

Lec 14: Specialized Optimization Methods I

15-369/669/769: Numerical Computing

Instructor: Minchen Li

Table of Content

- Nonlinear Least-Squares
- Iteratively Reweighted Least-Squares
- Coordinate Descent and Alternation

Table of Content

- Nonlinear Least-Squares
- Iteratively Reweighted Least-Squares
- Coordinate Descent and Alternation

Nonlinear Least-Squares

Formulation

Example (exp. regression). Fit $y = ce^{ax}$ to data $\{(x_i, y_i)\}_{i=1}^k$:

$$E(a, c) := \sum_{i=1}^k (y_i - ce^{ax_i})^2.$$

General NLS. Given residuals $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$ and $\mathbf{x} \in \mathbb{R}^n$,

$$E_{\text{NLS}}(\mathbf{x}) := \frac{1}{2} \sum_{i=1}^k [f_i(\mathbf{x})]^2 = \frac{1}{2} \|F(\mathbf{x})\|_2^2, \quad F(\mathbf{x}) := [f_1(\mathbf{x}) \quad \cdots \quad f_k(\mathbf{x})]^\top.$$

Goal: find \mathbf{x}^* with $f_i(\mathbf{x}^*) \approx 0$ for all i .

Nonlinear Least-Squares

Gauss–Newton: Linearization and Quadratic Model

- Linearize each residual at current iterate \mathbf{x}_0 : $f_i(\mathbf{x}) \approx f_i(\mathbf{x}_0) + \nabla f_i(\mathbf{x}_0)^\top (\mathbf{x} - \mathbf{x}_0)$.
- Stacking gives with Jacobian $DF(\mathbf{x}_0) \in \mathbb{R}^{k \times n}$, $F(\mathbf{x}) \approx F(\mathbf{x}_0) + DF(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0)$.
- Approximate NLS objective: $E_{\text{NLS}}^0(\mathbf{x}) = \frac{1}{2} \|F(\mathbf{x}_0) + DF(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0)\|_2^2$.
- **Minimizer update (normal equations):** $\mathbf{x} = \mathbf{x}_0 - (DF(\mathbf{x}_0)^\top DF(\mathbf{x}_0))^{-1} DF(\mathbf{x}_0)^\top F(\mathbf{x}_0)$.
- **Iterative Gauss–Newton:** $\mathbf{x}_{k+1} = \mathbf{x}_k - (DF(\mathbf{x}_k)^\top DF(\mathbf{x}_k))^{-1} DF(\mathbf{x}_k)^\top F(\mathbf{x}_k)$
- *Notes:* solves a linear least-squares each step; can achieve quadratic convergence near solution when residuals are small; otherwise may stagnate/diverge.

Nonlinear Least-Squares

Levenberg–Marquardt (LM): Trust-Region View

Trust-region on linearized model

$$\min_{\delta \mathbf{x}} \frac{1}{2} \|F(\mathbf{x}_0) + J_0 \delta \mathbf{x}\|_2^2 \quad \text{s.t.} \quad \|\delta \mathbf{x}\|_2^2 \leq \Delta,$$

with $J_0 = DF(\mathbf{x}_0)$. Let $H = J_0^\top J_0$, $\mathbf{g} = J_0^\top F(\mathbf{x}_0)$.

$$\text{Stationarity: } \mathbf{0} = H\delta \mathbf{x} + DF(\mathbf{x}_0)^\top F(\mathbf{x}_0) + 2\mu\delta \mathbf{x}$$

$$\text{Primal feasibility: } \|\delta \mathbf{x}\|_2^2 \leq \Delta$$

$$\text{Complementary slackness: } \mu(\Delta - \|\delta \mathbf{x}\|_2^2) = 0$$

$$\text{Dual feasibility: } \mu \geq 0.$$

Define $\lambda := 2\mu$. Then, the stationarity condition can be written as follows:

$$(H + \lambda I_{n \times n})\delta \mathbf{x} = -DF(\mathbf{x}_0)^\top F(\mathbf{x}_0).$$

Nonlinear Least-Squares

LM: Tikhonov View and Practical Update

- LM assumes the constraint is active, and starts the iterative process with a user specified $\lambda > 0$:

$$\boxed{(H + \lambda I) \delta \mathbf{x} = -\mathbf{g}} \quad \Rightarrow \quad \mathbf{x}_{k+1} = \mathbf{x}_k + \delta \mathbf{x}.$$

Adaptive damping λ_k :

- Increase λ_k if the linearized model poorly predicts decrease (shrink trust region).
- Decrease λ_k when the step is successful (expand region).

Interpretation: damped Gauss–Newton; λ controls step length (small $\lambda \Rightarrow$ GN; large $\lambda \Rightarrow$ gradient descent).

Rule of thumb: $\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2^2 \approx \Delta$ corresponds to a posteriori choice of λ from KKT.

