Sparsity Methods for Systems and Control

Curve Fitting and Sparse Optimization

Masaaki Nagahara[1]

[1]The University of Kitakyushu
nagahara@ieee.org

Table of Contents

Table of Contents

Minimum-ℓ^2 solution

- Linear equation: $y = \Phi x$
  - $y \in \mathbb{R}^m$ is given
  - $\Phi \in \mathbb{R}^{m \times n}$ is given
  - $x \in \mathbb{R}^n$ is unknown
- Assume $m < n$ and Φ has full row rank, i.e. $\mathrm{rank}(\Phi) = m$.

ℓ^2 optimization problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|x\|_2^2 \ \text{ subject to } \ \Phi x = y.$$

Minimum-ℓ^2 solution

- Linear equation: $\boldsymbol{y} = \Phi \boldsymbol{x}$
  - $\boldsymbol{y} \in \mathbb{R}^m$ is given
  - $\Phi \in \mathbb{R}^{m \times n}$ is given
  - $\boldsymbol{x} \in \mathbb{R}^n$ is unknown
- Assume $m < n$ and Φ has full row rank, i.e. $\operatorname{rank}(\Phi) = m$.

ℓ^2 optimization problem

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\boldsymbol{x}\|_2^2 \ \text{ subject to } \ \Phi \boldsymbol{x} = \boldsymbol{y}.$$

Minimum-ℓ^2 solution

- Linear equation: $\boldsymbol{y} = \Phi \boldsymbol{x}$
  - $\boldsymbol{y} \in \mathbb{R}^m$ is given
  - $\Phi \in \mathbb{R}^{m \times n}$ is given
  - $\boldsymbol{x} \in \mathbb{R}^n$ is unknown
- Assume $m < n$ and Φ has full row rank, i.e. $\text{rank}(\Phi) = m$.

ℓ^2 optimization problem

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\boldsymbol{x}\|_2^2 \ \text{subject to} \ \Phi\boldsymbol{x} = \boldsymbol{y}.$$

Minimum-ℓ^2 solution

- Linear equation: $\boldsymbol{y} = \Phi \boldsymbol{x}$
  - $\boldsymbol{y} \in \mathbb{R}^m$ is given
  - $\Phi \in \mathbb{R}^{m \times n}$ is given
  - $\boldsymbol{x} \in \mathbb{R}^n$ is unknown
- Assume $m < n$ and Φ has full row rank, i.e. $\operatorname{rank}(\Phi) = m$.

ℓ^2 optimization problem

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\boldsymbol{x}\|_2^2 \ \text{subject to} \ \Phi \boldsymbol{x} = \boldsymbol{y}.$$

Minimum-ℓ^2 solution

- Linear equation: $y = \Phi x$
  - $y \in \mathbb{R}^m$ is given
  - $\Phi \in \mathbb{R}^{m \times n}$ is given
  - $x \in \mathbb{R}^n$ is unknown
- Assume $m < n$ and Φ has full row rank, i.e. $\text{rank}(\Phi) = m$.

ℓ^2 optimization problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|x\|_2^2 \ \text{subject to} \ \Phi x = y.$$

Minimum-ℓ^2 solution

- Linear equation: $\boldsymbol{y} = \Phi \boldsymbol{x}$
  - $\boldsymbol{y} \in \mathbb{R}^m$ is given
  - $\Phi \in \mathbb{R}^{m \times n}$ is given
  - $\boldsymbol{x} \in \mathbb{R}^n$ is unknown
- Assume $m < n$ and Φ has full row rank, i.e. $\operatorname{rank}(\Phi) = m$.

ℓ^2 optimization problem

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\boldsymbol{x}\|_2^2 \ \text{subject to} \ \Phi \boldsymbol{x} = \boldsymbol{y}.$$

Solution of ℓ^2 optimization problem

- Lagrangian

$$L(x,\lambda) = \frac{1}{2}x^\top x + \lambda^\top(\Phi x - y).$$

- Differentiate L by x

$$\frac{\partial L}{\partial x} = \frac{\partial}{\partial x}\left(\frac{1}{2}x^\top x + \lambda^\top \Phi x\right) = x + \Phi^\top \lambda.$$

- The stationary point equation

$$x^* + \Phi^\top \lambda^* = \mathbf{0}. \quad \text{(i)}$$

- Also, x^* satisfies the equation $\Phi x = y$, we have

$$\Phi x^* = y \quad \text{(ii)}$$

Solution of ℓ^2 optimization problem

- Lagrangian

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{x} + \boldsymbol{\lambda}^\top (\Phi \boldsymbol{x} - \boldsymbol{y}).$$

- Differentiate L by $\boldsymbol{x}$

$$\frac{\partial L}{\partial \boldsymbol{x}} = \frac{\partial}{\partial \boldsymbol{x}} \left(\frac{1}{2}\boldsymbol{x}^\top \boldsymbol{x} + \boldsymbol{\lambda}^\top \Phi \boldsymbol{x} \right) = \boldsymbol{x} + \Phi^\top \boldsymbol{\lambda}.$$

- The stationary point equation

$$\boldsymbol{x}^* + \Phi^\top \boldsymbol{\lambda}^* = \mathbf{0}. \qquad \text{(i)}$$

- Also, $\boldsymbol{x}^*$ satisfies the equation $\Phi \boldsymbol{x} = \boldsymbol{y}$, we have

$$\Phi \boldsymbol{x}^* = \boldsymbol{y} \qquad \text{(ii)}$$

Solution of ℓ^2 optimization problem

- Lagrangian

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{x} + \boldsymbol{\lambda}^\top(\Phi \boldsymbol{x} - \boldsymbol{y}).$$

- Differentiate L by $\boldsymbol{x}$

$$\frac{\partial L}{\partial \boldsymbol{x}} = \frac{\partial}{\partial \boldsymbol{x}} \left(\frac{1}{2}\boldsymbol{x}^\top \boldsymbol{x} + \boldsymbol{\lambda}^\top \Phi \boldsymbol{x} \right) = \boldsymbol{x} + \Phi^\top \boldsymbol{\lambda}.$$

- The stationary point equation

$$\boldsymbol{x}^* + \Phi^\top \boldsymbol{\lambda}^* = \boldsymbol{0}. \quad \text{(i)}$$

- Also, $\boldsymbol{x}^*$ satisfies the equation $\Phi \boldsymbol{x} = \boldsymbol{y}$, we have

$$\Phi \boldsymbol{x}^* = \boldsymbol{y} \quad \text{(ii)}$$

Solution of ℓ^2 optimization problem

- Lagrangian

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{x} + \boldsymbol{\lambda}^\top (\Phi \boldsymbol{x} - \boldsymbol{y}).$$

- Differentiate L by $\boldsymbol{x}$

$$\frac{\partial L}{\partial \boldsymbol{x}} = \frac{\partial}{\partial \boldsymbol{x}} \left(\frac{1}{2}\boldsymbol{x}^\top \boldsymbol{x} + \boldsymbol{\lambda}^\top \Phi \boldsymbol{x} \right) = \boldsymbol{x} + \Phi^\top \boldsymbol{\lambda}.$$

