02610
Optimization and Data Fitting
Week 8: More on Data Fitting

Yiqiu Dong

DTU Compute
Technical University of Denmark

Lecture Material

- Exponential data fitting
  - P. C. Hansen, V. Pereyra and G. Scherer, *Least Squares Data Fitting with Applications*, Johns Hopkins University Press.
  - Chapter 9: Algorithms for solving nonlinear LSQ problems.
  - We cover: 9.6.

- Data Fitting in other norms
  - K. Madsen and H. B. Nielsen, *Introducetion to Optimization and Data Fitting*, lecture notes, 2010.
  - Chapter 7: Fitting in other norms.
  - We cover: 7.1, 7.2, and 7.3.1.

Fit with an exponential model

Problem: Given the data (t_i, y_i) with $i = 1, \cdots, m$ and all $y_i > 0$, we want to fit a nonlinear exponential model

$$\phi(c, a; t) = ce^{at}$$

to the data, i.e., we want to find c and a such that

$$y_i \approx ce^{at_i}, \quad i = 1, \cdots, m.$$

Modification: Taking the natural logarithm on both sides, we get

$$\log y_i \approx \log c + at_i, \quad i = 1, \cdots, m.$$

$$\mathbf{x} = \begin{bmatrix} \log c \\ a \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} \log y_1 \\ \vdots \\ \log y_m \end{bmatrix}, \quad \text{and} \quad A = \begin{bmatrix} 1 & t_1 \\ \vdots & \vdots \\ 1 & t_m \end{bmatrix},$$

and the linear LSQ data fitting problem is

$$\min_{\mathbf{x}} \|\mathbf{y} - A\mathbf{x}\|_2^2.$$

Fit with multiexponential model

Problem: Given the data (t_i, y_i) with $i = 1, \cdots, m$, we want to fit the data with the model

$$\phi(\mathbf{c}, \mathbf{a}; t) = \sum_{j=1}^{n} c_j e^{a_j t}.$$

Note that

- the elements in the unknown vector **c** appear linearly.
- the elements in the unknown vector **a** appear nonlinearly.

LSQ problem

The least-squares fit gives the problem:

$$\min_{\mathbf{c},\mathbf{a}} \|\mathbf{y} - \phi(\mathbf{c}, \mathbf{a}; \mathbf{t})\|_2^2$$

Define

$$F(\mathbf{a}) = \begin{bmatrix} e^{a_1 t_1} & e^{a_2 t_1} & \cdots & e^{a_n t_1} \\ e^{a_1 t_2} & e^{a_2 t_2} & \cdots & e^{a_n t_2} \\ \vdots & \vdots & \vdots & \vdots \\ e^{a_1 t_m} & e^{a_2 t_m} & \cdots & e^{a_n t_m} \end{bmatrix}.$$

Then, the LSQ problem can be written as

$$\min_{\mathbf{c},\mathbf{a}} \|\mathbf{y} - F(\mathbf{a})\mathbf{c}\|_2^2.$$

- With respect to $\mathbf{c}$, it is a linear LSQ problem.
- With respect to $\mathbf{a}$, it is a nonlinear LSQ problem.

Given $\mathbf{a}$

Then, the unknown vector $\mathbf{c}$ can be obtained by solving the linear LSQ data fitting problem

$$\min_{\mathbf{c}} \|\mathbf{y} - F(\mathbf{a})\mathbf{c}\|_2^2,$$

i.e., the minimizer $\mathbf{c}^*$ should satisfy the normal equation

$$F(\mathbf{a})^T F(\mathbf{a})\mathbf{c} = F(\mathbf{a})^T \mathbf{y}.$$

If $F(\mathbf{a})$ has full column rank, then we have

$$\mathbf{c}(\mathbf{a})^* = \left(F(\mathbf{a})^T F(\mathbf{a})\right)^{-1} F(\mathbf{a})^T \mathbf{y} := F(\mathbf{a})^\dagger \mathbf{y}.$$

- $F(\mathbf{a})^\dagger$ is the Moore-Penrose pseudoinverse of $F(\mathbf{a})$, where $F(\mathbf{a})$ can be ill-conditioned or even rank-deficient.

