■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... Considering this possibility, we must distinguish between these two cases:

(i) $L(X_1) = \underline{1 + c_1(X_1 - 1) = 0} \Rightarrow \underline{X_1 = 1 - 1/c_1}$, $c_1 \neq 0$

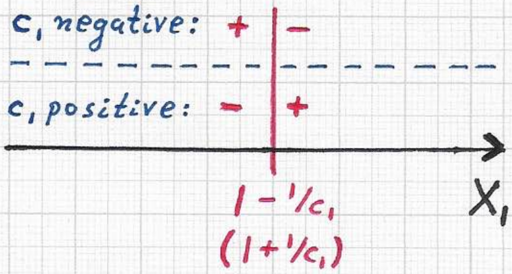
$$\Rightarrow L(X_1) \begin{cases} \leq 0, & -\infty < X_1 \leq 1 - 1/c_1 \\ > 0, & 1 - 1/c_1 < X_1 < \infty \end{cases} \quad \text{for } 0 < c_1 < \infty;$$

$$L(X_1) \begin{cases} > 0, & -\infty < X_1 < 1 - 1/c_1 \\ \leq 0, & 1 - 1/c_1 \leq X_1 < \infty \end{cases} \quad \text{for } -\infty < c_1 < 0.$$

(ii) $L(X_1) = \underline{-1 + c_1(X_1 - 1) = 0} \Rightarrow \underline{X_1 = 1 + 1/c_1}$, $c_1 \neq 0$

$$\Rightarrow L(X_1) \begin{cases} \leq 0, & -\infty < X \leq 1 + 1/c_1 \\ > 0, & 1 + 1/c_1 < X_1 < \infty \end{cases} \quad \text{for } 0 < c_1 < \infty;$$

$$L(X_1) \begin{cases} > 0, & -\infty < X_1 < 1 + 1/c_1 \\ \leq 0, & 1 + 1/c_1 \leq X_1 < \infty \end{cases} \quad \text{for } -\infty < c_1 < 0.$$



The X_1-line is split into half-spaces for $c_0 + c_1 = +/-1$.

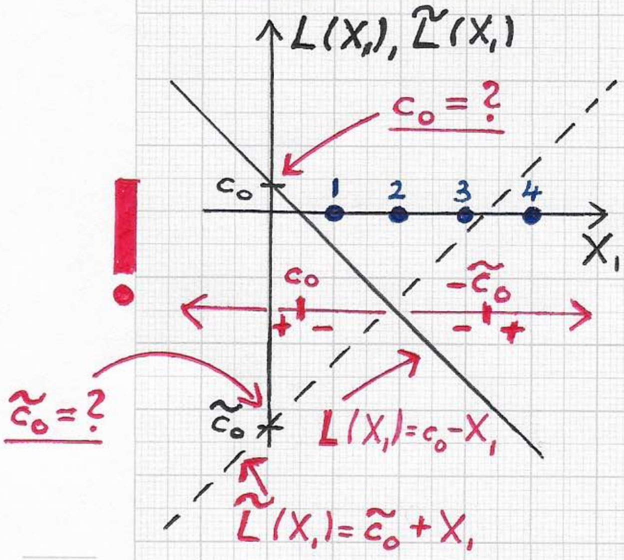
The left figure shows how the X_1-line is split into negative and positive half-spaces at $X_1 = 1 - 1/c_1$ for $c_0 + c_1 = 1$ (at $X_1 = 1 + 1/c_1$ for $c_0 + c_1 = -1$).

• **Note.** We have considered the linear polynomial families $L(X_1) = +/- 1 + c_1 X_1$; $L(X_1) = c_0 +/- 1 \cdot X_1$ and $L(X_1) = (+/- 1 - c_1) + c_1 X_1$. The functions L split the X_1-line at the location X_1 where $L = 0$, as discussed. In all three cases, the positive '+' (negative '−') half-space is the space "left of X_1 where $L = 0$" when the c_1-value is negative (positive).

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

- Laplacian eigenfunctions and neural networks:...