- The stationary point equation

$$\boldsymbol{x}^* + \Phi^\top \boldsymbol{\lambda}^* = \boldsymbol{0}. \qquad \text{(i)}$$

- Also, $\boldsymbol{x}^*$ satisfies the equation $\Phi \boldsymbol{x} = \boldsymbol{y}$, we have

$$\Phi \boldsymbol{x}^* = \boldsymbol{y} \qquad \text{(ii)}$$

Solution of ℓ^2 optimization problem

- Inserting (i) into (ii) gives

$$-\Phi\Phi^\top \boldsymbol{\lambda}^* = \boldsymbol{y}$$

- Since $\mathrm{rank}(\Phi) = m$, $\Phi\Phi^\top$ is invertible and

$$\boldsymbol{\lambda}^* = -(\Phi\Phi^\top)^{-1}\boldsymbol{y}.$$

- Finally, we obtain the solution from (i) as

$$\boldsymbol{x}^* = \Phi^\top(\Phi\Phi^\top)^{-1}\boldsymbol{y}.$$

Solution of ℓ^2 optimization problem

- Inserting (i) into (ii) gives

$$-\Phi\Phi^\top \boldsymbol{\lambda}^* = \boldsymbol{y}$$

- Since $\text{rank}(\Phi) = m$, $\Phi\Phi^\top$ is invertible and

$$\boldsymbol{\lambda}^* = -(\Phi\Phi^\top)^{-1}\boldsymbol{y}.$$

- Finally, we obtain the solution from (i) as

$$\boldsymbol{x}^* = \Phi^\top(\Phi\Phi^\top)^{-1}\boldsymbol{y}.$$

Solution of ℓ^2 optimization problem

- Inserting (i) into (ii) gives

$$-\Phi\Phi^\top \boldsymbol{\lambda}^* = \boldsymbol{y}$$

- Since $\mathrm{rank}(\Phi) = m$, $\Phi\Phi^\top$ is invertible and

$$\boldsymbol{\lambda}^* = -(\Phi\Phi^\top)^{-1}\boldsymbol{y}.$$

- Finally, we obtain the solution from (i) as

$$\boldsymbol{x}^* = \Phi^\top(\Phi\Phi^\top)^{-1}\boldsymbol{y}.$$

Polynomial curve fitting

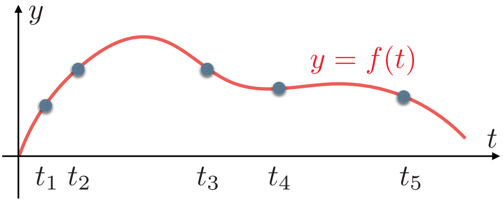
- Two-dimensional data:

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \ldots, (t_m, y_m)\}.$$

- Polynomial curve fitting

$$y = f(t) = a_{n-1}t^{n-1} + a_{n-2}t^{n-2} + \cdots + a_1 t + a_0.$$

to find coefficients $a_0, a_1, \ldots, a_{n-1}$ with which the polynomial curve has the best fit to the m-point data.



Polynomial curve fitting

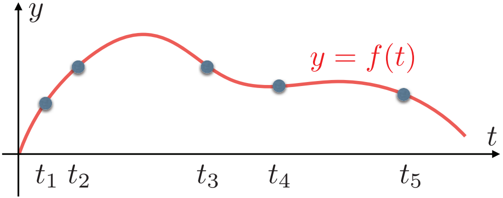
- Two-dimensional data:

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \dots, (t_m, y_m)\}.$$

- Polynomial curve fitting

$$y = f(t) = a_{n-1}t^{n-1} + a_{n-2}t^{n-2} + \cdots + a_1 t + a_0.$$

to find coefficients $a_0, a_1, \dots, a_{n-1}$ with which the polynomial curve has the best fit to the m-point data.



Interpolating polynomial

- The polynomial curve

$$y = f(t) = a_{n-1}t^{n-1} + a_{n-2}t^{n-2} + \cdots + a_1 t + a_0.$$

goes through the data points.

- linear equations with for unknown coefficients

$$\begin{aligned} a_{n-1}t_1^{n-1} + a_{n-2}t_1^{n-2} + \cdots + a_1 t_1 + a_0 &= y_1, \\ a_{n-1}t_2^{n-1} + a_{n-2}t_2^{n-2} + \cdots + a_1 t_2 + a_0 &= y_2, \\ &\cdots \\ a_{n-1}t_m^{n-1} + a_{n-2}t_m^{n-2} + \cdots + a_1 t_m + a_0 &= y_m. \end{aligned}$$

Interpolating polynomial

- The polynomial curve

$$y = f(t) = a_{n-1}t^{n-1} + a_{n-2}t^{n-2} + \cdots + a_1 t + a_0.$$

goes through the data points.

- linear equations with for unknown coefficients

$$\begin{aligned}
a_{n-1}t_1^{n-1} + a_{n-2}t_1^{n-2} + \cdots + a_1 t_1 + a_0 &= y_1, \\
a_{n-1}t_2^{n-1} + a_{n-2}t_2^{n-2} + \cdots + a_1 t_2 + a_0 &= y_2, \\
&\cdots \\
a_{n-1}t_m^{n-1} + a_{n-2}t_m^{n-2} + \cdots + a_1 t_m + a_0 &= y_m.
\end{aligned}$$

Vandermonde's matrix

- Define

$$\Phi \triangleq \begin{bmatrix} t_1^{n-1} & t_1^{n-2} & \dots & t_1 & 1 \\ t_2^{n-1} & t_2^{n-2} & \dots & t_2 & 1 \\ \vdots & \vdots & \ddots & \vdots & \\ t_m^{n-1} & t_m^{n-2} & \dots & t_m & 1 \end{bmatrix} \in \mathbb{R}^{m\times n},$$

$$\boldsymbol{x} \triangleq \begin{bmatrix} a_{n-1} \\ a_{n-2} \\ \vdots \\ a_1 \\ a_0 \end{bmatrix} \in \mathbb{R}^n, \quad \boldsymbol{y} \triangleq \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \in \mathbb{R}^m.$$

- Then the linear equations becomes $\Phi \boldsymbol{x} = \boldsymbol{y}$.
- Φ is called Vandermonde's matrix.

Vandermonde's matrix

- Define

$$\Phi \triangleq \begin{bmatrix} t_1^{n-1} & t_1^{n-2} & \dots & t_1 & 1 \\ t_2^{n-1} & t_2^{n-2} & \dots & t_2 & 1 \\ \vdots & \vdots & \ddots & \vdots & \\ t_m^{n-1} & t_m^{n-2} & \dots & t_m & 1 \end{bmatrix} \in \mathbb{R}^{m\times n},$$

$$\boldsymbol{x} \triangleq \begin{bmatrix} a_{n-1} \\ a_{n-2} \\ \vdots \\ a_1 \\ a_0 \end{bmatrix} \in \mathbb{R}^n, \quad \boldsymbol{y} \triangleq \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \in \mathbb{R}^m.$$

- Then the linear equations becomes $\Phi\boldsymbol{x} = \boldsymbol{y}$.
- Φ is called Vandermonde's matrix.

