Why does least angle regression work?

Sven Laur
swen@math.ut.ee

Helsinki University of Technology

Minimisation goal of the LASSO algorithm

GIVEN: an output vector $\boldsymbol{y}$ and a design matrix X with columns $\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}$.

FIND: a coefficient vector $\boldsymbol{\beta}$ that minimises

$$E_{\text{lasso}} = \tfrac{1}{2} \cdot \|\boldsymbol{y} - X\boldsymbol{\beta}\|_2^2 + \lambda \cdot \|\boldsymbol{\beta}\|_1 \tag{1}$$

EQUIVALENT FORMULATION: Find a coefficient vector $\boldsymbol{\beta}$ that minimises

$$E_{\text{ols}} = \|\boldsymbol{y} - X\boldsymbol{\beta}\|_2^2 \qquad \text{s.t.} \qquad \|\boldsymbol{\beta}\|_1 \leq t \tag{2}$$

CORRESPONDENCE: Task (1) is Lagrange functional of Task (2).

Explicit gradient of the cost function

Divide the set of feasible solutions $\mathbb{R}^n$ into octants $\operatorname{sign}(\beta_i) = \text{const}$.

Let $\boldsymbol{s}$ be the sign vector, i.e. $s_i = \operatorname{sign}(\beta_i)$. Then in each octant

$$E_{\text{lasso}} = \tfrac{1}{2} \cdot (\boldsymbol{y} - X\boldsymbol{\beta})^2 + \lambda \cdot \boldsymbol{s}^{\mathrm{T}}\boldsymbol{\beta}$$
$$\nabla_{\boldsymbol{\beta}} E_{\text{lasso}} = X^{\mathrm{T}}X\boldsymbol{\beta} - X^{\mathrm{T}}\boldsymbol{y} + \lambda \cdot \boldsymbol{s}$$

If the minimum is an internal point, then the solution has a form

$$\boldsymbol{\beta}^{\star} = (X^{\mathrm{T}}X)^{-1}(X^{\mathrm{T}}\boldsymbol{y} - \lambda \boldsymbol{s})$$

What happens in the boundaries?

For the minimisation over a boundary, we explicitly require

$$\beta_i = 0 \qquad \text{for all} \qquad i \in \mathcal{N}$$
$$\beta_i \in \mathbb{R} \qquad \text{for all} \qquad i \in \mathcal{A}$$

Hence, the cost function simplifies

$$E_{\text{lasso}} = \tfrac{1}{2} \cdot (\boldsymbol{y} - X_{\mathcal{A}} \boldsymbol{\beta}_{\boldsymbol{\mathcal{A}}})^2 + \lambda \cdot \boldsymbol{s}_{\boldsymbol{\mathcal{A}}}{}^{\mathrm{T}} \boldsymbol{\beta}_{\boldsymbol{\mathcal{A}}}$$
$$\nabla_{\boldsymbol{\beta}_{\boldsymbol{\mathcal{A}}}} E_{\text{lasso}} = X_{\mathcal{A}}^{\mathrm{T}} X_{\mathcal{A}} \boldsymbol{\beta}_{\boldsymbol{\mathcal{A}}} - X_{\mathcal{A}}^{\mathrm{T}} \boldsymbol{y} + \lambda \cdot \boldsymbol{s}_{\boldsymbol{\mathcal{A}}}$$

and thus

$$\boldsymbol{\beta}^{\star}_{\boldsymbol{\mathcal{A}}} = (X_{\mathcal{A}}^{\mathrm{T}} X_{\mathcal{A}})^{-1} (X_{\mathcal{A}}^{\mathrm{T}} \boldsymbol{y} - \lambda \boldsymbol{s}_{\boldsymbol{\mathcal{A}}})$$
$$\boldsymbol{\beta}^{\star}_{\boldsymbol{\mathcal{N}}} = \mathbf{0}$$