Variable projection

Substituting $\mathbf{c}(\mathbf{a})^* = F(\mathbf{a})^\dagger \mathbf{y}$ into

$$\min_{\mathbf{c},\mathbf{a}} \|\mathbf{y} - F(\mathbf{a})\mathbf{c}\|_2^2$$

gives the problem

$$\min_{\mathbf{a}} \left\| \left(I - F(\mathbf{a})F(\mathbf{a})^\dagger \right) \mathbf{y} \right\|_2^2 .$$

- $\underbrace{\left(I - F(\mathbf{a})F(\mathbf{a})^\dagger \right) F(\mathbf{a})}_{\text{Orthogonal matrices}} = 0.$
- $\left(I - F(\mathbf{a})F(\mathbf{a})^\dagger \right)$ is a projector onto the orthogonal complement of the column space of $F(\mathbf{a})$.
- $\mathbf{r}_{VP}(\mathbf{a}) = \mathbf{y} - F(\mathbf{a})\mathbf{c}(\mathbf{a})^* = \left(I - F(\mathbf{a})F(\mathbf{a})^\dagger \right) \mathbf{y}$ is called the variable projection of $\mathbf{y}$.
- Now, we only need solve the original problem on a space of smaller dimension, i.e., only on $\mathbf{a}$.

Jacobian matrix

To use nonlinear solvers like Levenberg-Marquardt method to solve

$$\min_{\mathbf{a}} \|\mathbf{r}_{VP}(\mathbf{a})\|_2^2 = \left\| \left(I - F(\mathbf{a})F(\mathbf{a})^\dagger \right) \mathbf{y} \right\|_2^2$$

we would need the Jacobian $J(\mathbf{a})$ of the vector function $\mathbf{r}_{VP}(\mathbf{a})$, which has the entries

$$[J(\mathbf{a})]_{ij} = \frac{\partial r_i(\mathbf{a})}{\partial a_j} \qquad i = 1, \ldots, m, \quad j = 1, \ldots, n.$$

$$J = -F\left(F^T F\right)^{-1}\left(\Lambda_{H^T \mathbf{r}_{VP}} - H^T F \Lambda_{\mathbf{c}}\right) - H\Lambda_{\mathbf{c}}$$

- $H = \Lambda_{\mathbf{t}} F$.
- $\Lambda_{\mathbf{c}}$ denotes a diagonal matrix with the vector $\mathbf{c}$ on the main diagonal.
- $\mathbf{c} = F^\dagger \mathbf{y}$.

Variable projection algorithm (L.-M. based)

Here, we give an example of Variable projection algorithm.

Set the starting point $\mathbf{a}_0$
loop
Compute $\mathbf{c}_{k+1}$ by solving the linear LSQ problem

$$\min_{\mathbf{c}} \|\mathbf{y} - F(\mathbf{a}_k)\mathbf{c}\|_2^2.$$

Choose the Lagrange parameter λ_k;
Solve the linear LSQ problem

$$\min_{\mathbf{p}} \left\| \begin{bmatrix} J(\mathbf{a}_k) \\ \sqrt{\lambda_k} I \end{bmatrix} \mathbf{p} - \begin{bmatrix} -\mathbf{r}(\mathbf{a}_k) \\ \mathbf{0} \end{bmatrix} \right\|_2^2$$

to obtain the step $\mathbf{p}_k^{\mathrm{LM}}$.
Calculate the new iterate: $\mathbf{a}_{k+1} = \mathbf{a}_k + \mathbf{p}_k^{\mathrm{LM}}$;
Check for convergence;
end loop
Output $\mathbf{a}_{k+1}$ and $\mathbf{c}_{k+1}$.

Variable projection method

- In variable projection algorithm, we can choose any nonlinear LSQ solver if it only requires Jacobian.
- Nonlinear LSQ solver is applied on a space of smaller dimension than the original problem.
- It converges faster and more stable than using nonlinear LSQ solver directly on the original problem.
- The idea of variable projection algorithm can be generalized to any model, where some of the parameters occur linearly.