Equivalently, LM solves a **Tikhonov-regularized** linear system:

$$\boxed{\mathbf{x}_{k+1} = \mathbf{x}_k - (J_k^\top J_k + \lambda_k I)^{-1} J_k^\top F(\mathbf{x}_k)}.$$

Nonlinear Least-Squares

GN and LM Remarks

- Both are methods that computes search directions, can be used with line search.
- Can be applied to general nonlinear optimization problems: e.g.
 - $\min_{\mathbf{x}} f(g(\mathbf{x}))$, with $f: \mathbb{R}^m \rightarrow \mathbb{R}$, $g: \mathbb{R}^n \rightarrow \mathbb{R}^m$
 - Gradient: $(\nabla g(\mathbf{x}))^T \nabla_g f(g(\mathbf{x}))$
 - Hessian: $(\nabla g(\mathbf{x}))^T \nabla_g^2 f(g(\mathbf{x})) \nabla g(\mathbf{x}) + \sum_i^m (\nabla_g f(g(\mathbf{x})))_i \nabla^2 g_i(\mathbf{x})$
 - GN approximation: $(\nabla g(\mathbf{x}))^T \nabla_g^2 f(g(\mathbf{x})) \nabla g(\mathbf{x})$, assuming g is linear
 - LM approximation: $(\nabla g(\mathbf{x}))^T \nabla_g^2 f(g(\mathbf{x})) \nabla g(\mathbf{x}) + \lambda I$.

Table of Content

- Nonlinear Least-Squares
- Iteratively Reweighted Least-Squares
- Coordinate Descent and Alternation

Iteratively Reweighted Least-Squares

Formulation

Goal: minimize a weighted least-squares energy

$$E_{\text{IRLS}}(\mathbf{x}) := \sum_i f_i(\mathbf{x}) [g_i(\mathbf{x})]^2,$$

where $f_i(\mathbf{x})$ acts as a **weight** on residual $g_i(\mathbf{x})$.

Example (L^p least-squares). Given $A \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$,

$$E_p(\mathbf{x}) := \|A\mathbf{x} - \mathbf{b}\|_p^p = \sum_i |a_i \cdot \mathbf{x} - b_i|^p = \sum_i |a_i \cdot \mathbf{x} - b_i|^{p-2} (a_i \cdot \mathbf{x} - b_i)^2.$$

Define: $f_i(\mathbf{x}) = |a_i \cdot \mathbf{x} - b_i|^{p-2}$, $g_i(\mathbf{x}) = a_i \cdot \mathbf{x} - b_i$.

Then $E_p = E_{\text{IRLS}}$. For fixed \mathbf{x}_k , the next iterate solves

$$\mathbf{x}_{k+1} = \arg \min_{\mathbf{x}} \sum_i f_i(\mathbf{x}_k) [g_i(\mathbf{x})]^2.$$

→ *A weighted least-squares problem in \mathbf{x} at each iteration.*

Iteratively Reweighted Least-Squares

Example: L1 Optimization

For $p = 1$,

$$E_1(\mathbf{x}) = \sum_i |a_i \cdot \mathbf{x} - b_i| = \sum_i \frac{(a_i \cdot \mathbf{x} - b_i)^2}{|a_i \cdot \mathbf{x} - b_i|}.$$

The IRLS update (with small $\delta > 0$ to avoid division by zero):

$w_i \leftarrow [\max(a_i \cdot \mathbf{x} - b_i , \delta)]^{-1}$	▷ Recompute weights
$\mathbf{x} \leftarrow (A^\top \text{diag}(\mathbf{w})A)^{-1} A^\top \text{diag}(\mathbf{w})\mathbf{b}$	▷ Weighted LS solve.

