][m_template][m_template]
Dimension of A: [frequency_bin][m_template]
Dimension of Q: [frequency_bin][m_template]

$$P_{\alpha\,\beta} \times A_\beta = Q_\alpha$$

<u>**OF_NxMx2:**</u>

What are P, A, Q and chi2?

$$P_{\alpha\beta} \times A_\beta = Q_\alpha$$

$$P_{\alpha\beta} = \Sigma_{ij} \frac{T_{\alpha i} \cdot T_{\beta j} \cdot e^{-i\omega\cdot(\Delta_{\alpha\beta})}}{J_{ij}}$$

1/J_{ij} is actually inverse of J_{ij}(current noise CSD).

$$Q_\alpha = \Sigma_{ij} \frac{Vj \cdot T_{\alpha i} \cdot e^{i\omega\cdot(t_\alpha)}}{J_{ij}}$$

$$\chi^2 = \Sigma_{ij} \frac{V_i \cdot V_j}{J_{ij}} + \Sigma_{\alpha\beta} \Sigma_{ij} \frac{A_\alpha \cdot A_\beta \cdot T_{\alpha i} \cdot T_{\beta j} \cdot e^{-i\omega\cdot(\Delta_{\alpha\beta})}}{J_{ij}} - 2.\ \Sigma_\alpha \Sigma_{ij} \frac{A_\alpha V_j T_{\alpha i}\ e^{-i\omega t_\alpha}}{J_{ij}}$$

But, the chi2 actually simplifies to a simpler value; if you put the constrain that you are interest in chi2 only at the point where one satisfies P.A=Q

$$\Sigma_{\alpha\beta} \Sigma_{ij} \frac{A_\alpha \cdot A_\beta \cdot T_{\alpha i} \cdot T_{\beta j} \cdot e^{-i\omega\cdot(\Delta_{\alpha\beta})}}{J_{ij}} = \Sigma_\alpha \Sigma_{ij} \frac{A_\alpha V_j T_{\alpha i}\ e^{-i\omega t_\alpha}}{J_{ij}}$$

Than Chi2 simplifies to, remember you have to use the original chi2 for polarity constrain.

$$\chi^2 = \Sigma_{ij} \frac{V_i \cdot V_j}{J_{ij}} - \Sigma_\alpha \Sigma_{ij} \frac{A_\alpha V_j T_{\alpha i}\ e^{-i\omega t_\alpha}}{J_{ij}}$$

OF_NxMx2:

Approximation: signal in “ith” channel as a combination of “alpha” different templates all moving separately in time.

$$P_{\alpha\beta} \times A_{\beta} = Q_{\alpha}$$

$$P_{\alpha\beta} = \Sigma_{ij} \frac{T_{\alpha i} \cdot T_{\beta j} \cdot e^{-i\omega \cdot (\Delta_{\alpha\beta})}}{J_{ij}}$$

$$Q_{\alpha} = \Sigma_{ij} \frac{Vj \cdot T_{\alpha i} \cdot e^{i\omega \cdot (t_{\alpha})}}{J_{ij}}$$

$$\chi^2 = \Sigma_{ij} \frac{V_i \cdot V_j}{J_{ij}} - \Sigma_{\alpha} \Sigma_{ij} \frac{A_{\alpha} V_j T_{\alpha i} e^{-i\omega t_{\alpha}}}{J_{ij}}$$

This cannot be solved with limited computing resources, thus we decide to only move each templates with either t1 or t2

New assumption: t_alpha and Δ’s are decided by user;

Each template can either move with

a. phase $e^{-i w t_1}$:
b. phase $e^{-i w t_2}$:

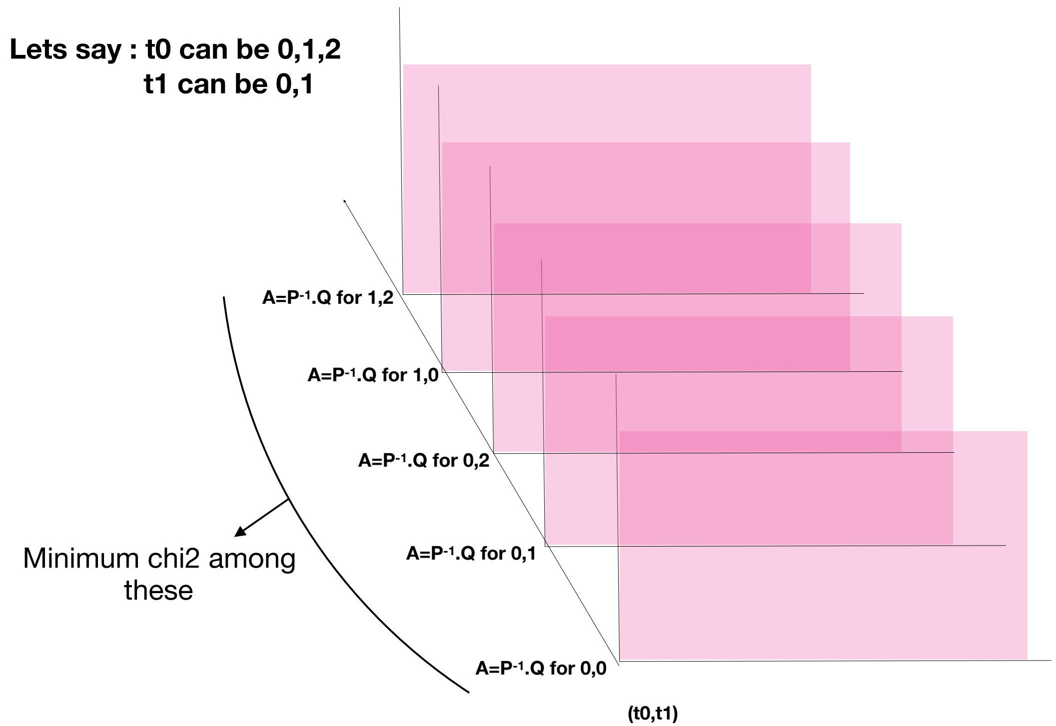
New Goal: Minimise chi2 all possible combination of order-pair (t_1, t_2)

OF_NxMx2:

New Goal: Minimise chi2 all possible combination of order-pair (t_1, t_2)

These are the steps to find chi2 for all possible order-pair(t1,t2)

1. Find Pmatrix for each order pair of (t1,t2)
2. Find Pmatrix_inverse for each order pair of (t1,t2)
3. Find Q matrix for each order pair of (t1,t2)
4. Multiply Pmatrix_inverse for each order pair of (t1,t2) with Q matrix for each order pair of (t1,t2) to get A(amplitude) for each order pair (t1,t2)
5. Get the [A,chi2] values for each order pair (t1,t2)
6. Minimise chi2, min [chi2 for each order pair (t1,t2)]



OF_NxMx2:

Lets say we have 4 templates and 2 channels, and the user hasn't decided which of the templates are gonna move together, thus assigning each template a different time..

case for 2x2xO [O time degrees of freedom]

$$\chi^2 =$$

$$\left(-A_1 . T_{11} . e^{-i.\omega.t_1} - A_2 . T_{21\omega} . e^{-i.\omega.t_2} - A_3 . T_{31} . e^{-i.\omega.t_3} - A_4 . T_{41} . e^{-i.\omega.t_4} + V_1 \quad -A_1 . T_{12} . e^{-i.\omega.t_1} - A_2 . T_{22\omega} . e^{-i.\omega.t_2} - A_3 . T_{32} . e^{-i.\omega.t_3} - A_4 . T_{42} . e^{-i.\omega.t_4} + V_2 \right)$$

$$\begin{pmatrix} \frac{1}{J_{11}} & \frac{1}{J_{12}} \\ \frac{1}{J_{21}} & \frac{1}{J_{22}} \end{pmatrix} \begin{pmatrix} -A_1 . T_{11} . e^{i.\omega.t_1} - A_2 . T_{21} . e^{i.\omega.t_2} - A_3 . T_{31} . e^{i.\omega.t_3} - A_4 . T_{41} . e^{i.\omega.t_4} + V_1 \\ -A_1 . T_{12} . e^{i.\omega.t_1} - A_2 . T_{22} . e^{i.\omega.t_2} - A_3 . T_{32} . e^{i.\omega.t_3} - A_4 . T_{42} . e^{i.\omega.t_4} + V_2 \end{pmatrix}$$

$$\frac{\partial \chi^2}{\partial A_\gamma} = 0$$ **-> P_matrix . A_vector = Q_vector**