Vandermonde's matrix

- Define

$$\Phi \triangleq \begin{bmatrix} t_1^{n-1} & t_1^{n-2} & \dots & t_1 & 1 \\ t_2^{n-1} & t_2^{n-2} & \dots & t_2 & 1 \\ \vdots & \vdots & \ddots & \vdots & \\ t_m^{n-1} & t_m^{n-2} & \dots & t_m & 1 \end{bmatrix} \in \mathbb{R}^{m \times n},$$

$$\boldsymbol{x} \triangleq \begin{bmatrix} a_{n-1} \\ a_{n-2} \\ \vdots \\ a_1 \\ a_0 \end{bmatrix} \in \mathbb{R}^n, \quad \boldsymbol{y} \triangleq \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \in \mathbb{R}^m.$$

- Then the linear equations becomes $\Phi \boldsymbol{x} = \boldsymbol{y}$.
- Φ is called Vandermonde's matrix.

Vandermonde's matrix

- If $m = n$, then the determinant of Φ is given by

$$\det(\Phi) = \prod_{1 \leq i < j \leq m} (t_i - t_j) = (t_1 - t_2)(t_1 - t_3) \cdots (t_{m-1} - t_m).$$

- If $t_i \neq t_j$ for all i, j such that $i \neq j$, then Φ is invertible and the coefficients of the interpolating polynomial are obtained by

$$x^* = \Phi^{-1} y.$$

Vandermonde's matrix

- If $m = n$, then the determinant of Φ is given by

$$\det(\Phi) = \prod_{1 \leq i < j \leq m} (t_i - t_j) = (t_1 - t_2)(t_1 - t_3) \cdots (t_{m-1} - t_m).$$

- If $t_i \neq t_j$ for all i, j such that $i \neq j$, then Φ is invertible and the coefficients of the interpolating polynomial are obtained by

$$\boldsymbol{x}^* = \Phi^{-1} \boldsymbol{y}.$$

MATLAB Simulation

- Data:

t	1	2	3	...	14	15
y	2	4	6	...	28	30

$\rightarrow y = 2t$

- Result:

```
x =

   2.274746684520826e-24
  -5.565271161256770e-21
   9.137367918505765e-19
  -9.452887691357992e-18
  -3.658098129966092e-16
  -1.608088662230500e-15
   3.569367024169878e-14
  -6.021849685566849e-13
   5.346834086594754e-13
  -1.267963511963899e-11
   4.878586423728848e-11
   2.088995643134695e-12
   1.366515789413825e-10
   1.999999999995282e+00 <--
  -4.014566457044566e-12
```

MATLAB Simulation

- Data:

t	1	2	3	...	14	15
y	2	4	6	...	28	30

$\rightarrow y = 2t$

- Result:

```
x =

   2.274746684520826e-24
  -5.565271161256770e-21
   9.137367918505765e-19
  -9.452887691357992e-18
  -3.658098129966092e-16
  -1.608088662230500e-15
   3.569367024169878e-14
  -6.021849685566849e-13
   5.346834086594754e-13
  -1.267963511963899e-11
   4.878586423728848e-11
   2.088995643134695e-12
   1.366515789413825e-10
   1.999999999995282e+00 <--
  -4.014566457044566e-12
```

MATLAB Simulation

- Data:

t	1	2	3	...	14	15
y	2	4	6	...	28	30

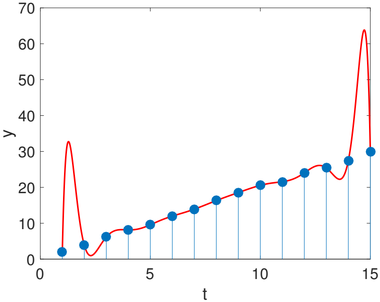
$\rightarrow y = 2t$

- Result:

```
x =
    2.274746684520826e-24
   -5.565271161256770e-21
    9.137367918505765e-19
   -9.452887691357992e-18
   -3.658098129966092e-16
   -1.608088662230500e-15
    3.569367024169878e-14
   -6.021849685566849e-13
    5.346834086594754e-13
   -1.267963511963899e-11
    4.878586423728848e-11
    2.088995643134695e-12
    1.366515789413825e-10
    1.999999999995282e+00 <--
   -4.014566457044566e-12
```

MATLAB Simulation

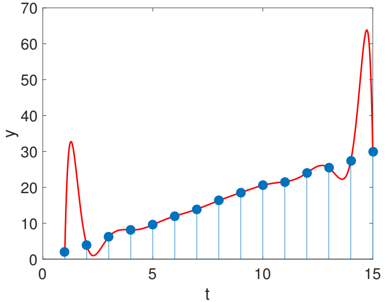
- Add Gaussian noise with mean 0 and variance 0.5^2 to the data y.
- Curve fitting via Vandermonde's matrix inversion $\Phi^{-1} y$.



- This is known as overfitting.

MATLAB Simulation

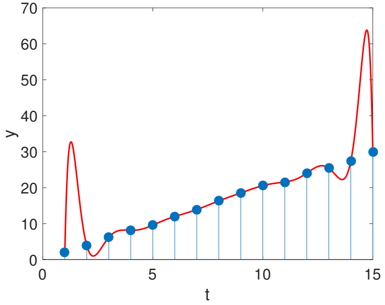
- Add Gaussian noise with mean 0 and variance 0.5^2 to the data y.
- Curve fitting via Vandermonde's matrix inversion $\Phi^{-1}\boldsymbol{y}$.



- This is known as overfitting.

MATLAB Simulation

- Add Gaussian noise with mean 0 and variance 0.5^2 to the data y.
- Curve fitting via Vandermonde's matrix inversion $\Phi^{-1}\boldsymbol{y}$.



- This is known as overfitting.

Least squares method

- The order of the polynomial was too large.
- We can first assume a first-order polynomial $y = a_1 t + a_0$.
- The line does not interpolate the noisy data.
- Find a polynomial that minimizes the ℓ^2 error:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi x - y\|_2^2,$$

- This is called the least squares method.

Least squares method

- The order of the polynomial was too large.
- We can first assume a first-order polynomial $y = a_1 t + a_0$.
- The line does not interpolate the noisy data.
- Find a polynomial that minimizes the ℓ^2 error:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi x - y\|_2^2,$$

- This is called the least squares method.

Least squares method

- The order of the polynomial was too large.
- We can first assume a first-order polynomial $y = a_1 t + a_0$.
- The line does not interpolate the noisy data.
- Find a polynomial that minimizes the ℓ^2 error:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi x - y\|_2^2,$$

- This is called the least squares method.

Least squares method

- The order of the polynomial was too large.
- We can first assume a first-order polynomial $y = a_1 t + a_0$.
- The line does not interpolate the noisy data.
- Find a polynomial that minimizes the ℓ^2 error:

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi\boldsymbol{x} - \boldsymbol{y}\|_2^2,$$

- This is called the least squares method.