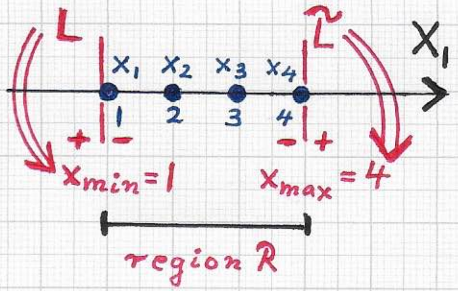
We now consider a first, very simple, motivating example to see how a linear polynomial L can be constructed to define a negative half-space '–' that contains four given points (X_1-values), $x_i = i$, $i=1\ldots4$. For simplicity, we consider the polynomial type $L(X_1) = c_0 - 1 \cdot X_1$. Thus, the goal is to compute an optimal c_0-value. This problem involves four inequalities to be satisfied: $L(x_i) = c_0 - x_i \le 0$, $i = 1\ldots4$, i.e., $c_0 - 1 \le 0$, $c_0 - 2 \le 0$, $c_0 - 3 \le 0$, $c_0 - 4 \le 0$. In addition to these four inequality constraints, one wants to minimize the sum of the squared distances between the x_i-values and the X_1-line split point/location c_0. We define $d_i = d(X_1 = x_i) = L(x_i)/l = L(x_i) = c_0 - x_i$, $i = 1\ldots4$. (Here, $l = 1$.) Thus, the sum to be minimized is $S = (c_0-1)^2 + \ldots + (c_0-4)^2$. In order to minimize this sum, $S' = \frac{d}{dc_0} S(c_0) = 2(c_0-1) + \ldots + 2(c_0-4) = 0$ must hold. The resulting condition is $4c_0 = 10 \Leftrightarrow c_0 = \mathbf{2.5}$. SINCE ALL FOUR INEQUALITIES MUST BE SATISFIED, $c_0 \le 1$ MUST BE SATISFIED – AND THEREFORE $c_0 = 1$ IS THE NEEDED OPTIMAL SOLUTION: $L(X_1) = 1 - X_1$.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... In addition to this first linear polynomial, we must calculate a second polynomial to represent the interval (the "region") on the X_1-line that contains all four x_i-values and has minimal length. The second polynomial is chosen as $\tilde{L}(X_1) = \tilde{c}_0 + 1 \cdot X_1$. The sequence of steps to determine the value of $\tilde{c}_0$ is the same sequence; we summarize only the main steps:

- $\tilde{L}(x_i) = \tilde{c}_0 + x_i \leq 0 \Rightarrow \tilde{c}_0 \leq -1 \wedge \tilde{c}_0 \leq -2 \wedge \tilde{c}_0 \leq -3 \wedge \tilde{c}_0 \leq -4$.
- $S = \sum_{i=1}^{4} (\tilde{c}_0 + x_i)^2 = \sum_{i=1}^{4} (\tilde{c}_0 + i)^2$.
- $d/d\tilde{c}_0 \; S = S' = 2 \cdot (4\tilde{c}_0 + 10)$.
- $S' = 8\tilde{c}_0 + 20 = 0 \Leftrightarrow \tilde{c}_0 = -2.5$
- Condition: $\tilde{c}_0 \leq -1 \wedge \ldots \wedge \tilde{c}_0 \leq -4 \wedge \tilde{c}_0 \leq -2.5 \Rightarrow \tilde{c}_0 = -4$.
- Solution: $\tilde{L}(X_1) = -4 + X_1$.