Regression for linear models

- l_2-regression (least squares)

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{y} - A\mathbf{x}\|_2^2$$

- l_1-regression

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{y} - A\mathbf{x}\|_1$$

- l_∞-regression

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{y} - A\mathbf{x}\|_\infty$$

- Huber-regression

$$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{i=1}^{m} \phi_\gamma(r_i(\mathbf{x})), \qquad \mathbf{r}(\mathbf{x}) = \mathbf{y} - A\mathbf{x}$$

where the Huber function is defined as

$$\phi_\gamma(u) = \begin{cases} \frac{1}{2\gamma}u^2, & |u| \le \gamma \\ |u| - \frac{\gamma}{2}, & |u| > \gamma \end{cases}$$

Simple example

We try to fit the data (t_i, y_i) for $i = 1, \cdots, m$ by a simple function $\phi(x, t_i) = x$. Then, the residual is $\mathbf{r} = \mathbf{y} - x$.

- l_2-**regression:** We need solve

$$\min_{x \in \mathbb{R}} \|\mathbf{y} - x\|_2^2 \iff \min_{x \in \mathbb{R}} \sum_{i=1}^{m} (y_i - x)^2$$
$$\iff \min_{x \in \mathbb{R}} mx^2 - 2 \left(\sum_{i=1}^{m} y_i \right) x + \sum_{i=1}^{m} y_i^2.$$

  According to the optimality condition, we have $x^*_{(2)} = \frac{1}{m} \sum_{i=1}^{m} y_i$.

- l_1-**regression:** We need solve

$$\min_{x \in \mathbb{R}} \|\mathbf{y} - x\|_1 \iff \min_{x \in \mathbb{R}} \sum_{i=1}^{m} |y_i - x|.$$

  The minimizer is $x^*_{(1)} = \mathrm{median}(y_1, \cdots, y_m)$.

Simple example

- l_2**-regression:** We have $x^*_{(2)} = \frac{1}{m}\sum_{i=1}^m y_i$.
- l_1**-regression:** We have $x^*_{(1)} = \mathrm{median}(y_1, \cdots, y_m)$.
- l_∞**-regression:** We need solve

$$\min_{x\in\mathbb{R}} \|\mathbf{y} - x\|_\infty \Longleftrightarrow \min_{x\in\mathbb{R}} \max\{|y_1 - x|, \cdots, |y_m - x|\}.$$

The minimizer is $x^*_{(\infty)} = \frac{1}{2}(\min\{y_i\} + \max\{y_i\})$.

Simple example

- l_2**-regression:** We have $x^*_{(2)} = \frac{1}{m}\sum_{i=1}^m y_i$.
- l_1**-regression:** We have $x^*_{(1)} = \mathrm{median}(y_1, \cdots, y_m)$.
- l_∞**-regression:** We have $x^*_{(\infty)} = \frac{1}{2}(\min\{y_i\} + \max\{y_i\})$.

These three minimizers have different response to outliers.

Let $y_K = \max\{y_i\}$ and assume that it is perturbed to $y_K + \Delta$, where $\Delta > 0$. Then, the three minimizers change to $x_{(p)} + \delta_{(p)}$ with

$$\delta_{(2)} = \frac{\Delta}{m}, \qquad \delta_{(1)} = 0, \qquad \delta_{(\infty)} = \frac{\Delta}{2}.$$

The l_1-regression is robust to the outliers.

Quadratic programs

The quadratic programming problem

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T H\mathbf{x} + \mathbf{g}^T\mathbf{x} + \gamma \\ \text{s.t.} \quad & A\mathbf{x} = \mathbf{b} \\ & C\mathbf{x} \leq \mathbf{d} \end{aligned}$$

- If H is positive semidefinite, it is a convex QP.
- If H is positive definite, it is a strictly convex QP.