P_matrix=

$$\begin{pmatrix}
\Sigma \frac{T_{1i} . T_{1j}}{J_{ij}} & \Sigma \frac{T_{1i} . T_{2j} . e^{-i\omega.(\Delta_{12})}}{J_{ij}} & \Sigma \frac{T_{1i} . T_{3j} . e^{-i\omega.(\Delta_{13})}}{J_{ij}} & \Sigma \frac{T_{1i} . T_{4j} . e^{-i\omega.(\Delta_{14})}}{J_{ij}} \\
\Sigma \frac{T_{2i} . T_{1j} . e^{-i\omega.(\Delta_{12})}}{J_{ij}} & \Sigma \frac{T_{2i} . T_{2j}}{J_{ij}} & \Sigma \frac{T_{2i}\ T_{3j} . e^{-i\omega.(\Delta_{23})}}{J_{ij}} & \Sigma \frac{T_{2i} . T_{4j} . e^{-i\omega.(\Delta_{24})}}{J_{ij}} \\
\Sigma \frac{T_{3i} . T_{1j} . e^{-i\omega.(\Delta_{13})}}{J_{ij}} & \Sigma \frac{T_{3i} . T_{2j} . e^{-i\omega.(\Delta_{23})}}{J_{ij}} & \Sigma \frac{T_{3i} . T_{3j}}{J_{ij}} & \Sigma \frac{T_{3i} . T_{4j} . e^{-i\omega.(\Delta_{34})}}{J_{ij}} \\
\Sigma \frac{T_{4i} . T_{1j} . e^{-i\omega.(\Delta_{14})}}{J_{ij}} & \Sigma \frac{T_{4i} . T_{2j} . e^{-i\omega.(\Delta_{24})}}{J_{ij}} & \Sigma \frac{T_{4i} . T_{3j} . e^{-i\omega.(\Delta_{34})}}{J_{ij}} & \Sigma \frac{T_{4i} . T_{4j}}{J_{ij}}
\end{pmatrix}$$

Q_matrix=

$$\begin{pmatrix}
\Sigma \frac{Vj . T_{1i} . e^{i\omega.(t_p)}}{J_{ij}} \\
\Sigma \frac{Vj . T_{2i} . e^{i\omega.(t_p)}}{J_{ij}} \\
\Sigma \frac{Vj . T_{3i} . e^{i\omega.(t_p)}}{J_{ij}} \\
\Sigma \frac{Vj . T_{4i} . e^{i\omega.(t_p)}}{J_{ij}}
\end{pmatrix}$$

OF_NxMx2:

The system of equations we need to solve: to exactly solve this we need to have inputs from the users; regarding which template they want to move together in time.

$$\begin{pmatrix} \Sigma \frac{T_{1i} . T_{1j}}{J_{ij}} & \Sigma \frac{T_{1i} \cdot T_{2j} \cdot e^{-i\omega \cdot (\Delta_{12})}}{J_{ij}} & \Sigma \frac{T_{1i} \cdot T_{3j} \cdot e^{-i\omega \cdot (\Delta_{13})}}{J_{ij}} & \Sigma \frac{T_{1i} \cdot T_{4j} \cdot e^{-i\omega \cdot (\Delta_{14})}}{J_{ij}} \\ \Sigma \frac{T_{2i} \cdot T_{1j} \cdot e^{-i\omega \cdot (\Delta_{12})}}{J_{ij}} & \Sigma \frac{T_{2i} \cdot T_{2j}}{J_{ij}} & \Sigma \frac{T_{2i}\, T_{3j} \cdot e^{-i\omega \cdot (\Delta_{23})}}{J_{ij}} & \Sigma \frac{T_{2i} \cdot T_{4j} \cdot e^{-i\omega \cdot (\Delta_{24})}}{J_{ij}} \\ \Sigma \frac{T_{3i} . T_{1j} \cdot e^{-i\omega \cdot (\Delta_{13})}}{J_{ij}} & \Sigma \frac{T_{3i} \cdot T_{2j} \cdot e^{-i\omega \cdot (\Delta_{23})}}{J_{ij}} & \Sigma \frac{T_{3i} \cdot T_{3j}}{J_{ij}} & \Sigma \frac{T_{3i} \cdot T_{4j} \cdot e^{-i\omega \cdot (\Delta_{34})}}{J_{ij}} \\ \Sigma \frac{T_{4i} \cdot T_{1j} \cdot e^{-i\omega \cdot (\Delta_{14})}}{J_{ij}} & \Sigma \frac{T_{4i} \cdot T_{2j} \cdot e^{-i\omega \cdot (\Delta_{24})}}{J_{ij}} & \Sigma \frac{T_{4i} \cdot T_{3j} \cdot e^{-i\omega \cdot (\Delta_{34})}}{J_{ij}} & \Sigma \frac{T_{4i} \cdot T_{4j}}{J_{ij}} \end{pmatrix} \begin{pmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{pmatrix}$$