Least squares method

- The order of the polynomial was too large.
- We can first assume a first-order polynomial $y = a_1 t + a_0$.
- The line does not interpolate the noisy data.
- Find a polynomial that minimizes the ℓ^2 error:

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi\boldsymbol{x} - \boldsymbol{y}\|_2^2,$$

- This is called the least squares method.

The least squares solution

- The error function

$$E(x) = \frac{1}{2}\|\Phi x - y\|_2^2 = \frac{1}{2}(\Phi x - y)^\top(\Phi x - y)$$
$$= \frac{1}{2}x^\top\Phi^\top\Phi x - y^\top\Phi x + \frac{1}{2}y^\top y.$$

- Φ is $m \times n$ with $m > n$ (a tall matrix).
- If $t_i \neq t_j$ for all i, j such that $i \neq j$, then Φ has full column rank, and $\Phi^\top\Phi > 0$ (positive definite).
- The minimizer x^* satisfies

$$\frac{\partial E}{\partial x}(x^*) = (\Phi^\top\Phi)x^* - \Phi^\top y = \mathbf{0}.$$

- The solution

$$x^* = (\Phi^\top\Phi)^{-1}\Phi^\top y.$$

The least squares solution

- The error function

$$E(\boldsymbol{x}) = \frac{1}{2}\|\Phi\boldsymbol{x} - \boldsymbol{y}\|_2^2 = \frac{1}{2}(\Phi\boldsymbol{x} - \boldsymbol{y})^\top(\Phi\boldsymbol{x} - \boldsymbol{y})$$
$$= \frac{1}{2}\boldsymbol{x}^\top\Phi^\top\Phi\boldsymbol{x} - \boldsymbol{y}^\top\Phi\boldsymbol{x} + \frac{1}{2}\boldsymbol{y}^\top\boldsymbol{y}.$$

- Φ is $m \times n$ with $m > n$ (a tall matrix).
- If $t_i \neq t_j$ for all i, j such that $i \neq j$, then Φ has full column rank, and $\Phi^\top\Phi > 0$ (positive definite).
- The minimizer $\boldsymbol{x}^*$ satisfies

$$\frac{\partial E}{\partial \boldsymbol{x}}(\boldsymbol{x}^*) = (\Phi^\top\Phi)\boldsymbol{x}^* - \Phi^\top\boldsymbol{y} = \boldsymbol{0}.$$

- The solution

$$\boldsymbol{x}^* = (\Phi^\top\Phi)^{-1}\Phi^\top\boldsymbol{y}.$$

The least squares solution

- The error function

$$E(\boldsymbol{x}) = \frac{1}{2}\|\Phi\boldsymbol{x} - \boldsymbol{y}\|_2^2 = \frac{1}{2}(\Phi\boldsymbol{x} - \boldsymbol{y})^\top(\Phi\boldsymbol{x} - \boldsymbol{y})$$
$$= \frac{1}{2}\boldsymbol{x}^\top\Phi^\top\Phi\boldsymbol{x} - \boldsymbol{y}^\top\Phi\boldsymbol{x} + \frac{1}{2}\boldsymbol{y}^\top\boldsymbol{y}.$$

- Φ is $m \times n$ with $m > n$ (a tall matrix).
- If $t_i \neq t_j$ for all i, j such that $i \neq j$, then Φ has full column rank, and $\Phi^\top\Phi > 0$ (positive definite).
- The minimizer $\boldsymbol{x}^*$ satisfies

$$\frac{\partial E}{\partial \boldsymbol{x}}(\boldsymbol{x}^*) = (\Phi^\top\Phi)\boldsymbol{x}^* - \Phi^\top\boldsymbol{y} = \boldsymbol{0}.$$

- The solution

$$\boldsymbol{x}^* = (\Phi^\top\Phi)^{-1}\Phi^\top\boldsymbol{y}.$$

The least squares solution

- The error function

$$E(\boldsymbol{x}) = \frac{1}{2}\|\Phi\boldsymbol{x} - \boldsymbol{y}\|_2^2 = \frac{1}{2}(\Phi\boldsymbol{x} - \boldsymbol{y})^\top(\Phi\boldsymbol{x} - \boldsymbol{y})$$
$$= \frac{1}{2}\boldsymbol{x}^\top\Phi^\top\Phi\boldsymbol{x} - \boldsymbol{y}^\top\Phi\boldsymbol{x} + \frac{1}{2}\boldsymbol{y}^\top\boldsymbol{y}.$$

- Φ is $m \times n$ with $m > n$ (a tall matrix).
- If $t_i \neq t_j$ for all i, j such that $i \neq j$, then Φ has full column rank, and $\Phi^\top\Phi > 0$ (positive definite).
- The minimizer $\boldsymbol{x}^*$ satisfies

$$\frac{\partial E}{\partial \boldsymbol{x}}(\boldsymbol{x}^*) = (\Phi^\top\Phi)\boldsymbol{x}^* - \Phi^\top\boldsymbol{y} = \boldsymbol{0}.$$

- The solution

$$\boldsymbol{x}^* = (\Phi^\top\Phi)^{-1}\Phi^\top\boldsymbol{y}.$$

The least squares solution

- The error function

$$E(\boldsymbol{x}) = \frac{1}{2}\|\Phi\boldsymbol{x} - \boldsymbol{y}\|_2^2 = \frac{1}{2}(\Phi\boldsymbol{x} - \boldsymbol{y})^\top(\Phi\boldsymbol{x} - \boldsymbol{y})$$
$$= \frac{1}{2}\boldsymbol{x}^\top\Phi^\top\Phi\boldsymbol{x} - \boldsymbol{y}^\top\Phi\boldsymbol{x} + \frac{1}{2}\boldsymbol{y}^\top\boldsymbol{y}.$$

- Φ is $m \times n$ with $m > n$ (a tall matrix).
- If $t_i \neq t_j$ for all i, j such that $i \neq j$, then Φ has full column rank, and $\Phi^\top\Phi > 0$ (positive definite).
- The minimizer $\boldsymbol{x}^*$ satisfies

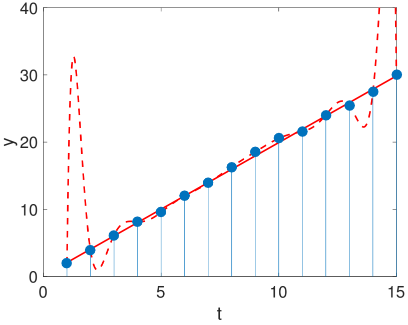
$$\frac{\partial E}{\partial \boldsymbol{x}}(\boldsymbol{x}^*) = (\Phi^\top\Phi)\boldsymbol{x}^* - \Phi^\top\boldsymbol{y} = \boldsymbol{0}.$$

- The solution

$$\boldsymbol{x}^* = (\Phi^\top\Phi)^{-1}\Phi^\top\boldsymbol{y}.$$

MATLAB Simulation

- Assume the curve is a first-order polynomial.
- The data is noisy.



