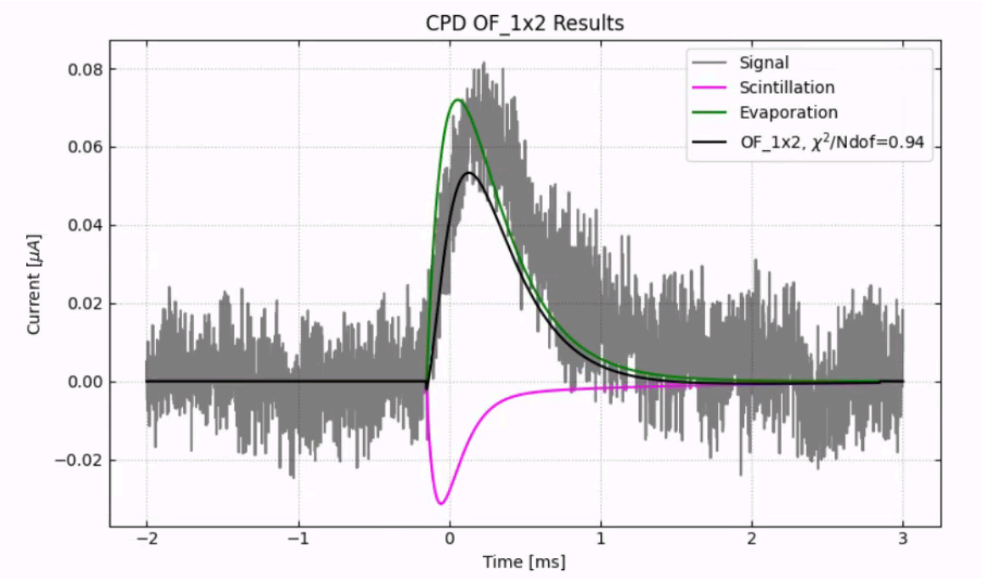
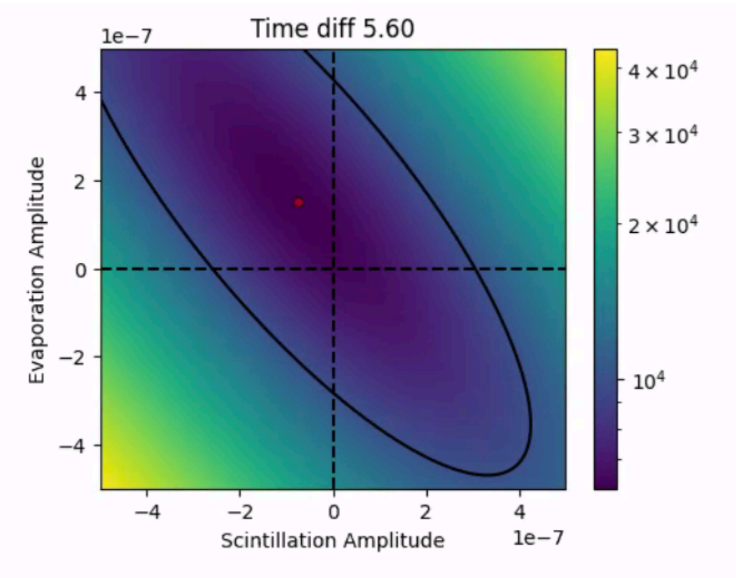
Problem that we are trying to solve: Sometime the fit is good; but is unphysical…

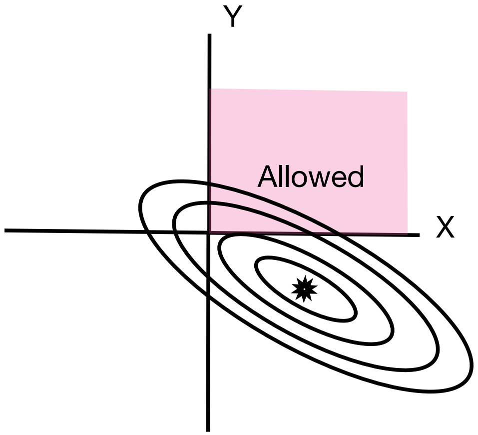
Means chi2/DOF ~1 but the Ampiltude of the fit A1(scintillation) and A2(evaporation) are negative