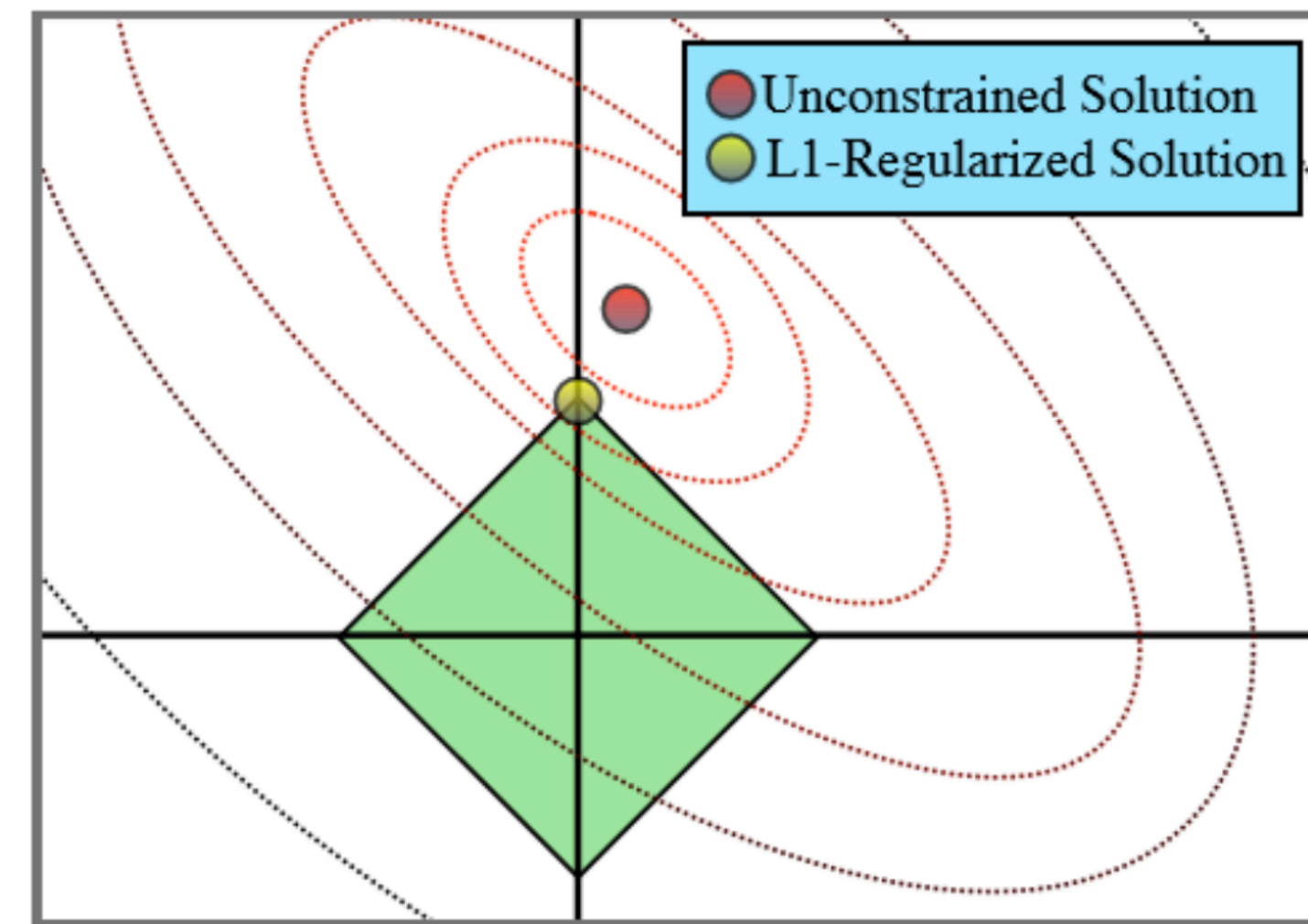
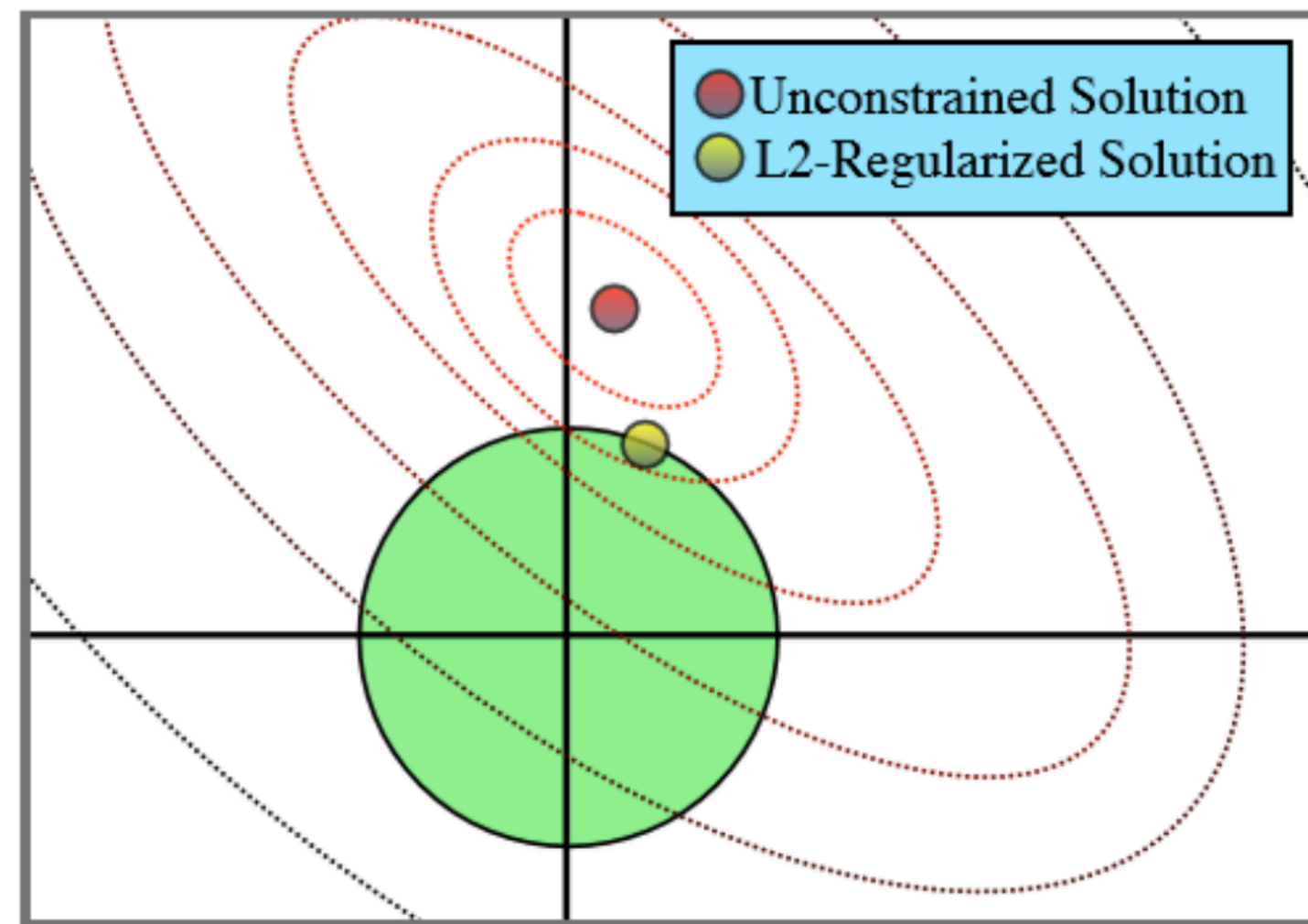
Interpretation:

- Small residuals \Rightarrow large weights w_i .
- Larger $\delta \Rightarrow$ more stable but less accurate L^1 approximation.
- Each step solves a linear least-squares system.