Another example

- How can we know a proper order of the polynomial?
- It is often difficult.
- Data set

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \dots, (t_m, y_m)\},$$

where $y_i = \sin(t_i) + \epsilon_i$, with $t_i = i - 1$, $i = 1, 2, \dots, 11$ and ϵ_i is Gaussian noise.

- The data:

t_i	0	1	2	3	4	5
y_i	-0.0343	1.0081	0.8326	0.4047	-0.7585	-0.9285
t_i	6	7	8	9	10	
y_i	-0.2110	0.6626	0.8492	0.2761	-0.6962	

Another example

- How can we know a proper order of the polynomial?
- It is often difficult.
- Data set

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \ldots, (t_m, y_m)\},$$

  where $y_i = \sin(t_i) + \epsilon_i$, with $t_i = i - 1$, $i = 1, 2, \ldots, 11$ and ϵ_i is Gaussian noise.
- The data:

t_i	0	1	2	3	4	5
y_i	-0.0343	1.0081	0.8326	0.4047	-0.7585	-0.9285
t_i	6	7	8	9	10	
y_i	-0.2110	0.6626	0.8492	0.2761	-0.6962	

Another example

- How can we know a proper order of the polynomial?
- It is often difficult.
- Data set

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \dots, (t_m, y_m)\},$$

where $y_i = \sin(t_i) + \epsilon_i$, with $t_i = i - 1$, $i = 1, 2, \dots, 11$ and ϵ_i is Gaussian noise.

- The data:

t_i	0	1	2	3	4	5
y_i	-0.0343	1.0081	0.8326	0.4047	-0.7585	-0.9285
t_i	6	7	8	9	10	
y_i	-0.2110	0.6626	0.8492	0.2761	-0.6962	

Another example

- How can we know a proper order of the polynomial?
- It is often difficult.
- Data set

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \ldots, (t_m, y_m)\},$$

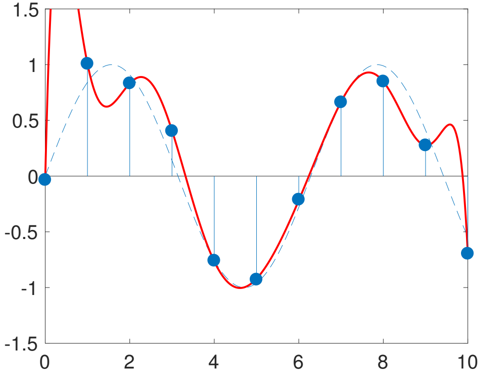
where $y_i = \sin(t_i) + \epsilon_i$, with $t_i = i - 1$, $i = 1, 2, \ldots, 11$ and ϵ_i is Gaussian noise.

- The data:

t_i	0	1	2	3	4	5
y_i	-0.0343	1.0081	0.8326	0.4047	-0.7585	-0.9285
t_i	6	7	8	9	10	
y_i	-0.2110	0.6626	0.8492	0.2761	-0.6962	

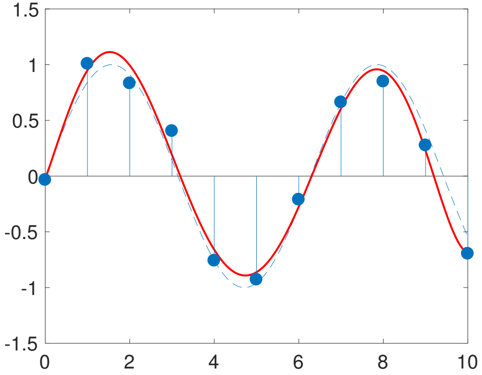
Another example

- 10th order interpolating polynomial



Another example

- 6th order polynomial by the least squares method



Another example

- What is the difference?
- The coefficients:

$$x_{10} = \begin{bmatrix} -0.0343 \\ 16.2400 \\ -38.0984 \\ 37.8369 \\ -20.2842 \\ 6.5035 \\ -1.3100 \\ 0.1677 \\ -0.0133 \\ 0.0006 \\ -0.0000 \end{bmatrix}, \quad x_6 = \begin{bmatrix} -0.0260 \\ 1.0636 \\ 0.3067 \\ -0.5225 \\ 0.1426 \\ -0.0146 \\ 0.0005 \end{bmatrix},$$

- x_{10} is “bigger” than x_6.

Another example

- What is the difference?
- The coefficients:

$$x_{10} = \begin{bmatrix} -0.0343 \\ 16.2400 \\ -38.0984 \\ 37.8369 \\ -20.2842 \\ 6.5035 \\ -1.3100 \\ 0.1677 \\ -0.0133 \\ 0.0006 \\ -0.0000 \end{bmatrix}, \quad x_6 = \begin{bmatrix} -0.0260 \\ 1.0636 \\ 0.3067 \\ -0.5225 \\ 0.1426 \\ -0.0146 \\ 0.0005 \end{bmatrix},$$

- x_{10} is "bigger" than x_6.

Another example

- What is the difference?
- The coefficients:

$$x_{10} = \begin{bmatrix} -0.0343 \\ 16.2400 \\ -38.0984 \\ 37.8369 \\ -20.2842 \\ 6.5035 \\ -1.3100 \\ 0.1677 \\ -0.0133 \\ 0.0006 \\ -0.0000 \end{bmatrix}, \quad x_6 = \begin{bmatrix} -0.0260 \\ 1.0636 \\ 0.3067 \\ -0.5225 \\ 0.1426 \\ -0.0146 \\ 0.0005 \end{bmatrix},$$

- x_{10} is “bigger” than x_6.

Regularized least squares

- Idea: keep the polynomial order high and reduce the norm of the coefficient vector
- That is,

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; \frac{1}{2}\|\Phi x - y\|_2^2 + \frac{\lambda}{2}\|x\|_2^2.$$

  with $\lambda > 0$.
- This is called the regularized least squares.
- The solution is obtained by

$$x^* = (\lambda I + \Phi^\top \Phi)^{-1} \Phi^\top y.$$

Regularized least squares

- Idea: keep the polynomial order high and reduce the norm of the coefficient vector
- That is,

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi x - y\|_2^2 + \frac{\lambda}{2}\|x\|_2^2.$$

  with $\lambda > 0$.
- This is called the regularized least squares.
- The solution is obtained by

$$x^* = (\lambda I + \Phi^\top \Phi)^{-1}\Phi^\top y.$$

Regularized least squares

- Idea: keep the polynomial order high and reduce the norm of the coefficient vector
- That is,

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi x - y\|_2^2 + \frac{\lambda}{2}\|x\|_2^2.$$

  with $\lambda > 0$.
- This is called the regularized least squares.
- The solution is obtained by

$$x^* = (\lambda I + \Phi^\top \Phi)^{-1}\Phi^\top y.$$

Regularized least squares

- Idea: keep the polynomial order high and reduce the norm of the coefficient vector
- That is,

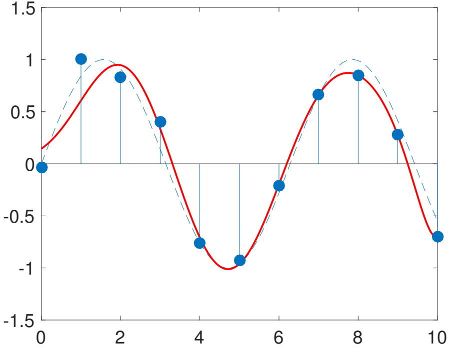
$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi\boldsymbol{x}-\boldsymbol{y}\|_2^2+\frac{\lambda}{2}\|\boldsymbol{x}\|_2^2.$$

  with $\lambda > 0$.
- This is called the regularized least squares.
- The solution is obtained by

$$\boldsymbol{x}^* = (\lambda I + \Phi^\top\Phi)^{-1}\Phi^\top\boldsymbol{y}.$$

Regularized least squares

- 10th order polynomial by the regularized least squares.