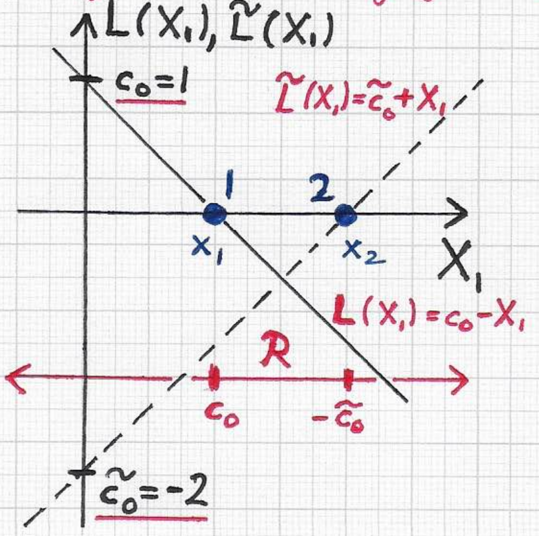


The left sketch illustrates the optimal result. The minimal-length interval, region R, is $R = [x_{min}, x_{max}] = [1, 4]$. The two "boundaries" of R are the two hyper-planes $X_1 = x_{min} = 1$ and $X_1 = x_{max} = 4$, i.e., the X_1-values where the half-space-defining linear polynomials L and $\tilde{L}$ have the value 0. Thus, a value $\underline{x}$ is VIEWED AS A VALUE BELONGING TO THE CLASS REPRESENTED BY R IF $(L(x) \leq 0 \wedge \tilde{L}(x) \leq 0) = \text{TRUE}$.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... We now consider the linear programming problem that one must define and solve to calculate the coefficients c_0 and $\tilde{c}_0$ of the linear polynomials simultaneously – by minimizing just one cost function. We use the even simpler example shown in the left figure, where only two X_1-values define the region R associated with the class of interest, i.e., $X_1 = x_1$ and $X_1 = x_2$.



Using the notation from the previous example, we obtain: 2 inequalities and 1 cost function for $L(X_1) = c_0 - X_1$: $c_0 \leq 1 \wedge c_0 \leq 2 \wedge S = (c_0-1)^2 + (c_0-2)^2 \rightarrow min$

and for $\tilde{L}(X_1) = \tilde{c}_0 + X_1$: $\tilde{c}_0 \leq -1 \wedge \tilde{c}_0 \leq -2 \wedge S = (\tilde{c}_0+1)^2 + (\tilde{c}_0+2)^2 \rightarrow min$.

For simple linear programming problems like these, one can use the **WOLFRAM ALPHA** linear programming solver (widget); for the two minimization problems one would use this input:

$L(X_1)$: Optimize Min; function $5+2c_0^2-6c_0$;
subject to $c_0 <= 1$, $c_0 <= 2$ $\Rightarrow c_0 = 1$.

$\tilde{L}(X_1)$: Optimize Min; function $5+2\tilde{c}_0^2+6\tilde{c}_0$;
subject to $\tilde{c}_0 <= -1$, $\tilde{c}_0 <= -2$ $\Rightarrow \tilde{c}_0 = -2$.

•••

OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

- Laplacian eigenfunctions and neural networks:... Thus, the two linear polynomials defining the two hyper-plane boundaries, and therefore the region/interval R, are $L(X_1) = 1 - X_1$ and $\tilde{L}(X_1) = -2 + X_1$. It is possible to compute the coefficients c_0 and $\tilde{c}_0$ by combining the two univariate linear programming problems into one bivariate linear programming problem. We only must consider all inequality conditions and define the needed cost function via addition:

  Optimize Min ; function $10+2(c_0^2-\tilde{c}_0^2)+6(-c_0+\tilde{c}_0)$;
  subject to $c_0<=1$, $c_0<=2$, $\tilde{c}_0<=-1$, $\tilde{c}_0<=-2$
  $$\Rightarrow (c_0, \tilde{c}_0) = (1, -2).$$

  The fact that it is possible to compute the unknown coefficient values of multiple linear polynomials — that define a "class region R via negative half-spaces" — simultaneously is relevant for our classification application. Thus, the proper definition and solution of the linear programming problem, with all proper inequalities and a proper cost function, is the most important, and also challenging, part of such an approach.

- **Note.** For example, if one is given a point set $\{(x_i, y_i)^T\}_{i=1}^N$ in the (X_1, X_2)-plane and wants to optimally compute three negative half-spaces defining a region R for the point set, one must calculate $L_j(X_1, X_2) = c_0^j + c_1^j X_1 + c_2^j X_2$, $j = 1, 2, 3$, i.e. nine coefficients c.

...