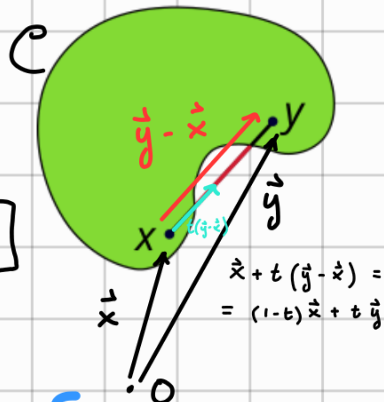
DIJKSTRA'S ALGORITHM

- A HILBERT SPACE H IS A VECTOR SPACE EQUIPPED WITH AN INNER PRODUCT THAT INDUCES A DISTANCE FUNCTION FOR WHICH THE SPACE IS A COMPLETE METRIC SPACE

- A CONVEX SET C IS A SET THAT SATISFIES

$$\vec{x}\lambda + (1-\lambda)\vec{y} \in C$$

$$\forall\ \vec{x}, \vec{y} \in C \ \wedge\ \lambda \in [0, 1]$$



PROJECTION ONTO CONVEX SETS

Let H BE A HILBERT SPACE, let $C_1, C_2 \subset H$ BE TWO CONVEX SETS.
THE PROJECTION $\vec{p}$ OF A POINT $\vec{x}_0$ ONTO THE INTERSECTION OF C_1 AND C_2 IS GIVEN BY:

$$\vec{p} \in C_1 \cap C_2 : \|\vec{p} - \vec{x}_0\| \leq \|\vec{r} - \vec{x}_0\| \quad \forall\ \vec{r} \in C_1 \cup C_2$$

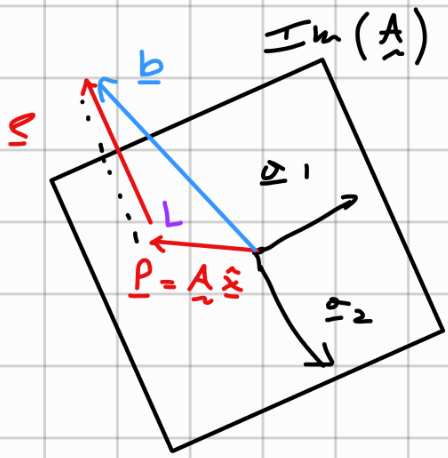
$$\vec{p} = P_{C_1 \cap C_2}(\vec{x}_0)$$ IS SOLVED RECURSIVELY BY:

$$\boxed{\vec{p}_{n+1} = P_{C_1}\left(P_{C_2}(\vec{p}_n)\right)}$$ (ALTERNATING PROJECTIONS) WHERE:

$$P_C(\vec{x}) = \left[\underset{\sim}{C}\left(\underset{\sim}{C}^T\underset{\sim}{C}\right)^{-1}\underset{\sim}{C}^T\right]\vec{x} \in Im(\underset{\sim}{C})$$

* DERIVATION BELOW FROM A1

PROJECTIONS DERIVATION ☆



- GIVEN : $\underline{p} \in \operatorname{Im}(\underset{\sim}{A})$

WE CAN RE-WRITE $\underline{p}$ AS:

$$\underline{p} = \underset{\sim}{A}\,\underline{\hat{x}} \quad - \quad (1)$$

WHERE $\underline{\hat{x}}$ IS THE VECTOR FROM THE ORIGIN TO THE POINT WHERE $\underline{b}$ IS PROJECTED ON $\operatorname{Im}(\underset{\sim}{A})$

- WE CAN DEFINE THE VECTOR $\underline{b}$ AS THE SUM OF VECTOR $\underline{p}$ PLUS AN ERROR VECTOR $\underline{e}$:

$$\underline{b} = \underline{p} + \underline{e} \quad - \quad (2)$$

SUCH THAT $\underline{e}$ IS $\perp$ TO THE IMAGE OF $\underset{\sim}{A}$. BUT, AS SEEN IN Q4 a), THIS IMPLIES THAT $\underline{e}$ LIES IN THE LEFT NULLSET OF $\underset{\sim}{A}$

$$\underline{e} \perp \operatorname{Im}(\underset{\sim}{A}) \Rightarrow \underline{e} \in N(\underset{\sim}{A}^T)$$

$$\therefore \underset{\sim}{A}^T \underline{e} = \underline{0} \quad - \quad (3)$$