Iteratively Reweighted Least-Squares

Example: L1 Regularization

- $\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|_p^p + \lambda \|\mathbf{x}\|_1$
- L2-regularization conceptually restricts \mathbf{x} to a ball.
- L1-regularization restricts to the L1 “ball”: – Solutions tend to be at corners where \mathbf{x}_i are zero.



Iteratively Reweighted Least-Squares

Example: Weiszfeld Algorithm (Geometric Median)

- **Objective:** $E(\mathbf{x}) = \sum_i \|\mathbf{x} - \mathbf{x}_i\|_2$.

- Equivalent weighted LS form: $E(\mathbf{x}) = \sum_i \frac{\|\mathbf{x} - \mathbf{x}_i\|_2^2}{\|\mathbf{x} - \mathbf{x}_i\|_2}$.

- **IRLS formulation:**

$$\begin{aligned} w_i &\leftarrow [\max(\|\mathbf{x} - \mathbf{x}_i\|_2, \delta)]^{-1} &> \text{Recompute weights} \\ \mathbf{x} &\leftarrow \arg \min_{\mathbf{x}} \sum_i w_i \|\mathbf{x} - \mathbf{x}_i\|_2^2 &> \text{Weighted LS} \end{aligned}$$

- **Closed-form solution:** $\mathbf{x} = \frac{\sum_i w_i \mathbf{x}_i}{\sum_i w_i}$.

- **Weiszfeld iteration:**

$$\begin{aligned} w_i &\leftarrow [\max(\|\mathbf{x} - \mathbf{x}_i\|_2, \delta)]^{-1}, \\ \mathbf{x} &\leftarrow \frac{\sum_i w_i \mathbf{x}_i}{\sum_i w_i}. \end{aligned}$$

Simple, parallelizable, no Hessians or gradients needed.

Iteratively Reweighted Least-Squares

Properties and Remarks

- Each IRLS step solves a (weighted) least-squares problem: $\min_{\mathbf{x}} \sum_i f_i(\mathbf{x}_k) [g_i(\mathbf{x})]^2$.
- Efficient with Cholesky, QR, or CG (if g_i linear).
- Convergence guarantees are limited — often require safeguards:
 - δ prevents division by zero.
 - For L^1 problems, provably convergent under mild conditions.
- Easy to implement, robust to outliers, and useful for sparse and geometric problems.

Table of Content

- Nonlinear Least-Squares
- Iteratively Reweighted Least-Squares
- **Coordinate Descent and Alternation**

Coordinate Descent and Alternation

Formulation

Goal: Minimize $f : \mathbb{R}^{n+m} \rightarrow \mathbb{R}$ where variables can be split as $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^m$.

Alternating minimization:

for $i = 1, 2, \dots$

$\mathbf{x}_{i+1} \leftarrow \arg \min_{\mathbf{x}} f(\mathbf{x}, \mathbf{y}_i)$ \triangleright Optimize \mathbf{x} with \mathbf{y} fixed

$\mathbf{y}_{i+1} \leftarrow \arg \min_{\mathbf{y}} f(\mathbf{x}_{i+1}, \mathbf{y})$ \triangleright Optimize \mathbf{y} with \mathbf{x} fixed.

Each step reduces $f(\mathbf{x}_i, \mathbf{y}_i)$ monotonically (if minimizations are exact), though convergence to the global minimum is not guaranteed.

Coordinate Descent and Alternation

When to Use Alternating Optimization

Advantages:

- Each subproblem (over \mathbf{x} or \mathbf{y}) may have **lower dimension** or **simpler structure**.
- Efficient updates possible when one variable's minimization has closed-form.

Example: Generalized PCA. Given data $X \in \mathbb{R}^{n \times k}$, find $C \in \mathbb{R}^{n \times d}$ and $Y \in \mathbb{R}^{d \times k}$ minimizing

$$\min_{C, Y} \mu(X - CY),$$

where μ is an energy measure:

- $\mu(M) = \|M\|_F^2$: classical PCA (**Alternating Least Squares, ALS**).
- $\mu(M) = \sum_{ij} |M_{ij}|$: **Robust PCA**.

Alternation: fix Y , solve for C ; fix C , solve for Y — each step is a simpler least-squares or linear program.