Polynomial curve fitting: summary

- data

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \ldots, (t_m, y_m)\}.$$

- polynomial

$$y = f(t) = a_{n-1}t^{n-1} + a_{n-2}t^{n-2} + \cdots + a_1 t + a_0,$$

Problem	Matrix size	Opt. problem	Solution
min ℓ^2 norm	$m < n$	$\min_{x} \frac{1}{2}\|x\|_2^2$ s.t. $y = \Phi x$	$\Phi^\top(\Phi\Phi^\top)^{-1}y$
least squares (LS)	$m > n$	$\min_{x} \frac{1}{2}\|y - \Phi x\|_2^2$	$(\Phi^\top\Phi)^{-1}\Phi^\top y$
regularized LS	any m and n	$\min_{x} \frac{1}{2}\|y - \Phi x\|_2^2 + \frac{\lambda}{2}\|x\|_2^2$	$\Phi^\top(\lambda I + \Phi\Phi^\top)^{-1}y$ $= (\lambda I + \Phi^\top\Phi)^{-1}\Phi^\top y$

Polynomial curve fitting: summary

- data

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \dots, (t_m, y_m)\}.$$

- polynomial

$$y = f(t) = a_{n-1}t^{n-1} + a_{n-2}t^{n-2} + \cdots + a_1 t + a_0,$$

Problem	Matrix size	Opt. problem	Solution
min ℓ^2 norm	$m < n$	$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{x}\|_2^2$ s.t. $\boldsymbol{y} = \Phi\boldsymbol{x}$	$\Phi^\top(\Phi\Phi^\top)^{-1}\boldsymbol{y}$
least squares (LS)	$m > n$	$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - \Phi\boldsymbol{x}\|_2^2$	$(\Phi^\top\Phi)^{-1}\Phi^\top\boldsymbol{y}$
regularized LS	any m and n	$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - \Phi\boldsymbol{x}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{x}\|_2^2$	$\Phi^\top(\lambda I + \Phi\Phi^\top)^{-1}\boldsymbol{y}$
			$= (\lambda I + \Phi^\top\Phi)^{-1}\Phi^\top\boldsymbol{y}$

Table of Contents

An example

- 80th-order polynomial

$$y = -t^{80} + t.$$

- Generate data from this polynomial

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \dots, (t_{11}, y_{11})\}, \quad y_i = -t_i^{80} + t_i.$$

  on

$$t_1 = 0, t_2 = 0.1, t_3 = 0.2, \dots, t_{11} = 1.$$

- Can we reconstruct the 80th-order polynomial ($n = 80$) from these 11 points ($m = 11$)?
- Idea: the polynomial is sparse (i.e. almost all coefficients are zero).

An example

- 80th-order polynomial

$$y = -t^{80} + t.$$

- Generate data from this polynomial

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \dots, (t_{11}, y_{11})\}, \quad y_i = -t_i^{80} + t_i.$$

on

$$t_1 = 0, t_2 = 0.1, t_3 = 0.2, \dots, t_{11} = 1.$$

- Can we reconstruct the 80th-order polynomial ($n = 80$) from these 11 points ($m = 11$)?
- Idea: the polynomial is sparse (i.e. almost all coefficients are zero).

An example

- 80th-order polynomial

$$y = -t^{80} + t.$$

- Generate data from this polynomial

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \dots, (t_{11}, y_{11})\}, \quad y_i = -t_i^{80} + t_i.$$

on

$$t_1 = 0, t_2 = 0.1, t_3 = 0.2, \dots, t_{11} = 1.$$

- Can we reconstruct the 80th-order polynomial ($n = 80$) from these 11 points ($m = 11$)?
- Idea: the polynomial is sparse (i.e. almost all coefficients are zero).

An example

- 80th-order polynomial

$$y = -t^{80} + t.$$

- Generate data from this polynomial

$$\mathcal{D} = \{(t_1, y_1), (t_2, y_2), \ldots, (t_{11}, y_{11})\}, \quad y_i = -t_i^{80} + t_i.$$

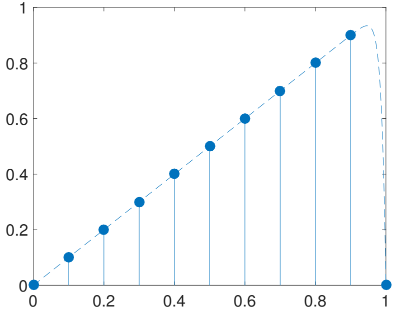
  on

$$t_1 = 0, t_2 = 0.1, t_3 = 0.2, \ldots, t_{11} = 1.$$

- Can we reconstruct the 80th-order polynomial ($n = 80$) from these 11 points ($m = 11$)?
- Idea: the polynomial is sparse (i.e. almost all coefficients are zero).

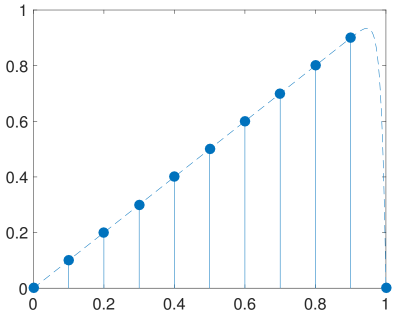
An example

- We assume we know the order is at most 80.
- There are infinitely many interpolating polynomials
  - Vandermonde's matrix Φ is a 11×81 fat matrix.
- We also assume we know the original polynomial is sparse.



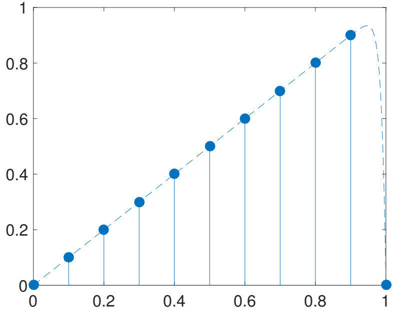
An example

- We assume we know the order is at most 80.
- There are infinitely many interpolating polynomials
  - Vandermonde's matrix Φ is a 11×81 fat matrix.
- We also assume we know the original polynomial is sparse.



An example

- We assume we know the order is at most 80.
- There are infinitely many interpolating polynomials
  - Vandermonde's matrix Φ is a 11×81 fat matrix.
- We also assume we know the original polynomial is sparse.