$$= \begin{pmatrix} \Sigma \frac{Vj . T_{1i} \cdot e^{i\omega \cdot (t\_p)}}{J_{ij}} \\ \Sigma \frac{Vj . T_{2i} \cdot e^{i\omega \cdot (t\_p)}}{J_{ij}} \\ \Sigma \frac{Vj . T_{3i} \cdot e^{i\omega \cdot (t\_p)}}{J_{ij}} \\ \Sigma \frac{Vj . T_{4i} \cdot e^{i\omega \cdot (t\_p)}}{J_{ij}} \end{pmatrix}$$

the user inputs, a time tag vector; specifying which template move together:

[0, 1, 0, 1] -> saying the first and the third template move together, and the second and last one move separately.

From the “time tag” [0, 1, 0, 1], we form a matrix

$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

If 1, I have to do ifft for calculating the P_matrix; if not, just normal sum.

OF_NxMx2:

$$
\begin{pmatrix}
\Sigma \frac{T_{1i}.T_{1j}}{J_{ij}} & \Sigma \frac{T_{1i}.T_{2j}.e^{-i\omega.(\Delta_{12})}}{J_{ij}} & \Sigma \frac{T_{1i}.T_{3j}.e^{-i\omega.(\Delta_{13})}}{J_{ij}} & \Sigma \frac{T_{1i}.T_{4j}.e^{-i\omega.(\Delta_{14})}}{J_{ij}} \\
\Sigma \frac{T_{2i}.T_{1j}.e^{-i\omega.(\Delta_{12})}}{J_{ij}} & \Sigma \frac{T_{2i}.T_{2j}}{J_{ij}} & \Sigma \frac{T_{2i}\,T_{3j}.e^{-i\omega.(\Delta_{23})}}{J_{ij}} & \Sigma \frac{T_{2i}.T_{4j}.e^{-i\omega.(\Delta_{24})}}{J_{ij}} \\
\Sigma \frac{T_{3i}.T_{1j}.e^{-i\omega.(\Delta_{13})}}{J_{ij}} & \Sigma \frac{T_{3i}.T_{2j}.e^{-i\omega.(\Delta_{23})}}{J_{ij}} & \Sigma \frac{T_{3i}.T_{3j}}{J_{ij}} & \Sigma \frac{T_{3i}.T_{4j}.e^{-i\omega.(\Delta_{34})}}{J_{ij}} \\
\Sigma \frac{T_{4i}.T_{1j}.e^{-i\omega.(\Delta_{14})}}{J_{ij}} & \Sigma \frac{T_{4i}.T_{2j}.e^{-i\omega.(\Delta_{24})}}{J_{ij}} & \Sigma \frac{T_{4i}.T_{3j}.e^{-i\omega.(\Delta_{34})}}{J_{ij}} & \Sigma \frac{T_{4i}.T_{4j}}{J_{ij}}
\end{pmatrix}
\begin{pmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{pmatrix}
=
\begin{pmatrix}
\Sigma \frac{V_j.T_{1i}.e^{i\omega.(t\_p)}}{J_{ij}} \\
\Sigma \frac{V_j.T_{2i}.e^{i\omega.(t\_p)}}{J_{ij}} \\
\Sigma \frac{V_j.T_{3i}.e^{i\omega.(t\_p)}}{J_{ij}} \\
\Sigma \frac{V_j.T_{4i}.e^{i\omega.(t\_p)}}{J_{ij}}
\end{pmatrix}
$$

These terms in the P matrix, we don't need to perform any ifft, just the normal sum over all frequencies

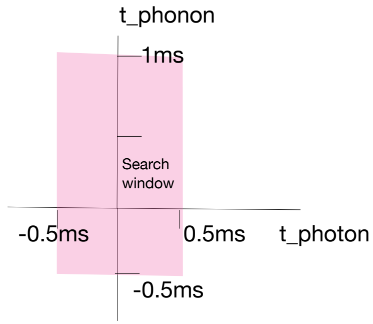
Once P matrix is computed, I compute P_inv Matrix and than form P_inv Matrix for the order pair (to-t1), next question is how do I find the time order pair?

