General Rules when using time fields

$$\mathcal{T}(\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_n) \stackrel{def}{=} \max\left(\mathcal{T}_1(\mathcal{O}_1), \mathcal{T}_2(\mathcal{O}_2), \ldots, \mathcal{T}_n(\mathcal{O}_n)\right)$$

$$\hookrightarrow \max(a_1, a_2, \ldots a_n)$$

$$= \begin{cases} a_1 & if\ (a_1 > a_2) \wedge (a_1 > a_3) \ldots \wedge (a_1 > a_n) \\ a_2 & if\ (a_2 > a_1) \wedge (a_2 > a_3) \ldots \wedge (a_2 > a_n) \\ \ldots\ldots & \\ a_n & otherwise \end{cases}$$

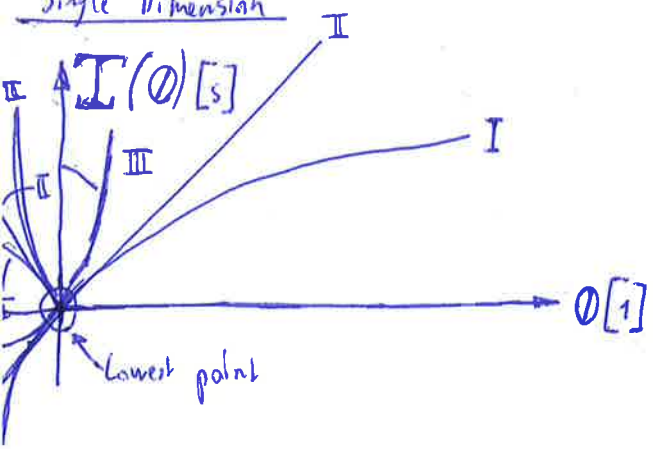
Overlaying the time fields

When overlaying time fields, every single field is pluged into the max() function

Problem is that max() ~~fucks~~ returns a unsteady function in most cases

Optimisation

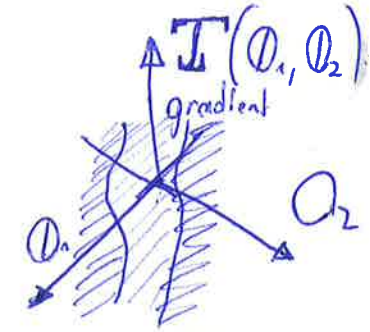
Single Dimension



Optimization in a single measurement does not make sense, as there are no parameters to play around

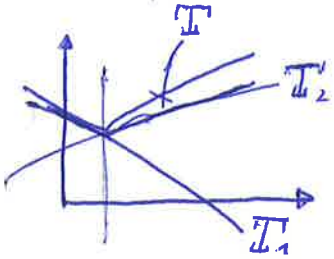
$$\mathcal{T}(\mathcal{O}) = \mathcal{O}$$

Second Dimension



$\mathcal{O}_1 \Longleftrightarrow \mathcal{O}_2$ dependent

$$\mathcal{T}_1(\mathcal{O}_1) = \mathcal{T}_2(\mathcal{O}_2)$$



$$\Rightarrow \mathcal{O}_1 = \mathcal{T}_1^{-1}\left(\mathcal{T}_2(\mathcal{O}_2)\right)$$

$$\Rightarrow \mathcal{O}_1(\mathcal{O}_2)$$