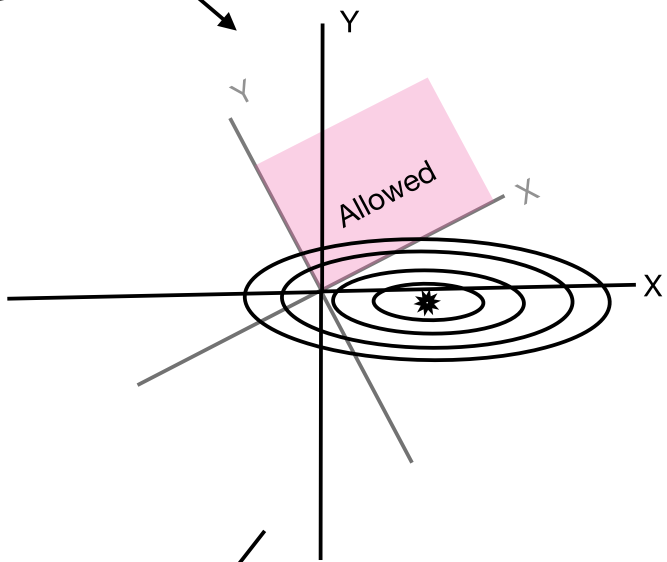


Red dot indicates the optimised point and one can clearly see that evaporation amplitude is not positive.

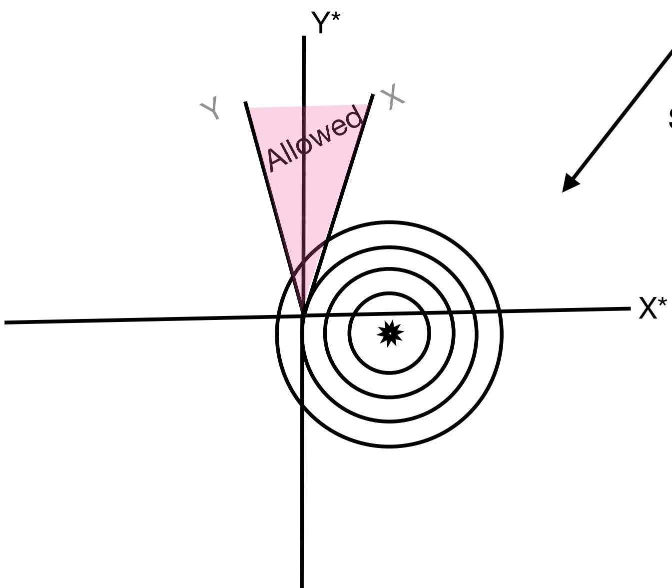
Polarity Constraints for OF_1x2x2:



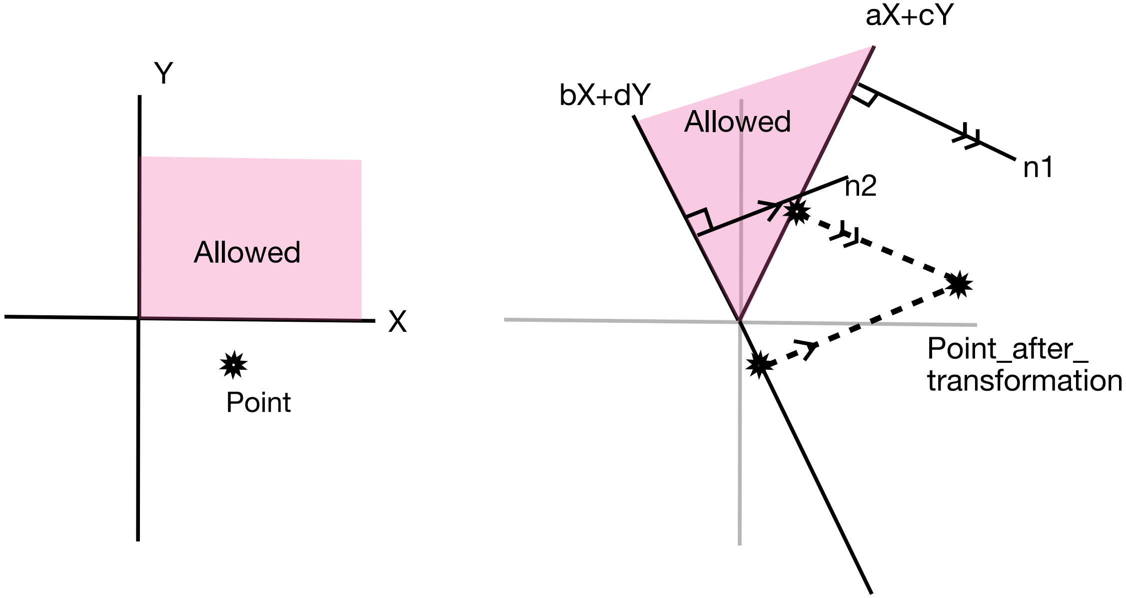
Rotate to get rid of cross
Terms



Scale to make spheriods



Polarity Constraints for OF_1x2x2:



$$\text{transformation} = [\text{Rotation}] \times [\text{Scaling}] = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

even though n1 and n2 are not orthogonal to each other, the shortest distance to the new boundaries is along these vector.