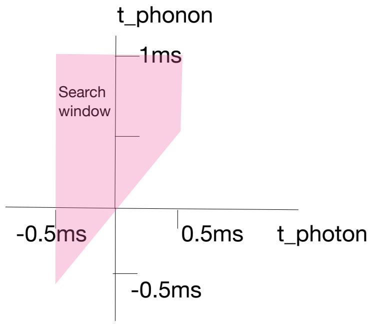
For the 2 time degree of freedom; photon and phonon

Previously we used to create all possible order pair: (t_photon, t_phonon)

now I have restricted to only the following order pair: (t_photon, t_phonon)

t_phonon>t_photon





OF_NxMx2:

Lets try to solve it for an actual case: OF_2x4x2

1. 1 independent scintillation and 1 independent Evaporation in 1 channel
2 .1 independent scintillation and 1 independent Evaporation in 2 channel

3. Independent scintillations in each channel moving together, Independent evaporation moving together

Template matrix:

$$\begin{pmatrix} \text{scintillation}1 & \text{evaporation}1 & 0 & 0 \\ 0 & 0 & \text{scintillation}2 & \text{evaporation}2 \end{pmatrix}$$

$$= \begin{pmatrix} S_{11} & S_{12} & 0 & 0 \\ 0 & 0 & S_{21} & S_{22} \end{pmatrix}$$

Fit window: [[start_window_Scint, end_window_Scint],
[start_window_Evaporation, end_window_Evaporation]]

$$\begin{pmatrix} \frac{s_{11}.s_{11}}{J_{11}} & \text{Real}\left[\frac{e^{i.\omega.(t_1-t_2)}.s_{11}.s_{12}}{J_{11}}\right] & \text{Real}\left[\frac{s_{11}.s_{21}}{J_{12}}\right] & \text{Real}\left[\frac{e^{i.\omega.(t_1-t_2)}.s_{11}.s_{22}}{J_{12}}\right] \\ \text{Real}\left[\frac{e^{i.\omega.(t_1-t_2)}.s_{11}.s_{12}}{J_{11}}\right] & \frac{s_{12}.s_{12}}{J_{11}} & \text{Real}\left[\frac{e^{i.\omega.(t_1-t_2)}.s_{12}.s_{21}}{J_{12}}\right] & \text{Real}\left[\frac{s_{12}.s_{22}}{J_{12}}\right] \\ \text{Real}\left[\frac{s_{11}.s_{21}}{J_{12}}\right] & \text{Real}\left[\frac{e^{i.\omega.(t_1-t_2)}.s_{12}.s_{21}}{J_{12}}\right] & \frac{s_{21}.s_{21}}{J_{22}} & \text{Real}\left[\frac{e^{i.\omega.(t_1-t_2)}.s_{21}.s_{22}}{J_{22}}\right] \\ \text{Real}\left[\frac{e^{i.\omega.(t_1-t_2)}.s_{11}.s_{22}}{J_{12}}\right] & \text{Real}\left[\frac{s_{12}.s_{22}}{J_{12}}\right] & \text{Real}\left[\frac{e^{i.\omega.(t_1-t_2)}.s_{21}.s_{22}}{J_{22}}\right] & \frac{s_{22}.s_{22}}{J_{22}} \end{pmatrix} \begin{pmatrix} A_{11} \\ A_{12} \\ A_{21} \\ A_{22} \end{pmatrix}$$