Optimality condition (necessary and sufficient):

$$\nabla F(\mathbf{x}) = H\mathbf{x} + \mathbf{g} = 0 \qquad \Longleftrightarrow \qquad H\mathbf{x} = -\mathbf{g}$$

The optimum is

$$\mathbf{x} = -H^{-1}\mathbf{g}$$

Quadratic programs

The unconstrained quadratic programming problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T H\mathbf{x} + \mathbf{g}^T\mathbf{x} + \gamma$$

- If H is positive semidefinite, it is a convex QP.
- If H is positive definite, it is a strictly convex QP.

Optimality condition (necessary and sufficient):

$$\nabla F(\mathbf{x}) = H\mathbf{x} + \mathbf{g} = 0 \qquad \Longleftrightarrow \qquad H\mathbf{x} = -\mathbf{g}$$

The optimum is

$$\mathbf{x} = -H^{-1}\mathbf{g}$$

Example: Constrained least squares regression

Constrained least squares regression problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x}-\mathbf{y}\|_2^2$$
$$\text{s.t.} \quad l \le \mathbf{x} \le u$$

The objective function is quadratic

$$\begin{aligned} F(\mathbf{x}) &= \frac{1}{2}\|A\mathbf{x}-\mathbf{y}\|_2^2 = \frac{1}{2}(A\mathbf{x}-\mathbf{y})^T(A\mathbf{x}-\mathbf{y}) \\ &= \frac{1}{2}\mathbf{x}^T A^T A\mathbf{x} - \frac{1}{2}\mathbf{y}^T A^T\mathbf{x} - \frac{1}{2}\mathbf{y}^T A\mathbf{x} + \frac{1}{2}\mathbf{y}^T\mathbf{y}. \end{aligned}$$

Define $H = A^T A$, $\mathbf{g} = -\frac{1}{2}(A\mathbf{y} + A^T\mathbf{y})$ and $\gamma = \frac{1}{2}\mathbf{y}^T\mathbf{y}$, then it shows that the LSQ is a convex QP.

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T H\mathbf{x} + \mathbf{g}^T\mathbf{x} + \gamma$$
$$\text{s.t.} \quad l \le \mathbf{x} \le u$$

Example: Constrained weighted least squares regression

Constrained weighted least squares regression problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x}-\mathbf{y}\|_W^2 \quad \text{with } W^T = W$$
$$\text{s.t.} \quad l \le \mathbf{x} \le u$$

The objective function is quadratic

$$\begin{aligned} F(\mathbf{x}) &= \frac{1}{2}\|A\mathbf{x}-\mathbf{y}\|_W^2 = \frac{1}{2}(A\mathbf{x}-\mathbf{y})^T W(A\mathbf{x}-\mathbf{y}) \\ &= \frac{1}{2}\mathbf{x}^T A^T W A\mathbf{x} - \frac{1}{2}\mathbf{y}^T A^T W\mathbf{x} - \frac{1}{2}\mathbf{y}^T W A\mathbf{x} + \frac{1}{2}\mathbf{y}^T W\mathbf{y}. \end{aligned}$$

Define $H = A^T W A$, $\mathbf{g} = -\frac{1}{2}(WA\mathbf{y} + A^T W\mathbf{y})$ and $\gamma = \frac{1}{2}\mathbf{y}^T W\mathbf{y}$, then it shows that the LSQ is a convex QP.

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T H\mathbf{x} + \mathbf{g}^T\mathbf{x} + \gamma$$
$$\text{s.t.} \quad l \le \mathbf{x} \le u$$

`cvxopt.solvers.qp`

- CVXOPT is a free software package for convex optimization based on the Python programming language.
- `cvxopt.solvers` includes convex optimization routines and optional interfaces to solvers from GLPK, MOSEK, and DSDP5.
- `cvxopt.solvers.qp` attempts to solve the quadratic programming problem:

$$\min_{\mathbf{x}} \ \frac{1}{2}\mathbf{x}^T P\mathbf{x} + \mathbf{q}^T\mathbf{x} \qquad \text{subject to: } G\mathbf{x} \leq \mathbf{h} \text{ and } A\mathbf{x} = \mathbf{b}$$

- `cvxopt.solvers.qp(P, q[, G, h[, A, b[, solver[, initvals]]]])`
  - `solvers.options` can be used to set the maximum number of iterations, tolerances, etc.