n1 and n2 is termed as "wedge vector"

$$n1 = \text{Normalised}\begin{bmatrix} a \\ c \end{bmatrix}, \quad n2 = \text{Normalised}\begin{bmatrix} b \\ d \end{bmatrix}$$

Polarity Constraints for OF_1x2x2:

Get rid of them by rotating to the right basis

Distance metric:

$$ds^2 = dx_1^2 + dx_2^2 + dx_3^2 + \dots + dx_n^2 + \text{cross } \cancel{\text{terms}}$$

Chi2 space is n-eplisoid :

$$\frac{x_1^2}{a_1^2} + \frac{x_2^2}{a_2^2} + \frac{x_3^2}{a_3^2} + \dots + \frac{x_4^2}{a_4^2} = 1$$

Redefine our space so that it turns into a space where an n-ellipsoid looks like n-sphere

$$ds^2 = (a_1 . dx_1)^2 + (a_2 . dx_2)^2 + (a_3\, dx_3)^2 + \dots + (a_n\, dx_n)^2$$

$$ds^2 = (dx'_1)^2 + (dx'_2)^2 + (dx'_3)^2 + \dots + (dx'_n)^2$$

Chi2 space is n-sphere in the new metric :

$$x'^2_1 + x'^2_2 + x'^2_3 + \dots + x'^2_n = 1$$

Now move to the nearest point on each boundaries, along the “wedge vector”

Now retransform back to the original co-ordinate and

if still the point falls in the unallowed region then what? Than move this point to (0,0)

Polarity Constraints for OF_1x2x2:

Pre-calculation happening in OF_base

A denotes: unconstrained optimised point

Covariance Matrix: $\mathbf{P^{-1}_{mxm}}$

Scaling_matrix = (EigenValue matrix of covariance matrix)$^{-1}_{mxm}$

Rotation_matrix = (EigenVectors of Covariance Matrix)$_{mxm}$

Transformation_Matrix = (Scaling_matrix . Rotation_matrix)$_{mxm}$

Inverse_Transformation_Matrix= (Transformation_Matrix)$^{-1}_{mxm}$

Allowed_boundary_condition = diagonal(1)$_{mxm}$

Wedge_matrix= (Transformation_Matrix . Allowed_boundary_condition)^T

Wedge_vector $_i$ =
(Transformation_Matrix . Allowed_boundary_condition)^T $_i$ /(norm)

If [product(unconstrained optimised point)<0]:
Transformed_point = Transformation_Matrix . **A**
# select along which wedge vectors one needs to move "ith"
min(Transformed_point . Wedge_vector $_i$)

Constraint_transformed_point = Transformed_point -
(Transformed_point . Wedge_vector $_i$) Wedge_vector $_i$

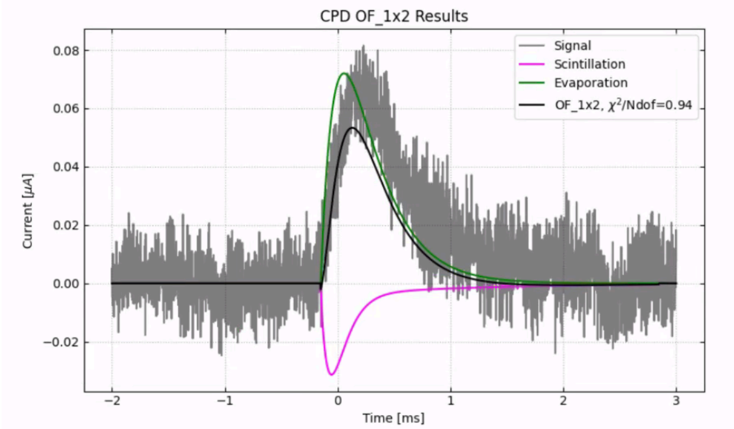
Constrained_Point =
Inverse_Transformation_Matrix . Constraint_transformed_point

If [product(Constrained_Point)<0] -> Constrained_Point -> [0,0]

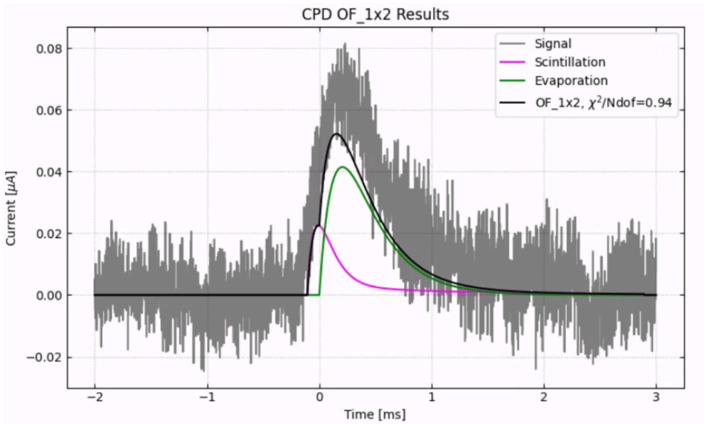
https://github.com/spice-herald/QETpy/tree/feature/OF_1x3/qetpy/core