Geometrical interpretation of $\boldsymbol{\beta}^\star$

If $\boldsymbol{\beta}^\star$ is an internal point, then the corresponding prediction vector

$$\boldsymbol{\mu} = X\boldsymbol{\beta}^\star = X(X^{\mathrm{T}}X)^{-1}(X^{\mathrm{T}}\boldsymbol{y} - \lambda\boldsymbol{s}) = \boldsymbol{\mu}_{\mathsf{ols}} - \lambda \cdot \underbrace{X(X^{\mathrm{T}}X)^{-1}\boldsymbol{s}}_{\boldsymbol{u}}$$

where $\boldsymbol{u}$ is an equiangular to the vectors $s_1\boldsymbol{x_1}, \ldots, s_n\boldsymbol{x_n}$

$$X^{\mathrm{T}}\boldsymbol{u} = X^{\mathrm{T}}X(X^{\mathrm{T}}X)^{-1}\boldsymbol{s} = \boldsymbol{s} = (\pm 1, \ldots, \pm 1)^t$$

To summarise, a small change in λ moves $\boldsymbol{\mu}$ in the direction of $\boldsymbol{u}$.

What happens in the boundaries?

Let $\boldsymbol{\beta}^\star$ be the internal point of a boundary with working set $\mathcal{A}$, i.e.

$$\begin{aligned} \beta_i &= 0 \qquad \text{for all} \qquad i \in \mathcal{N} \\ \beta_i &\neq 0 \qquad \text{for all} \qquad i \in \mathcal{A} \end{aligned}$$

Then the corresponding prediction vector

$$\boldsymbol{\mu} = X_{\mathcal{A}}\boldsymbol{\beta}^\star_{\boldsymbol{\mathcal{A}}} = X_{\mathcal{A}}(X_{\mathcal{A}}^{\mathrm{T}}X_{\mathcal{A}})^{-1}(X_{\mathcal{A}}^{\mathrm{T}}\boldsymbol{y} - \lambda \boldsymbol{s}_{\boldsymbol{\mathcal{A}}}) = \boldsymbol{\mu}_{\boldsymbol{\mathcal{A}}} - \lambda \cdot \underbrace{X_{\mathcal{A}}(X_{\mathcal{A}}^{\mathrm{T}}X_{\mathcal{A}})^{-1}\boldsymbol{s}_{\boldsymbol{\mathcal{A}}}}_{\boldsymbol{u}_{\boldsymbol{\mathcal{A}}}}$$

where $\boldsymbol{u}_{\boldsymbol{\mathcal{A}}}$ is equiangular to the vectors $s_i\boldsymbol{x}_{\boldsymbol{i}},\ i \in \mathcal{A}$

$$X_{\mathcal{A}}^{\mathrm{T}}\boldsymbol{u}_{\boldsymbol{\mathcal{A}}} = X^{\mathrm{T}}X_{\mathcal{A}}(X_{\mathcal{A}}^{\mathrm{T}}X_{\mathcal{A}})^{-1}\boldsymbol{s}_{\boldsymbol{\mathcal{A}}} = \boldsymbol{s}_{\boldsymbol{\mathcal{A}}} = (\pm 1, \ldots, \pm 1)^t$$

To summarise, a small change in λ moves $\boldsymbol{\mu}$ in the direction of $\boldsymbol{u}_{\boldsymbol{\mathcal{A}}}$.

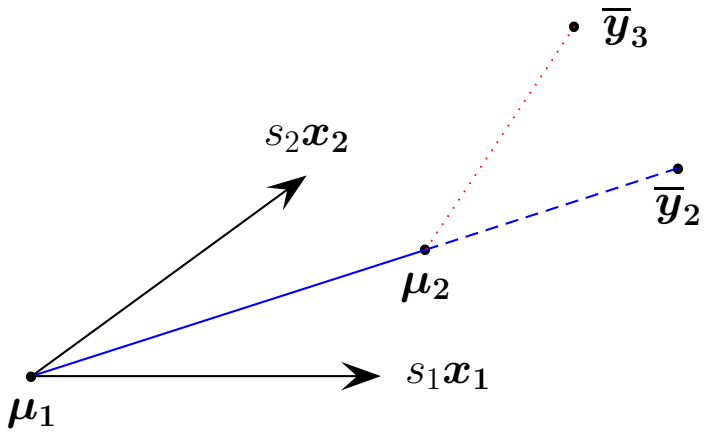
Informal description of LARS

LARS is a greedy optimisation algorithm:

- Starts from the extreme boundary: $\mathcal{A} = \emptyset$, $\boldsymbol{\beta_0} = \mathbf{0}$ and $\boldsymbol{\mu_0} = \mathbf{0}$.
- Moves along the "optimal" path in space vector space $\langle \boldsymbol{x_i}, i \in \mathcal{A} \rangle$.
- Occasionally, extends to higher dimension.
- Always chooses the most profitable vector $\boldsymbol{x_i}$ to add.
- Finally, reaches the ordinary least squares solution.