Linear programs

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & F(\mathbf{x}) = \mathbf{g}^T\mathbf{x} + \gamma \\ \text{s.t.} \quad & A\mathbf{x} = \mathbf{b} \\ & C\mathbf{x} \le \mathbf{d} \\ & l \le \mathbf{x} \le u \end{aligned}$$

- The objective function is linear.
- The constraints are linear.
- It is convex as well as concave.

Example: l_1-norm regression

The l_1 regression problem is

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & F(\mathbf{x}) = \|\mathbf{r}(\mathbf{x})\|_1 = \sum_{i=1}^{m} |r_i(\mathbf{x})| \\ \text{s.t.} \quad & \mathbf{r}(\mathbf{x}) = A\mathbf{x} - \mathbf{y} \end{aligned}$$

It can be equivalently expressed as

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n, \mathbf{s}\in\mathbb{R}^m} \quad & F(\mathbf{x}, \mathbf{s}) = \sum_{i=1}^{m} s_i \\ \text{s.t.} \quad & \mathbf{r}(\mathbf{x}) = A\mathbf{x} - \mathbf{y} \\ & s_i \geq |r_i(\mathbf{x})| \qquad i = 1, 2, \cdots, m. \end{aligned}$$

Example: l_1-norm regression

- The objective function:

$$F(\mathbf{x},\mathbf{s}) = \sum_{i=1}^{m} s_i = \mathbf{e}^T\mathbf{s} = \begin{bmatrix} \mathbf{0} \\ \mathbf{e} \end{bmatrix}^T \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix}$$

- The constraints:

$$\mathbf{s} \geq |\mathbf{r}(\mathbf{x})| \quad \Longleftrightarrow \quad -\mathbf{s} \leq \mathbf{r}(\mathbf{x}) \leq \mathbf{s} \quad \Longleftrightarrow \quad -\mathbf{s} \leq A\mathbf{x} - \mathbf{y} \leq \mathbf{s}$$

It is equivalent to

$$\begin{aligned} -A\mathbf{x} - \mathbf{s} &\leq -\mathbf{y} \\ A\mathbf{x} - \mathbf{s} &\leq \mathbf{y} \end{aligned}$$

Hence

$$\begin{bmatrix} -A & -I \\ A & -I \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix} \leq \begin{bmatrix} -\mathbf{y} \\ \mathbf{y} \end{bmatrix}$$

Example: l_1-norm regression

The l_1 regression problem is

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) = \|A\mathbf{x} - \mathbf{y}\|_1$$

It can be expressed as a linear program

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\mathbf{s}\in\mathbb{R}^m} \quad & \begin{bmatrix} \mathbf{0} \\ \mathbf{e} \end{bmatrix}^T \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix} \\ \text{s.t.} \quad & \begin{bmatrix} -A & -I \\ A & -I \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix} \leq \begin{bmatrix} -\mathbf{y} \\ \mathbf{y} \end{bmatrix} \end{aligned}$$

Example: l_1-norm regression

The constrained l_1 regression problem is

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & F(\mathbf{x}) = \|A\mathbf{x} - \mathbf{y}\|_1 \\ \text{s.t.} \quad & C\mathbf{x} \le \mathbf{d} \end{aligned}$$

It can be expressed as a linear program

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n, \mathbf{s}\in\mathbb{R}^m} \quad & \begin{bmatrix} \mathbf{0} \\ \mathbf{e} \end{bmatrix}^T \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix} \\ \text{s.t.} \quad & \begin{bmatrix} -A & -I \\ A & -I \\ C & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix} \le \begin{bmatrix} -\mathbf{y} \\ \mathbf{y} \\ \mathbf{d} \end{bmatrix} \end{aligned}$$

`scipy.optimize.linprog` OR `cvxopt.solvers.lp`

- Both attempt to solve the linear programming problem:

$$\min_{\mathbf{x}} \ \mathbf{c}^T\mathbf{x} \qquad \text{subject to: } G\mathbf{x} \leq \mathbf{h} \text{ and } A\mathbf{x} = \mathbf{b}$$

- `scipy.optimize.linprog(c, A_ub=G, b_ub=h, A_eq=A, b_eq=b, bounds=(0, None), method='highs', callback=None, options=None, x0=None, integrality=None)`
  - `bounds` defines the range of $\mathbf{x}$;
  - `method` by default applies the high-performance parallel linear programming software, HiGHS;
  - `integrality` indicates the type of integrality constraints on $\mathbf{x}$.