Sparse polynomial interpolation

- Consider the ℓ^0 optimization problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; \|x\|_0 \quad \text{subject to} \quad \Phi x = y.$$

    - This is difficult to solve.

- The idea: adopt convex relaxation by using the ℓ^1 norm

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; \|x\|_1 \quad \text{subject to} \quad \Phi x = y.$$

    - This is a convex optimization called the basis pursuit.
    - The solution can be obtained much faster than the exhaustive search for the ℓ^0 optimization.

Sparse polynomial interpolation

- Consider the ℓ^0 optimization problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \|x\|_0 \quad \text{subject to} \quad \Phi x = y.$$

  - This is difficult to solve.

- The idea: adopt convex relaxation by using the ℓ^1 norm

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \|x\|_1 \quad \text{subject to} \quad \Phi x = y.$$

  - This is a convex optimization called the basis pursuit.
  - The solution can be obtained much faster than the exhaustive search for the ℓ^0 optimization.

Sparse polynomial interpolation

- Consider the ℓ^0 optimization problem

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \ \|\boldsymbol{x}\|_0 \quad \text{subject to} \quad \Phi\boldsymbol{x} = \boldsymbol{y}.$$

  - This is difficult to solve.
- The idea: adopt convex relaxation by using the ℓ^1 norm

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \ \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad \Phi\boldsymbol{x} = \boldsymbol{y}.$$

  - This is a convex optimization called the basis pursuit.
  - The solution can be obtained much faster than the exhaustive search for the ℓ^0 optimization.

Sparse polynomial interpolation

- Consider the ℓ^0 optimization problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; \|x\|_0 \quad \text{subject to} \quad \Phi x = y.$$

  - This is difficult to solve.
- The idea: adopt convex relaxation by using the ℓ^1 norm

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; \|x\|_1 \quad \text{subject to} \quad \Phi x = y.$$

  - This is a convex optimization called the basis pursuit.
  - The solution can be obtained much faster than the exhaustive search for the ℓ^0 optimization.

Sparse polynomial interpolation

- Consider the ℓ^0 optimization problem

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \;\; \|\boldsymbol{x}\|_0 \;\; \text{subject to} \;\; \Phi\boldsymbol{x} = \boldsymbol{y}.$$

    - This is difficult to solve.
- The idea: adopt convex relaxation by using the ℓ^1 norm

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \;\; \|\boldsymbol{x}\|_1 \;\; \text{subject to} \;\; \Phi\boldsymbol{x} = \boldsymbol{y}.$$

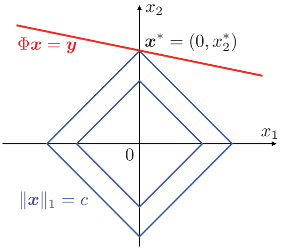
    - This is a convex optimization called the basis pursuit.
    - The solution can be obtained much faster than the exhaustive search for the ℓ^0 optimization.

ℓ^1 optimization and sparsity

- Consider ℓ^1 optimization in $\mathbb{R}^2$

$$\underset{\boldsymbol{x}\in\mathbb{R}^2}{\text{minimize}} \ \|\boldsymbol{x}\|_1 \quad \text{subject to} \ \ \Phi\boldsymbol{x} = \boldsymbol{y}.$$

- The contour $\{\boldsymbol{x} : \|\boldsymbol{x}\|_1 = c\}$ touches the linear subspace $\{\boldsymbol{x} : \Phi\boldsymbol{x} = \boldsymbol{y}\}$ on one of the corners that are on axes.



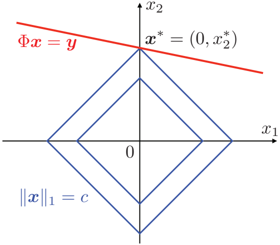
- The optimal solution has one 0 element $\rightarrow$ sparse.
- From this example, one can intuitively say that ℓ^1 optimization can promote sparsity.

ℓ^1 optimization and sparsity

- Consider ℓ^1 optimization in $\mathbb{R}^2$

$$\underset{\boldsymbol{x}\in\mathbb{R}^2}{\text{minimize}} \ \|\boldsymbol{x}\|_1 \quad \text{subject to} \ \ \Phi\boldsymbol{x} = \boldsymbol{y}.$$

- The contour $\{\boldsymbol{x} : \|\boldsymbol{x}\|_1 = c\}$ touches the linear subspace $\{\boldsymbol{x} : \Phi\boldsymbol{x} = \boldsymbol{y}\}$ on one of the corners that are on axes.



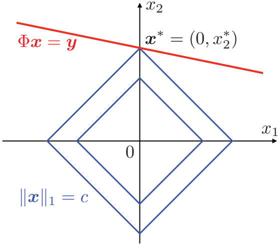
- The optimal solution has one 0 element $\rightarrow$ sparse.
- From this example, one can intuitively say that ℓ^1 optimization can promote sparsity.

ℓ^1 optimization and sparsity

- Consider ℓ^1 optimization in $\mathbb{R}^2$

$$\underset{\boldsymbol{x}\in\mathbb{R}^2}{\text{minimize}} \ \ \|\boldsymbol{x}\|_1 \quad \text{subject to} \ \ \Phi\boldsymbol{x} = \boldsymbol{y}.$$

- The contour $\{\boldsymbol{x} : \|\boldsymbol{x}\|_1 = c\}$ touches the linear subspace $\{\boldsymbol{x} : \Phi\boldsymbol{x} = \boldsymbol{y}\}$ on one of the corners that are on axes.



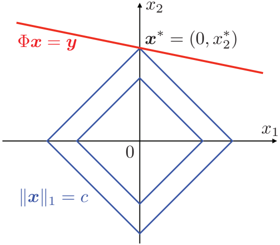
- The optimal solution has one 0 element → sparse.
- From this example, one can intuitively say that ℓ^1 optimization can promote sparsity.

ℓ^1 optimization and sparsity

- Consider ℓ^1 optimization in $\mathbb{R}^2$

$$\underset{\boldsymbol{x}\in\mathbb{R}^2}{\text{minimize}} \ \|\boldsymbol{x}\|_1 \quad \text{subject to} \ \ \Phi\boldsymbol{x} = \boldsymbol{y}.$$

- The contour $\{\boldsymbol{x} : \|\boldsymbol{x}\|_1 = c\}$ touches the linear subspace $\{\boldsymbol{x} : \Phi\boldsymbol{x} = \boldsymbol{y}\}$ on one of the corners that are on axes.



- The optimal solution has one 0 element → sparse.
- From this example, one can intuitively say that ℓ^1 optimization can promote sparsity.

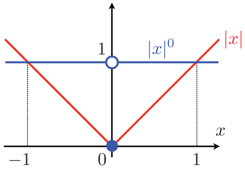
Relation between ℓ^0 and ℓ^1

- The ℓ^0 norm

$$\|\boldsymbol{x}\|_0 = \sum_{i=1}^{n} |x_i|^0$$

- The ℓ^1 norm

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$$



- $\|\boldsymbol{x}\|_1$ has the minimum exponent $p = 1$ among all ℓ^p norms that are convex.