W.l.o.g. we can assume that the working set $\mathcal{A}_k = \{1, \ldots, k\}$.

The LARS path. Steady phase

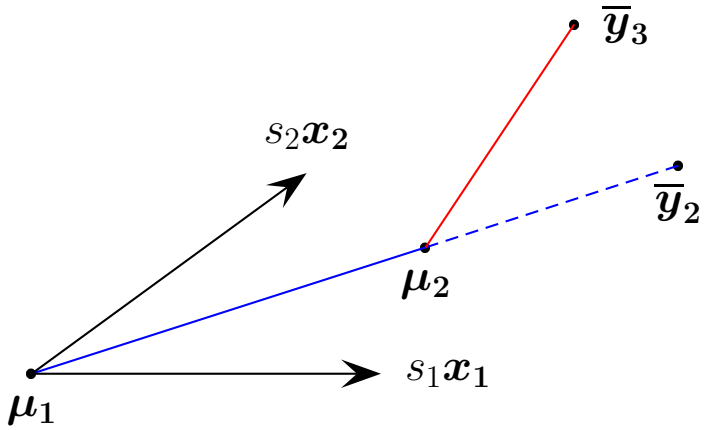


Equiangular vector $u_k = \overline{y}_k - \mu_{k-1}$ and $\mu(\gamma)$ moves along $\mu_k \rightarrow \mu_{k+1}$

$$\mu(\gamma) = \mu_{k-1} + \gamma u_k = \overline{y}_k - \gamma u_k$$

Parameter λ decreases and t increases in the path.

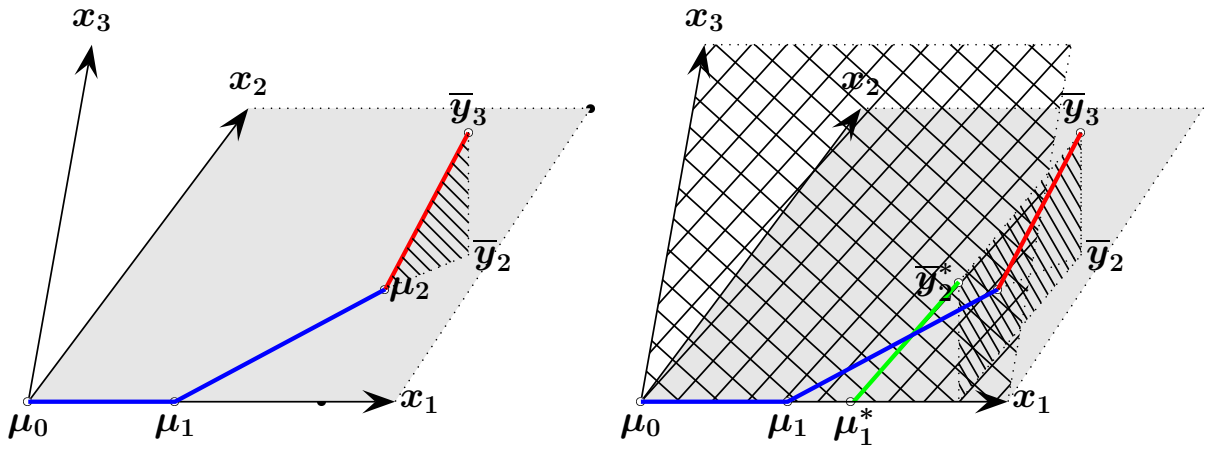
The LARS path. Regime change



Two paths $\boldsymbol{\mu}(\gamma) = \boldsymbol{y_k} - \gamma \boldsymbol{u_k}$ and $\boldsymbol{\mu}(\gamma) = \overline{\boldsymbol{y}}_{\boldsymbol{k+1}} - \gamma \boldsymbol{u_{k+1}}$ intersect at $\boldsymbol{\mu_k}$.