- `cvxopt.solvers.lp(c, G, h[, A, b[, solver[, primalstart[, dualstart]]]])`
  - `solvers.options` can be used to set the maximum number of iterations, tolerances, etc.

Example: l_∞-norm regression

The l_∞ regression problem is

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & F(\mathbf{x}) = \|\mathbf{r}(\mathbf{x})\|_\infty = \max_{i\in\{1,2,\cdots,m\}} |r_i(\mathbf{x})| \\ \text{s.t.} \quad & \mathbf{r}(\mathbf{x}) = A\mathbf{x} - \mathbf{y} \end{aligned}$$

Example: l_∞-norm regression

The l_∞ regression problem is

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & F(\mathbf{x}) = \|\mathbf{r}(\mathbf{x})\|_\infty = \max_{i\in\{1,2,\cdots,m\}} |r_i(\mathbf{x})| \\ \text{s.t.} \quad & \mathbf{r}(\mathbf{x}) = A\mathbf{x} - \mathbf{y} \end{aligned}$$

It can be equivalently expressed as

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n, s\in\mathbb{R}} \quad & F(\mathbf{x}, s) = s \\ \text{s.t.} \quad & \mathbf{r}(\mathbf{x}) = A\mathbf{x} - \mathbf{y} \\ & s \geq |r_i(\mathbf{x})| \qquad i = 1, 2, \cdots, m. \end{aligned}$$

Example: l_∞-norm regression

- The objective function:

$$F(\mathbf{x}, \mathbf{s}) = s = \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}^T \begin{bmatrix} \mathbf{x} \\ s \end{bmatrix}$$

- The constraints:

$$s\mathbf{e} \geq |\mathbf{r}(\mathbf{x})| \iff -s\mathbf{e} \leq \mathbf{r}(\mathbf{x}) \leq s\mathbf{e} \iff -s\mathbf{e} \leq A\mathbf{x} - \mathbf{y} \leq s\mathbf{e}$$

It is equivalent to

$$-A\mathbf{x} - s\mathbf{e} \leq -\mathbf{y}$$
$$A\mathbf{x} - s\mathbf{e} \leq \mathbf{y}$$

e = [1, 1, …., 1]^T
(same length as r)

Hence

$$\begin{bmatrix} -A & -\mathbf{e} \\ A & -\mathbf{e} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ s \end{bmatrix} \leq \begin{bmatrix} -\mathbf{y} \\ \mathbf{y} \end{bmatrix}$$

Example: l_∞-norm regression

The l_∞ regression problem is

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad F(\mathbf{x}) = \|A\mathbf{x} - \mathbf{y}\|_\infty$$

It can be expressed as a linear program

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n, s\in\mathbb{R}} \quad & \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}^T \begin{bmatrix} \mathbf{x} \\ s \end{bmatrix} \\ \text{s.t.} \quad & \begin{bmatrix} -A & -\mathbf{e} \\ A & -\mathbf{e} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ s \end{bmatrix} \leq \begin{bmatrix} -\mathbf{y} \\ \mathbf{y} \end{bmatrix} \end{aligned}$$

Example: l_∞-norm regression

The constrained l_∞ regression problem is

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & F(\mathbf{x}) = \|A\mathbf{x} - \mathbf{y}\|_\infty \\ \text{s.t.} \quad & C\mathbf{x} \le \mathbf{d} \end{aligned}$$