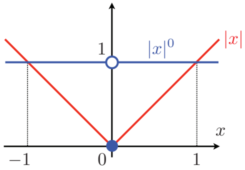
Relation between ℓ^0 and ℓ^1

- The ℓ^0 norm

$$\|\boldsymbol{x}\|_0 = \sum_{i=1}^{n} |x_i|^0$$

- The ℓ^1 norm

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$$



- $\|x\|_1$ has the minimum exponent $p = 1$ among all ℓ^p norms that are convex.

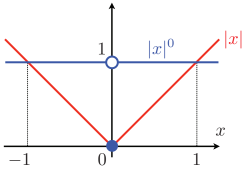
Relation between ℓ^0 and ℓ^1

- The ℓ^0 norm

$$\|\boldsymbol{x}\|_0 = \sum_{i=1}^{n} |x_i|^0$$

- The ℓ^1 norm

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$$



- $\|\boldsymbol{x}\|_1$ has the minimum exponent $p = 1$ among all ℓ^p norms that are convex.

ℓ^1 Regularization (LASSO)

- For noisy data, we consider ℓ^0 regularization:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi x - y\|_2^2 + \lambda \|x\|_0.$$

  to obtain a sparse solution.

- ℓ^1 relaxation

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi x - y\|_2^2 + \lambda \|x\|_1$$

  This is called the ℓ^1 regularization, or LASSO (Least Absolute Shrinkage and Selection Operator).

- This is a convex optimization problem.

ℓ^1 Regularization (LASSO)

- For noisy data, we consider ℓ^0 regularization:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi x - y\|_2^2 + \lambda \|x\|_0.$$

  to obtain a sparse solution.

- ℓ^1 relaxation

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi x - y\|_2^2 + \lambda \|x\|_1$$

  This is called the ℓ^1 regularization, or LASSO (Least Absolute Shrinkage and Selection Operator).

- This is a convex optimization problem.

ℓ^1 Regularization (LASSO)

- For noisy data, we consider ℓ^0 regularization:

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi\boldsymbol{x}-\boldsymbol{y}\|_2^2+\lambda\|\boldsymbol{x}\|_0.$$

  to obtain a sparse solution.
- ℓ^1 relaxation

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \ \frac{1}{2}\|\Phi\boldsymbol{x}-\boldsymbol{y}\|_2^2+\lambda\|\boldsymbol{x}\|_1$$

  This is called the ℓ^1 regularization, or LASSO (Least Absolute Shrinkage and Selection Operator).
- This is a convex optimization problem.

Table of Contents

ℓ^1 optimization by CVX

ℓ^1 Optimization

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \|x\|_1 \quad \text{subject to} \quad \Phi x = y.$$

The MATLAB CVX[1] code

```
cvx_begin
    variable x(n)
    minimize( norm(x, 1) )
    subject to
        y == Phi * x
cvx_end
```

[1] M. Grant & S. Boyd, http://cvxr.com/cvx, 2013.

ℓ^1 optimization by CVX

ℓ^1 Optimization

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \|x\|_1 \quad \text{subject to} \quad \Phi x = y.$$

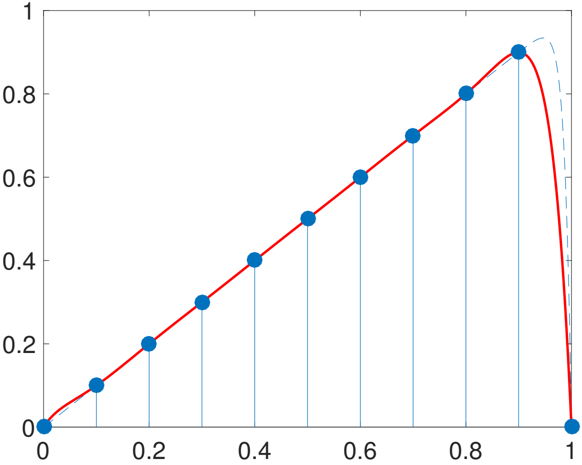
The MATLAB CVX[1] code

```
cvx_begin
    variable x(n)
    minimize( norm(x, 1) )
    subject to
        y == Phi * x
cvx_end
```

[1]M. Grant & S. Boyd, http://cvxr.com/cvx, 2013.

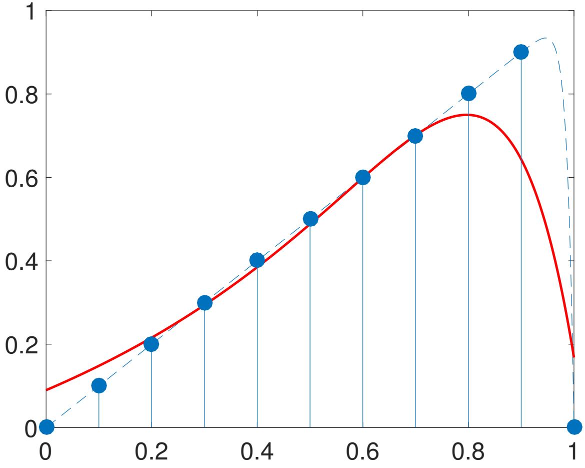
Sparse polynomial interpolation $y = t^{80} - 1$

- 10th-order interpolating polynomial



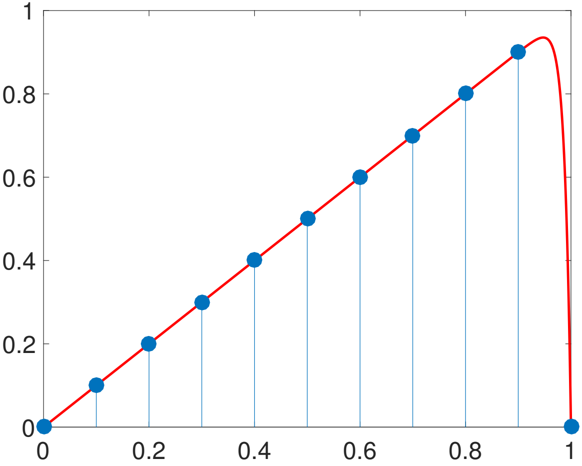
Sparse polynomial interpolation $y = t^{80} - 1$

- regularized least squares (10th order)



Sparse polynomial interpolation $y = t^{80} - 1$

- ℓ^1 optimization (80th order)



Summary

- Curve fitting is formulated as an optimization problem to choose one solution among (infinitely many) candidates.
- Regularization is used for avoiding over fitting.
- Sparse optimization is reduced to ℓ^1 optimization, which is convex and efficiently solved by numerical optimization.

Summary

- Curve fitting is formulated as an optimization problem to choose one solution among (infinitely many) candidates.
- Regularization is used for avoiding over fitting.
- Sparse optimization is reduced to ℓ^1 optimization, which is convex and efficiently solved by numerical optimization.

Summary

- Curve fitting is formulated as an optimization problem to choose one solution among (infinitely many) candidates.
- Regularization is used for avoiding over fitting.
- Sparse optimization is reduced to ℓ^1 optimization, which is convex and efficiently solved by numerical optimization.