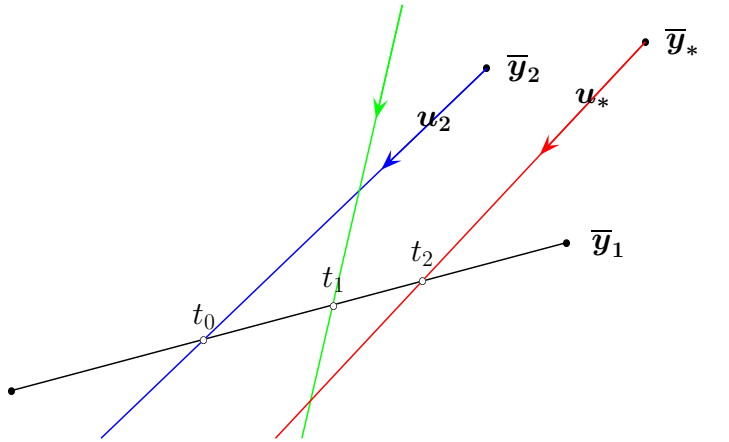
- Therefore, $\boldsymbol{c}(\gamma) = X^{\mathrm{T}}(\boldsymbol{y} - \boldsymbol{\mu}(\gamma))$ must have $k+1$ equal components.

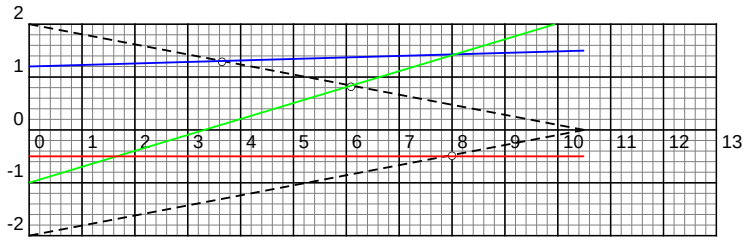
The LARS path. Greedy nature



The LARS algorithm chooses the most advantageous dimension to extend.

Guessing the correct take-off point





The first $c_j(\gamma)$ that intersects with the boundary reveals the next vector.

Guessing the correct take-off point

Consider a the LARS in steady phase. The path point $\boldsymbol{\mu}(\gamma) = \boldsymbol{\mu_{k-1}} + \gamma \boldsymbol{u_k}$ can belong to the optimal path for $\langle \boldsymbol{x_1}, \ldots, \boldsymbol{x_{k+1}} \rangle$ iff

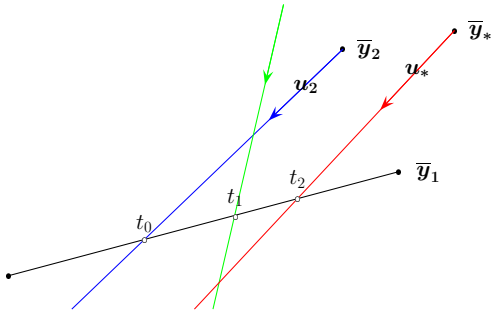
$$\begin{aligned} c_{k+1} &= \boldsymbol{x}_{\boldsymbol{k+1}}^{\mathrm{T}}(\boldsymbol{y} - \boldsymbol{\mu}(\gamma)) = \boldsymbol{x}_{\boldsymbol{k+1}}^{\mathrm{T}}(\overline{\boldsymbol{y}}_{\boldsymbol{k+1}} - \boldsymbol{\mu}(\gamma)) = c_{k+1} - \gamma a_{k+1} \\ c_k &= \boldsymbol{x}_{\boldsymbol{k}}^{\mathrm{T}}(\boldsymbol{y} - \boldsymbol{\mu}(\gamma)) \quad = \boldsymbol{x}_{\boldsymbol{k}}^{\mathrm{T}}(\overline{\boldsymbol{y}}_{\boldsymbol{k+1}} - \boldsymbol{\mu}(\gamma)) \quad = c^\star - \gamma a_k \end{aligned}$$

are equal.