- COMBINE EQUATIONS (1) AND (2):

$$\underline{e} = \underline{b} - \underline{p} = (\underline{b} - \underset{\sim}{A}\,\underline{\hat{x}})$$

- COMBINE THIS RESULT WITH (3):

$$\underset{\sim}{A}^T \underline{e} = \underset{\sim}{A}^T(\underline{b} - \underset{\sim}{A}\underline{\hat{x}}) = \underline{0}$$

- SOLVE FOR VECTOR $\underline{\hat{x}}$:

$$\underset{\sim}{A}^T \underline{b} - \underset{\sim}{A}^T \underset{\sim}{A}\underline{\hat{x}} = \underline{0}$$

$$\therefore \underline{\hat{x}} = (\underset{\sim}{A}^T \underset{\sim}{A})^{-1} \underset{\sim}{A}^T \underline{b} \quad //$$

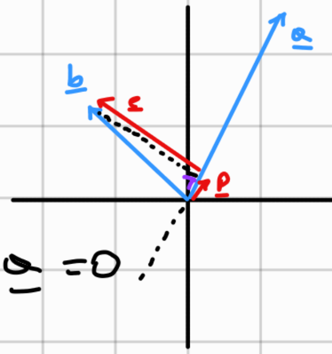
- AND NOW SUBSTITUTE INTO EQ. (1):

PROJECTION ONTO A SUBSPACE

$$\boxed{\underline{p} = \underset{\sim}{A}\underline{\hat{x}} = \underbrace{\left[\underset{\sim}{A}(\underset{\sim}{A}^T \underset{\sim}{A})^{-1} \underset{\sim}{A}^T\right]}_{\underset{\sim}{P}} \underline{b}}$$

THIS IS CALLED PROJECTION MATRIX

- SIMILARLY, BY A SIMILAR PROCEDURE, WE CAN OBTAIN THE 1D VERSION:



$$\underline{b} = \underline{p} + \underline{e} = \hat{x}\underline{a} + \underline{e} \Rightarrow \underline{e} = \underline{b} - \hat{x}\underline{a}$$

FOR MIN DISTANCE $\|\underline{e}\|$: $\underline{e}\cdot\underline{a} = 0 \Rightarrow (\underline{b} - \hat{x}\underline{a})\cdot\underline{a} = 0$

$$\therefore \hat{x}(\underline{a}^T\underline{a}) = \underline{a}^T\underline{b} \Rightarrow \hat{x} = \frac{\underline{a}^T\underline{b}}{\underline{a}^T\underline{a}} \quad \text{(SCALAR)}$$

PROJECTION ONTO LINE

- $\underline{p} = \hat{x}\underline{a} \Rightarrow$

$$\boxed{\underline{p} = \left[\frac{\underline{a}\,\underline{a}^T}{\underline{a}^T\underline{a}}\right]\underline{b}}$$

DO NOT FORM MATRIX:

1) OBTAIN SCALAR $\underline{a}^T\underline{b}$
2) DIVIDE BY SCALAR $\underline{a}^T\underline{a}$
3) MULTIPLY BY VECTOR $\underline{a}$