$$= \begin{pmatrix} \text{Real}\left[\frac{e^{i.\omega.t_1}.s_{11}\, v_1}{J_{11}}\right] + \text{Real}\left[\frac{e^{i.\omega.t_1}.s_{11}\, v_2}{J_{12}}\right] \\ \text{Real}\left[\frac{e^{i.\omega.t_2}.s_{12}\, v_1}{J_{11}}\right] + \text{Real}\left[\frac{e^{i.\omega.t_2}.s_{12}\, v_2}{J_{12}}\right] \\ \text{Real}\left[\frac{e^{i.\omega.t_1}.s_{21}\, v_1}{J_{21}}\right] + \text{Real}\left[\frac{e^{i.\omega.t_1}.s_{21}\, v_2}{J_{22}}\right] \\ \text{Real}\left[\frac{e^{i.\omega.t_2}.s_{22}\, v_1}{J_{21}}\right] + \text{Real}\left[\frac{e^{i.\omega.t_2}.s_{22}\, v_2}{J_{22}}\right] \end{pmatrix}$$

OF_NxMx2:

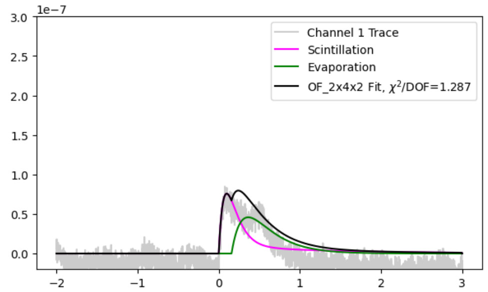
```
#We will use this function to instantiate the OF_NxMx2
def Call_NxMx2_optimal_Filter_object(template_array, template_tags,
templates_time_tag, csd,  channel_name_list, fs,
pretrigger_samples,fit_window):
    # instantiate OFnxm
    return qp.OFnxmx2(of_base=None, templates =template_array,
template_tags=template_tags ,templates_time_tag =templates_time_tag ,
channels= channel_name_list, csd=csd,sample_rate=fs,
pretrigger_samples=pretrigger_samples,fit_window = fit_window)
```

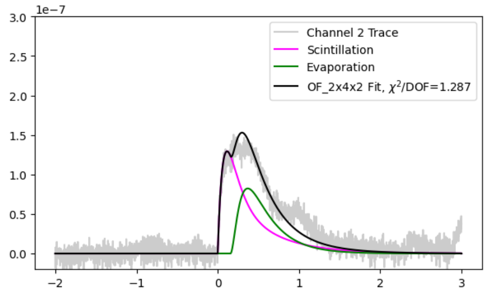
```
# to start calling OF_nxmx2, you should have all these things already setup

#First what is the channel list, you have to define it this way : (
channel_name_list="CPD1|CPD2"

#First lets define what are the different types of template you want to have: m
different types, they can different among the channels
template_tags_2x4 =
['scintillation_CPD1','evaporation_CPD1','scintillation_CPD2','evaporation_CPD2']


# m different types of templates for each channel: a total of 8 different templates
template_array_2x4 = np.asarray((CPD1_scintillation, CPD1_evaporation,
0*CPD2_scintillation, 0*CPD2_evaporation,\
                                 0*CPD1_scintillation, 0*CPD1_evaporation,
CPD2_scintillation, CPD2_evaporation))


# out of these m templates which of them you want to move together
tags_which_template_move_together = np.array([0,1,0,1])

#what is the fit window for nxmx2
fit_window = [[-625,625],[-625,1250]]

#csd cross spectral density, needs to evaluate early on
csd = np.copy(noise_csd)

#a noise trace is needed for the nxm function
noise_traces= traces[0]

#smapling frequency
fs = traces_metadata.item()['sample_rate']
```

```
OF_2x4x2 =
Call_NxMx2_optimal_Filter_object(template_array_2x4,templ
ate_tags_2x4,
tags_which_template_move_together,csd,channel_name_list,
fs, 2500,fit_window)
```

```
i=36
# now we will clcualte q_vector, signal_filt_mat_td etc ..all
matrices related to signal
OF_2x4x2.calc(channels= channel_name_list,
signal=signal_traces[i])

#here we will get the fit quantities
OF_2x4x2.get_fit(channels= channel_name_list,
signal=signal_traces[i]

amps_min= OF_2x4x2._of_amp

time_first_pulse= OF_2x4x2._index_first_pulse
time_second_pulse= OF_2x4x2._index_second_pulse
chi2_min_per_dOF= OF_2x4x2._of_chi2_per_DOF
```

After that you can plot the result:





https://github.com/spice-herald/QETpy/tree/working_nxmx2