The LARS algorithm chooses the next vector in a greedy way

$$\begin{aligned} \widehat{\gamma} &= \quad \min \left\{ \gamma > 0 : |c^\star - \gamma a_k| = |c_j - \gamma a_j| \, , j \in \mathcal{N} \right\} \\ \widehat{j} &= \operatorname{argmin} \left\{ \gamma > 0 : |c^\star - \gamma a_k| = |c_j - \gamma a_j| \, , j \in \mathcal{N} \right\} \end{aligned}$$

Guessing the correct take-off point



We have to minimise $\|\boldsymbol{y} - X\boldsymbol{\beta}\|_2^2$ w.r.t. $\|\beta\|_1 \leq t$

- Moving along the line increases t.
- Choosing the first take-off point guarantees that, we remain on optimum line after the direction change.
- Formally, for any t the covariance vector $\boldsymbol{c}(t) = X^{\mathrm{T}}(\boldsymbol{y} - \boldsymbol{\mu}(t))$ satisfies

$$|c_j(t)| \leq |c_i(t)| \qquad i \in \mathcal{A} \quad j \in \mathcal{N}$$

What about signs of $\boldsymbol{\beta_{k+1}}$?

If the LARS extends to next dimension it must correctly guess the signs

$$\boldsymbol{\mu}(\lambda) = \boldsymbol{\mu_{\mathcal{A}}} - \lambda \cdot \underbrace{X_{\mathcal{A}}(X_{\mathcal{A}}^{\mathrm{T}} X_{\mathcal{A}})^{-1} \boldsymbol{s_{\mathcal{A}}}}_{\boldsymbol{u_{\mathcal{A}}}}$$

- Sign variables s_i for $i = 1, \ldots, k$ are known from previous step.
- In steady phase $|\beta_{k+1}(\lambda)|$ grows monotonically—caused by equiangularity.

$$X^{\mathrm{T}} \underbrace{X X^{\mathrm{T}} (X^{\mathrm{T}} X)^{-1} s_j \boldsymbol{e_j}}_{\boldsymbol{v_j}} = s_j \boldsymbol{e_j} \qquad \Rightarrow \qquad \boldsymbol{x_i} \perp \boldsymbol{v_j} \quad \text{and} \quad s_j \boldsymbol{x_j} \uparrow\uparrow \boldsymbol{v_j}$$

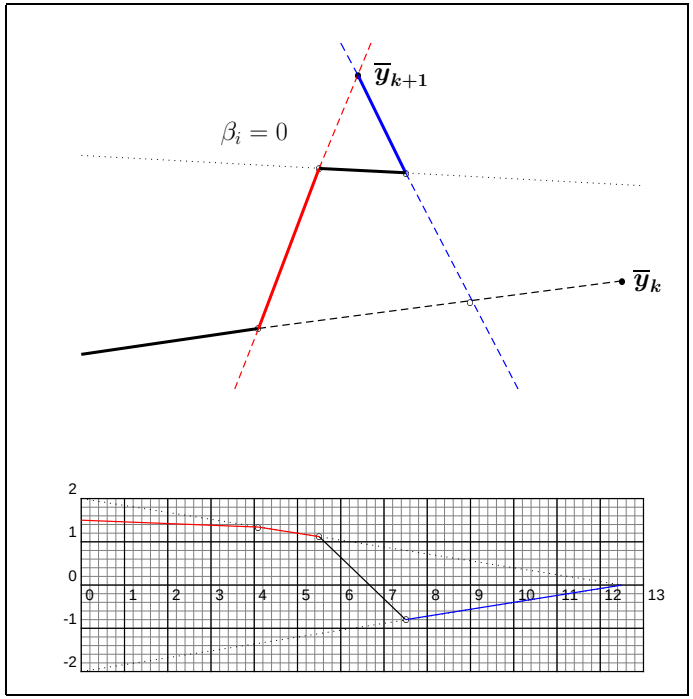
$$\boldsymbol{u_{k+1}} = \boldsymbol{v_1} + \cdots + \boldsymbol{v_{k+1}}$$

I.e. $c_{k+1} = \boldsymbol{x_{k+1}}^{\mathrm{T}} (\boldsymbol{y} - \boldsymbol{\mu_k})$ has the same sign as s_{k+1}.