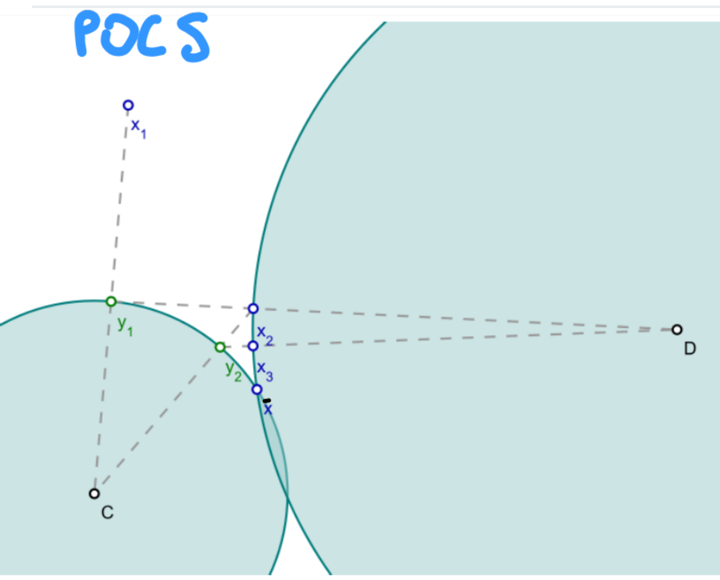
Dykstra's projection algorithm

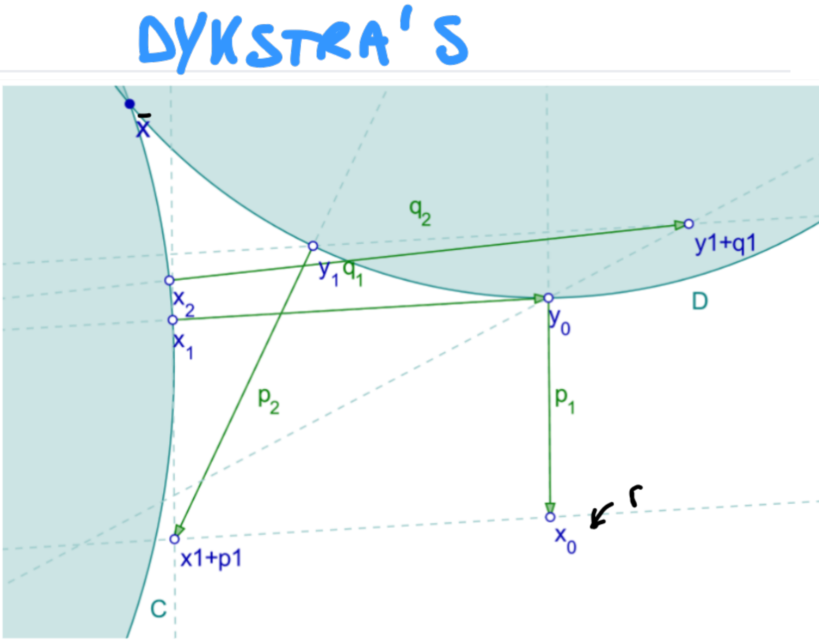
Dykstra's algorithm is a method that computes a point in the intersection of convex sets, and is a variant of the alternating projection method (also called the projections onto convex sets method). In its simplest form, the method finds a point in the intersection of two convex sets by iteratively projecting onto each of the convex set; it differs from the alternating projection method in that there are intermediate steps. A parallel version of the algorithm was developed by Gaffke and Mathar.

The method is named after Richard L. Dykstra who proposed it in the 1980s.

A key difference between Dykstra's algorithm and the standard alternating projection method occurs when there is more than one point in the intersection of the two sets. In this case, the alternating projection method gives some arbitrary point in this intersection, whereas Dykstra's algorithm gives a specific point: the projection of *r* onto the intersection, where *r* is the initial point used in the algorithm,

Algorithm





Dykstra's algorithm finds for each r the only $\bar{x} \in C \cap D$ such that:

$$\|\bar{x} - r\|^2 \le \|x - r\|^2, \text{for all } x \in C \cap D,$$

where C, D are convex sets. This problem is equivalent to finding the projection of r onto the set $C \cap D$, which we denote by $\mathcal{P}_{C \cap D}$.