Coordinate Descent and Alternation

Example: Coordinate Descent

A special case of alternation where $f = f(x_1, \dots, x_n)$ and we update each coordinate:

$$x_i \leftarrow \arg \min_{x_i} f(x_1, \dots, x_i, \dots, x_n).$$

For least-squares: minimize $\|A\mathbf{x} - \mathbf{b}\|_2^2$. Let columns of A be $\mathbf{a}_1, \dots, \mathbf{a}_n$, then

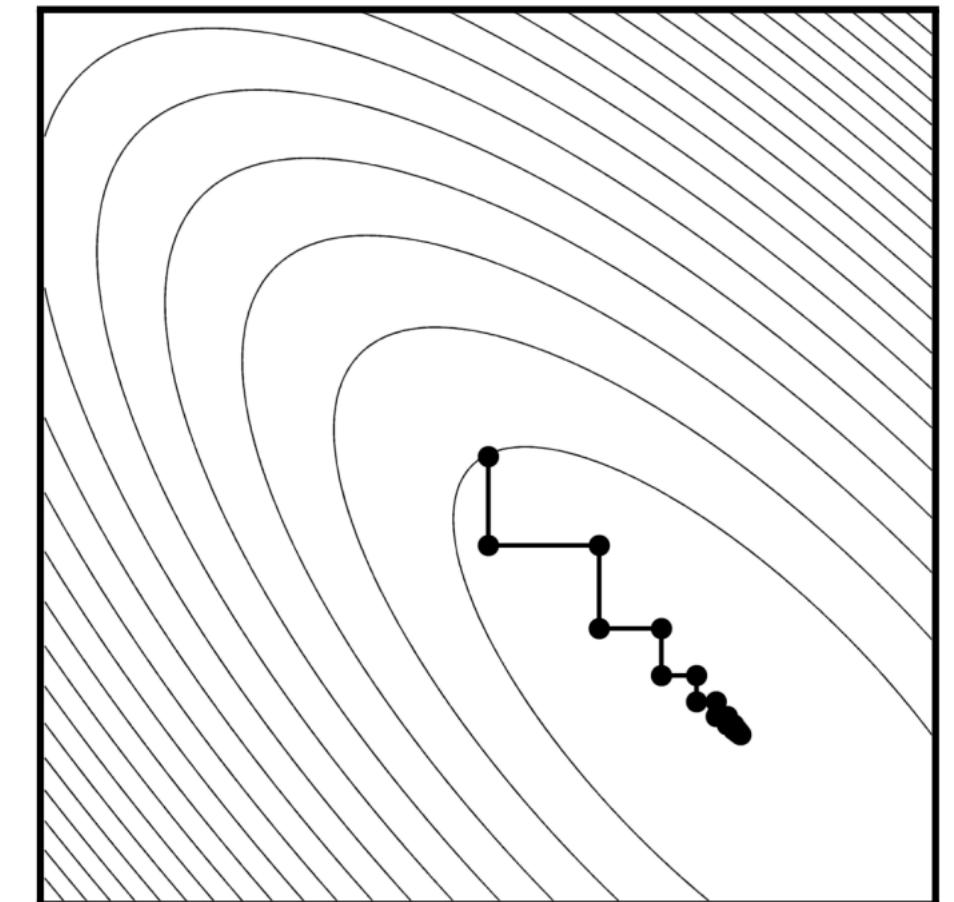
$$0 = \frac{\partial}{\partial x_i} \|A\mathbf{x} - \mathbf{b}\|_2^2 = 2(A\mathbf{x} - \mathbf{b}) \cdot \mathbf{a}_i.$$

Update rule:

$$x_i \leftarrow \frac{\mathbf{a}_i \cdot \mathbf{b} - \sum_{k \neq i} x_k (\mathbf{a}_i \cdot \mathbf{a}_k)}{\|\mathbf{a}_i\|_2^2}.$$

Repeat for $i = 1, \dots, n$ until convergence.

Notes: Lightweight, local updates—efficient for large, tall matrices A ; common in ML (e.g., Lasso, SVM).



Coordinate Descent and Alternation

Example: Alternation in K-Means Clustering

Given data $\mathbf{x}_1, \dots, \mathbf{x}_m \in \mathbb{R}^n$, group them into k clusters with centers $\mathbf{y}_1, \dots, \mathbf{y}_k$:

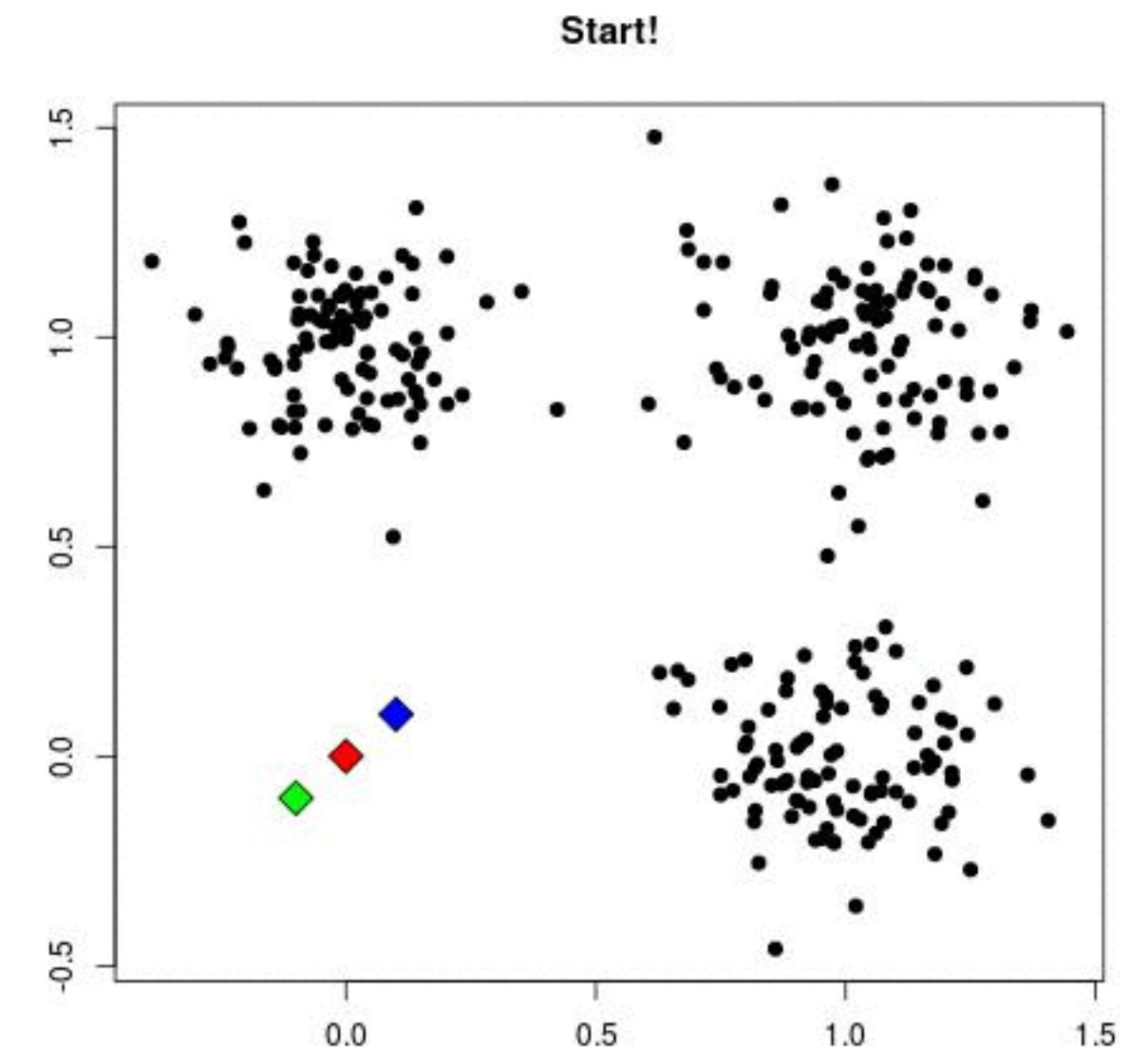
$$E(\mathbf{y}_1, \dots, \mathbf{y}_k) = \sum_{i=1}^m \min_{c \in \{1, \dots, k\}} \|\mathbf{x}_i - \mathbf{y}_c\|_2^2.$$

Introduce assignment variables c_i : $E(\mathbf{y}_1, \dots, \mathbf{y}_k; c_1, \dots, c_m) = \sum_{i=1}^m \|\mathbf{x}_i - \mathbf{y}_{c_i}\|_2^2$.

Alternation:

- Fix $\{c_i\}$, update cluster centers: $\mathbf{y}_j = \frac{\sum_{c_i=j} \mathbf{x}_i}{|\{c_i=j\}|}$.
- Fix $\{\mathbf{y}_j\}$, reassign points: $c_i = \arg \min_{c \in \{1, \dots, k\}} \|\mathbf{x}_i - \mathbf{y}_c\|_2^2$.

Known as the k-means algorithm. Sensitive to initialization; multiple restarts or *k-means++* improve results.



Coordinate Descent and Alternation

Example: Alternation in K-Means Clustering (cont.)

- K-Means++: spreading out the k initial cluster centers to improve results.
- K-Means++ initialization algorithm:
 1. Choose one center uniformly at random among the data points.
 2. For each data point \mathbf{x}_i not chosen yet, compute $d(\mathbf{x}_i)$, the distance between \mathbf{x}_i and its nearest center that has already been chosen.
 3. Choose one new data point at random as a new center, using a weighted probability distribution where a point \mathbf{x} is chosen with probability proportional to $(d(\mathbf{x}))^2$.
 4. Repeat Steps 2 and 3 until k centers have been chosen.
 5. Proceed using standard k-means clustering.

Coordinate Descent and Alternation

Augmented Lagrangians and ADMM

Motivation: Nonlinear constrained optimization

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{s.t. } g(\mathbf{x}) = 0$$

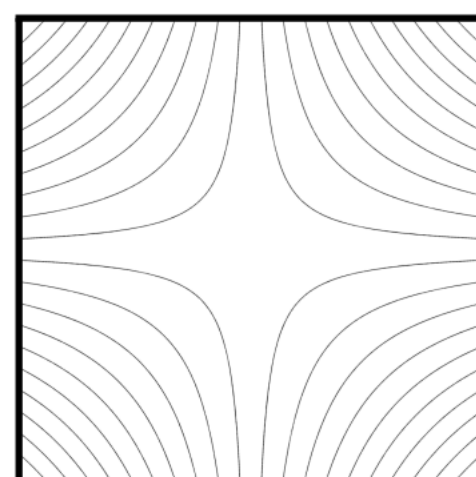
is often difficult — slow convergence, sensitive initialization, and large linear solves.