- Unfortunately, this is not true for other coordinates.

When does the LARS and LASSO coincide

If if the design matrix is orthogonal, we might be a sign problem.



The take-off point is correct, but there are more turns in the path.

Quick fix to LARS algorithm

Check for sign changes:

- Compute $\boldsymbol{\beta_k}$ and $\boldsymbol{\beta_{k+1}}$
- If there is no sign change, i.e. $\beta_{k,i}\beta_{k+1,i} \geq 0$, proceed as usual.
- Otherwise, find largest intermediate vector $\boldsymbol{\beta}$ such that $\beta_{k,i}\beta_{k+1,i} \geq 0$.
  - Find corresponding $\boldsymbol{\mu}$ and store it as $\boldsymbol{\mu_{k+1}}$.
  - Remove $k+1$ from the working list. Recompute direction $\boldsymbol{u}$.
  - Proceed with the next LARS step.

Quick recap to Stagewise algorithm

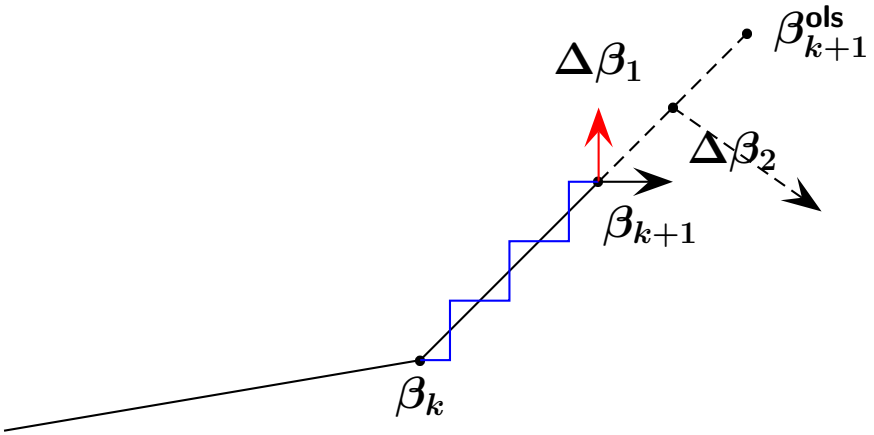
Let $\varepsilon > 0$ be small enough (infinitely small).

- Choose the coordinate i that has the biggest impact on squared error. Make a ϵ-step in appropriate direction towards c_i.

$$\nabla_{\boldsymbol{\beta}} E_{\text{sqe}} = X^{\mathrm{T}}(X\boldsymbol{\beta} - \boldsymbol{y}) = X^{\mathrm{T}}(\boldsymbol{\mu} - \boldsymbol{y}) = -\boldsymbol{c}$$

- If we take infinitesimal steps, we follow the minimising path, except:
  - active correlations c_j and $\Delta\beta_j$ have same sign.
- **Fix to the LARS algorithm.** When a jth coordinate of $\boldsymbol{u_{k+1}}$ has different sign than $\boldsymbol{c}$, determine active Stagewise coordinates with a projection. Update working list $\mathcal{A}$ and the vector $\boldsymbol{u_{\mathcal{A}}}$.

Why does the fix work?



The Stagewise algorithm with infinitesimal $\epsilon > 0$ assures that

$$|c_j(t)| \leq |c_i(t)| \qquad i \in \mathcal{A} \quad j \in \mathcal{N} \qquad \text{and} \qquad \Delta\beta_i c_i \geq 0$$

Projection is a clever way to determine active working set.

Final remarks

- The LASSO algorithm minimises the true objective, but sometimes make more steps than the LARS algorithm.
- The LARS skips several LASSO steps. Hopefully, the LASSO and LARS paths are different for small regions.
- The Stagewise algorithm provides the most heuristic approach, but is more widely applicable.
- For large datasets (design matrices) they all perform relatively similarly.

  The column vectors of the large design matrix are almost orthogonal *with high probability.*