Polarity Constraints for OF_1x2x2:

No Polarity Constrain



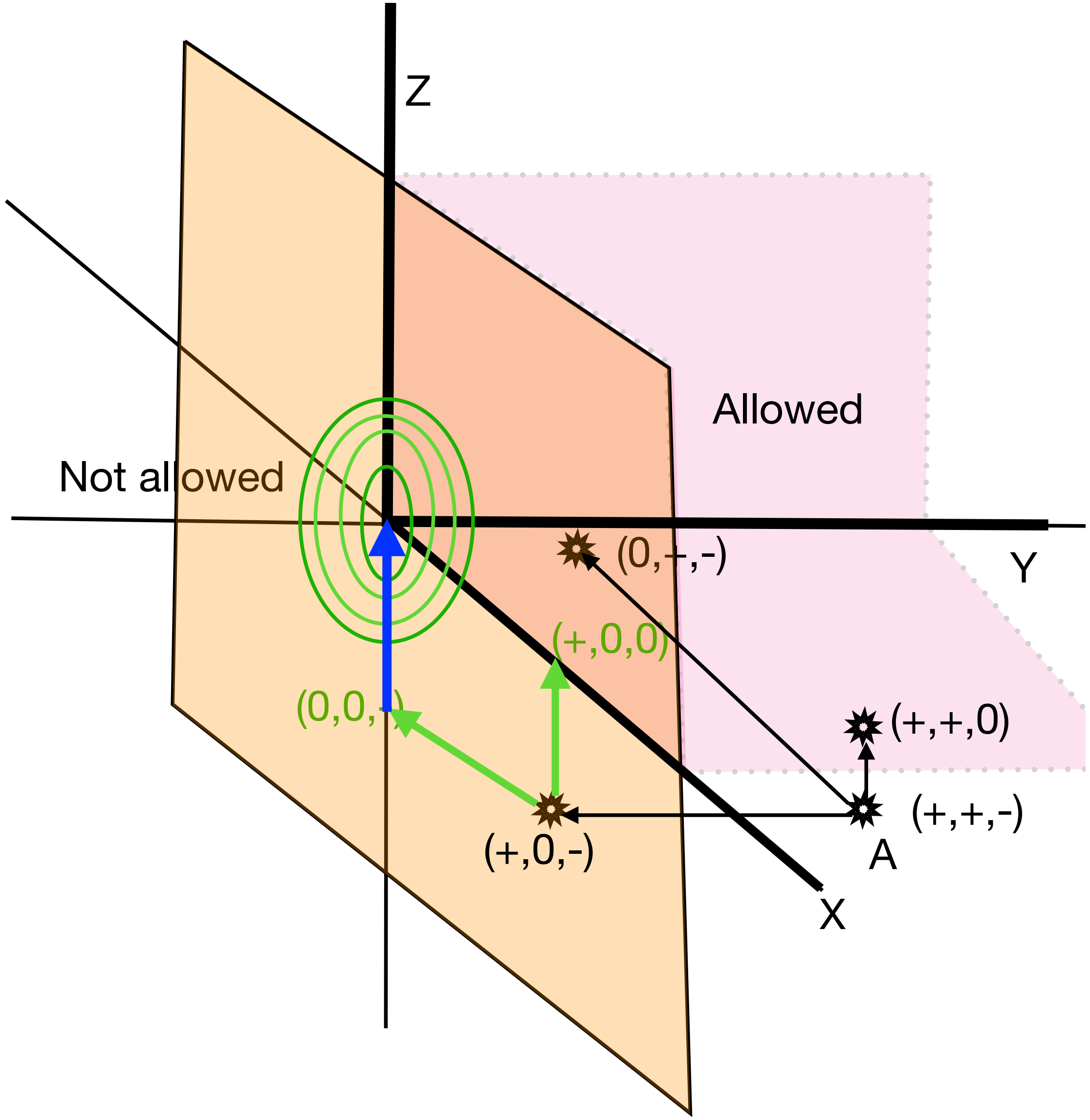
Polarity Constrain



We do get reasonable answer,
when we use polarity constrain for OF_1x2x2

This algorithm has not been implemented for
a generalised NxMx2 OF Filter, a visual representation of what needs to be done is shown in the next page.

The trickier part is to calculate the many wedge vectors that are possible as the dimension increases.



Point A, can move to three diff planes, along the three wedge vector, lets say we move to (+,0,-) point. But this is also disallowed. Now what do we do?

We constraint in this 2D plane? Find new wedge vector for this 2-D plane.

If for eg, the minium happens to (0,0,-), which is still disallowed than the only solution is (0,0,0)