Quadratic penalty method:

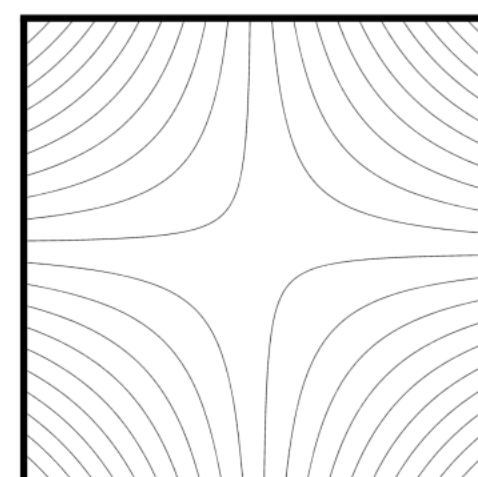
$$f_{\rho}(\mathbf{x}) = f(\mathbf{x}) + \frac{1}{2}\rho\|g(\mathbf{x})\|_2^2.$$

As $\rho \rightarrow \infty$, constraints are satisfied but conditioning worsens — most effort goes into enforcing $g(\mathbf{x}) = 0$ rather than minimizing f .

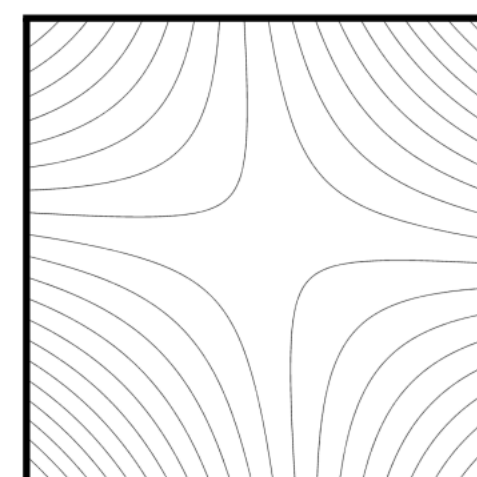
When applied to $\min_{x,y} xy \quad \text{s.t. } x + y = 1$:



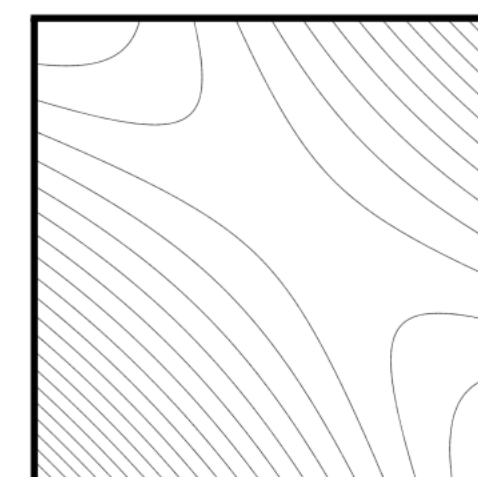
$\rho = 0$



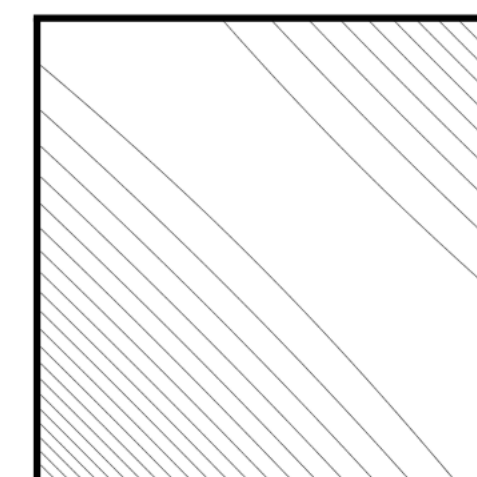
$\rho = 0.01$



$\rho = 0.1$



$\rho = 1$



$\rho = 10$

Coordinate Descent and Alternation

Lagrange Multipliers and the Augmented Lagrangian

Classical Lagrangian:

$$\Lambda(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) - \boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{x}).$$

\mathbf{x} minimizes and $\boldsymbol{\lambda}$ maximizes Λ .

Augmented Lagrangian: combines penalty + multiplier terms:

$$\Lambda_\rho(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \frac{1}{2}\rho\|\mathbf{g}(\mathbf{x})\|_2^2 - \boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{x}).$$

Alternating optimization:

$\boldsymbol{\lambda}_{i+1} \leftarrow \boldsymbol{\lambda}_i - \rho \mathbf{g}(\mathbf{x}_i)$	\triangleright Dual update
$\mathbf{x}_{i+1} \leftarrow \arg \min_{\mathbf{x}} \Lambda_\rho(\mathbf{x}, \boldsymbol{\lambda}_{i+1})$	\triangleright Primal update.

Intuition: $\rho\|\mathbf{g}(\mathbf{x})\|_2^2$ term acts like a “rubber band,” pulling \mathbf{x} closer to the constraint.