SEE ABOVE

To use Dykstra's algorithm, one must know how to project onto the sets C and D separately. ☆

First, consider the basic alternating projection (aka POCS) method (first studied, in the case when the sets C, D were linear subspaces, by John von Neumann[1]), which initializes $x_0 = r$ and then generates the sequence

$$x_{k+1} = \mathcal{P}_C\left(\mathcal{P}_D(x_k)\right).$$

Dykstra's algorithm is of a similar form, but uses additional auxiliary variables. Start with $x_0 = r, p_0 = q_0 = 0$ and update by

$$y_k = \mathcal{P}_D(x_k + p_k)$$

DISTANCE BETWEEN $x_k + p_k$ AND NEWEST D PROJECTION (y_k)

$$p_{k+1} = x_k + p_k - y_k$$

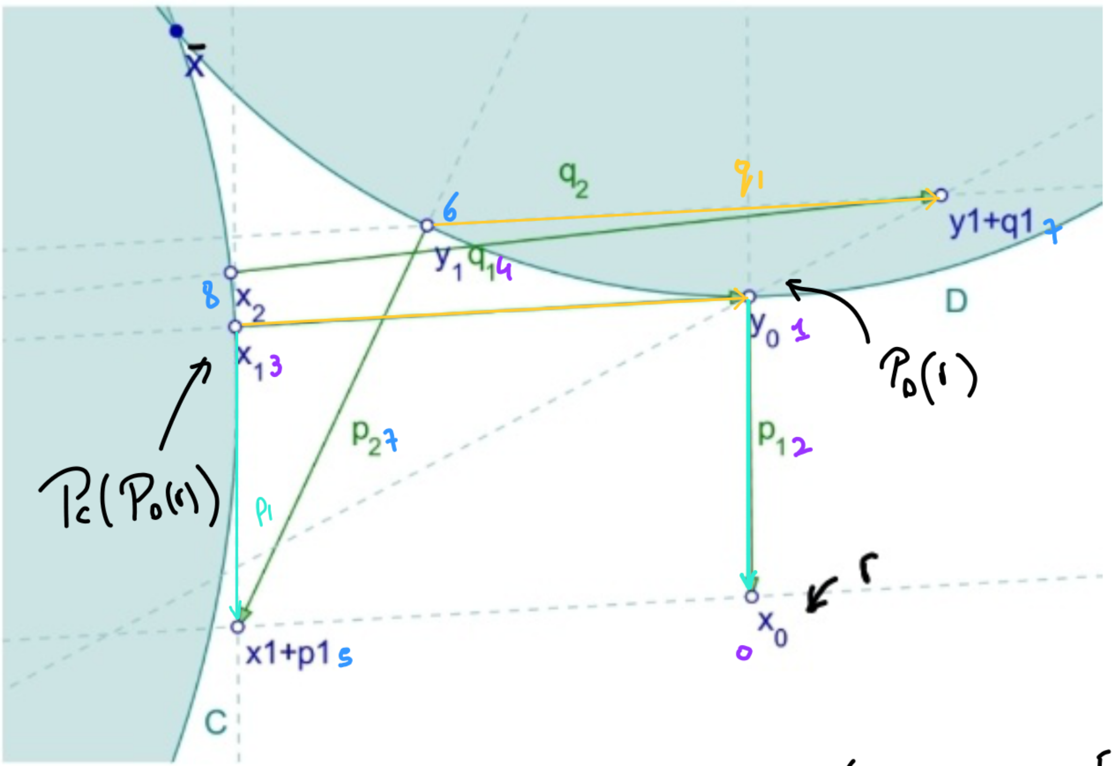
$$x_{k+1} = \mathcal{P}_C(y_k + q_k)$$

$$q_{k+1} = y_k + q_k - x_{k+1}.$$

DISTANCE BETWEEN $y_k + q_k$ AND NEWEST C PROJECTION

Then the sequence (x_k) converges to the solution of the original problem. For convergence results and a modern perspective on the literature, see [2]

for example, here:



$y_0 = \mathcal{P}_D(r)$

$p_1 = r - \mathcal{P}_D(r)$

$x_1 = \mathcal{P}_C(\mathcal{P}_D(r))$

$q_1 = \mathcal{P}_D(r) - \mathcal{P}_C(\mathcal{P}_D(r))$

$y_1 = \mathcal{P}_D\left(\mathcal{P}_C(\mathcal{P}_D(r)) + [r - \mathcal{P}_D(r)]\right)$

$p_2 = \mathcal{P}_C(\mathcal{P}_D(r)) + [r - \mathcal{P}_D(r)] - (\ldots)$

AND SO ON