It can be expressed as a linear program

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n, s\in\mathbb{R}} \quad & \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}^T \begin{bmatrix} \mathbf{x} \\ s \end{bmatrix} \\ \text{s.t.} \quad & \begin{bmatrix} -A & -\mathbf{e} \\ A & -\mathbf{e} \\ C & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ s \end{bmatrix} \le \begin{bmatrix} -\mathbf{y} \\ \mathbf{y} \\ \mathbf{d} \end{bmatrix} \end{aligned}$$

Example: Huber regression

Huber regression problem is

$$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{i=1}^{m} \phi_\gamma(r_i(\mathbf{x})), \qquad \mathbf{r}(\mathbf{x}) = A\mathbf{x} - \mathbf{y}$$

where the Huber function is defined as

$$\phi_\gamma(u) = \begin{cases} \frac{1}{2\gamma}u^2, & |u| \leq \gamma \\ |u| - \frac{\gamma}{2}, & |u| > \gamma \end{cases}$$

It can be equivalently expressed as

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\mathbf{a},\mathbf{b},\mathbf{c}\in\mathbb{R}^m} \quad & F(\mathbf{x},\mathbf{a},\mathbf{b},\mathbf{c}) = \frac{1}{2}\mathbf{c}^T\mathbf{c} + \gamma\mathbf{e}^T(\mathbf{a}+\mathbf{b}) \\ \text{s.t.} \quad & \mathbf{c} - A\mathbf{x} + \mathbf{y} - \mathbf{a} + \mathbf{b} = \mathbf{0} \\ & \mathbf{a} \geq 0 \\ & \mathbf{b} \geq 0 \end{aligned}$$

Example: Huber regression

- The objective function:

$$F(\mathbf{x},\mathbf{a},\mathbf{b},\mathbf{c}) = \frac{1}{2}\mathbf{c}^T\mathbf{c} + \gamma\mathbf{e}^T(\mathbf{a}+\mathbf{b}) = \frac{1}{2}\mathbf{z}^T H\mathbf{z} + \mathbf{g}^T\mathbf{z}$$

with

$$H = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & I \end{bmatrix}, \qquad \mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \end{bmatrix}, \qquad \mathbf{g} = \begin{bmatrix} \mathbf{0} \\ \gamma\mathbf{e} \\ \gamma\mathbf{e} \\ \mathbf{0} \end{bmatrix}$$

- The constraints:

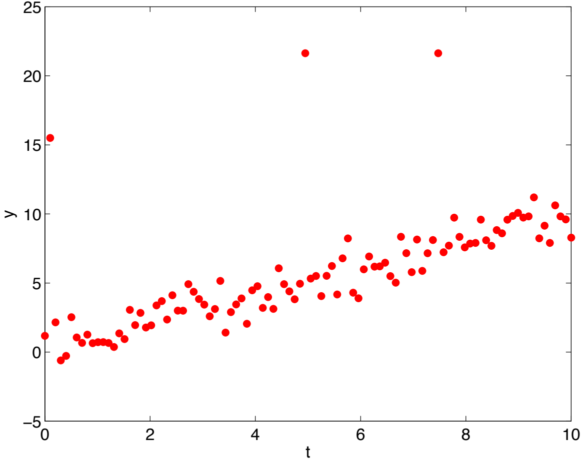
$$C\mathbf{z} = \mathbf{d} \qquad \text{and} \qquad \mathbf{l} \le \mathbf{z} \le \mathbf{u}$$

with

$$C = [-A,\ -I,\ I,\ I], \quad \mathbf{d} = -\mathbf{y}, \quad \mathbf{l} = \begin{bmatrix} -\infty \\ \mathbf{0} \\ \mathbf{0} \\ -\infty \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} +\infty \\ +\infty \\ +\infty \\ +\infty \end{bmatrix}$$