Coordinate Descent and Alternation

ADMM: Alternating Direction Method of Multipliers

Problem form:

$$\min_{\mathbf{x}, \mathbf{z}} f(\mathbf{x}) + h(\mathbf{z}) \quad \text{s.t. } A\mathbf{x} + B\mathbf{z} = \mathbf{c}.$$

Augmented Lagrangian:

$$\Lambda_\rho(\mathbf{x}, \mathbf{z}, \boldsymbol{\lambda}) = f(\mathbf{x}) + h(\mathbf{z}) + \frac{1}{2}\rho\|A\mathbf{x} + B\mathbf{z} - \mathbf{c}\|_2^2 + \boldsymbol{\lambda}^\top (A\mathbf{x} + B\mathbf{z} - \mathbf{c}).$$

ADMM iterations:

$\mathbf{x}_{i+1} \leftarrow \arg \min_{\mathbf{x}} \Lambda_\rho(\mathbf{x}, \mathbf{z}_i, \boldsymbol{\lambda}_i)$	▷ x -update
$\mathbf{z}_{i+1} \leftarrow \arg \min_{\mathbf{z}} \Lambda_\rho(\mathbf{x}_{i+1}, \mathbf{z}, \boldsymbol{\lambda}_i)$	▷ z -update
$\boldsymbol{\lambda}_{i+1} \leftarrow \boldsymbol{\lambda}_i + \rho(A\mathbf{x}_{i+1} + B\mathbf{z}_{i+1} - \mathbf{c})$	▷ Dual update.

Splitting \mathbf{x} and \mathbf{z} simplifies each subproblem and allows parallelism.

Coordinate Descent and Alternation

Example: ADMM for Nonnegative Least Squares

Goal: minimize $\|A\mathbf{x} - \mathbf{b}\|_2^2$ subject to $\mathbf{x} \geq 0$.

$$\min_{\mathbf{x}, \mathbf{z}} \|A\mathbf{x} - \mathbf{b}\|_2^2 + h(\mathbf{z}) \quad \text{s.t. } \mathbf{x} = \mathbf{z}, \quad h(\mathbf{z}) = \begin{cases} 0, & \mathbf{z} \geq 0, \\ \infty, & \text{otherwise.} \end{cases}$$

Augmented Lagrangian: $\Lambda_\rho(\mathbf{x}, \mathbf{z}, \boldsymbol{\lambda}) = \|A\mathbf{x} - \mathbf{b}\|_2^2 + h(\mathbf{z}) + \frac{1}{2}\rho\|\mathbf{x} - \mathbf{z}\|_2^2 + \boldsymbol{\lambda}^\top(\mathbf{x} - \mathbf{z})$.

ADMM updates:

$\mathbf{x}_{i+1} \leftarrow (2A^\top A + \rho I)^{-1}(2A^\top \mathbf{b} + \rho \mathbf{z}_i - \boldsymbol{\lambda}_i)$	▷ Least-squares step
$\mathbf{z}_i^0 \leftarrow \boldsymbol{\lambda}_i / \rho + \mathbf{x}_{i+1}$	▷ Unconstrained z candidate
$\mathbf{z}_{i+1} \leftarrow \max(\mathbf{z}_i^0, 0)$	▷ Enforce $\mathbf{z} \geq 0$
$\boldsymbol{\lambda}_{i+1} \leftarrow \boldsymbol{\lambda}_i + \rho(\mathbf{x}_{i+1} - \mathbf{z}_{i+1})$	▷ Dual update.

Simple, efficient, and avoids explicit constrained optimization.

Coordinate Descent and Alternation

Example: ADMM for Geometric Median

Objective: $E(\mathbf{x}) = \sum_{i=1}^N \|\mathbf{x} - \mathbf{x}_i\|_2$.

Reformulation: $\min_{\mathbf{x}, \{\mathbf{z}_i\}} \sum_i \|\mathbf{z}_i\|_2 \quad \text{s.t. } \mathbf{z}_i + \mathbf{x} = \mathbf{x}_i$.

Augmented Lagrangian: $\Lambda_\rho = \sum_i \left[\|\mathbf{z}_i\|_2 + \frac{1}{2}\rho \|\mathbf{z}_i + \mathbf{x} - \mathbf{x}_i\|_2^2 + \boldsymbol{\lambda}_i^\top (\mathbf{z}_i + \mathbf{x} - \mathbf{x}_i) \right]$.

Updates:

$$\mathbf{x} \leftarrow \frac{1}{N} \sum_i (\mathbf{x}_i - \mathbf{z}_i - \frac{1}{\rho} \boldsymbol{\lambda}_i) \quad \triangleright \mathbf{x} \text{ update}$$

$$\mathbf{z}_i \leftarrow t \mathbf{z}_i^0, \quad t = \max\left(0, 1 - \frac{1}{\rho \|\mathbf{z}_i^0\|_2}\right), \quad \mathbf{z}_i^0 = -\frac{1}{\rho} \boldsymbol{\lambda}_i + \mathbf{x}_i - \mathbf{x} \quad \triangleright \mathbf{z} \text{ update}$$

$$\boldsymbol{\lambda}_i \leftarrow \boldsymbol{\lambda}_i + \rho (\mathbf{z}_i + \mathbf{x} - \mathbf{x}_i) \quad \triangleright \text{Dual update.}$$

ADMM decouples and parallelizes \mathbf{z}_i updates. Converges robustly even for nonconvex settings.

Table of Content

- Nonlinear Least-Squares
- Iteratively Reweighted Least-Squares
- Coordinate Descent and Alternation