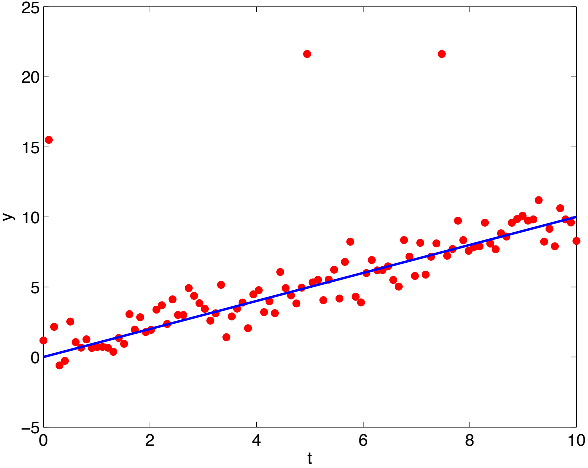
Example: Data fitting

$$y_i = \alpha t_i + \beta + e_i \qquad e_i \sim N(0, \sigma^2)$$



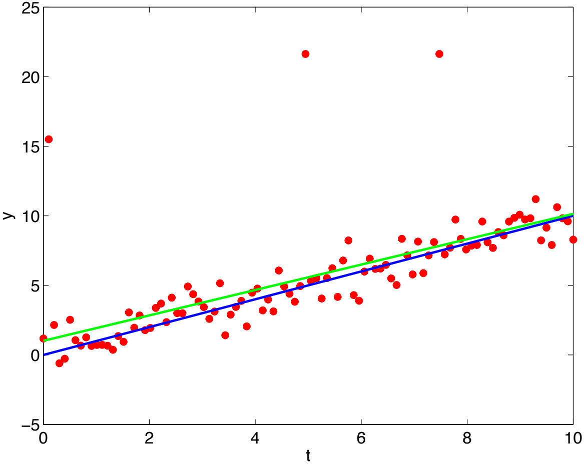
Example: True system

$$y_i = \alpha t_i + \beta + e_i \qquad e_i \sim N(0, \sigma^2)$$



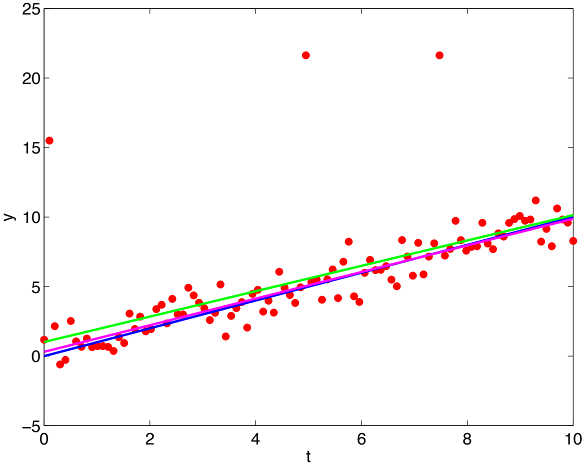
Example: Least squares fit

$$y_i = \alpha t_i + \beta + e_i \qquad e_i \sim N(0, \sigma^2)$$



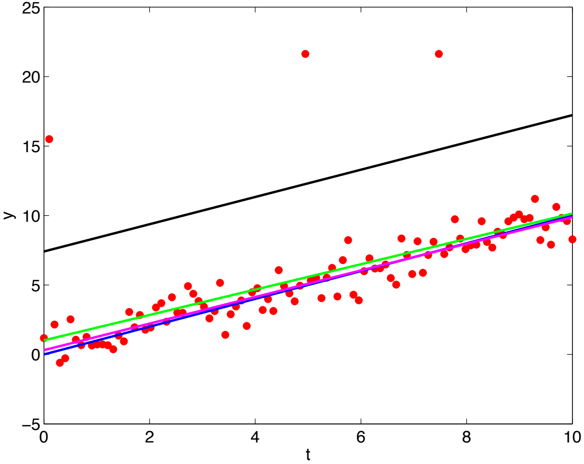
Example: l_1 fit

$$y_i = \alpha t_i + \beta + e_i \qquad e_i \sim N(0, \sigma^2)$$



Example: l_∞ fit

$$y_i = \alpha t_i + \beta + e_i \qquad e_i \sim N(0, \sigma^2)$$



Example: Huber fit ($\gamma = 3$)

$$y_i = \alpha t_i + \beta + e_i \qquad e_i \sim N(0, \sigma^2)$$