Optimalizace

Použití lineární úlohy nejmenších čtverců (a podobných)

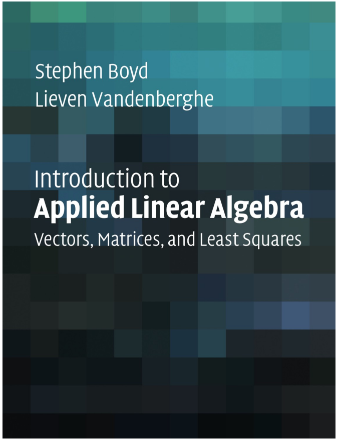
Tom Werner

FEL ČVUT

Mnoho aplikací úlohy

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

je v knize (zdarma ke stažení i se slajdy):



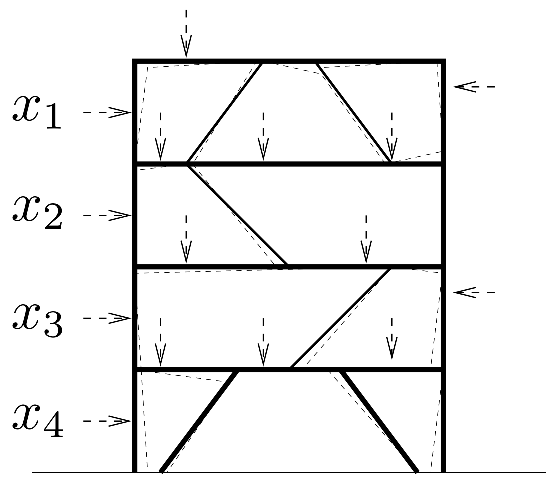
Slides on the following pages are compiled from various courses by S.Boyd and L.Vanderberghe.

Lecture 2
Linear functions and examples

- linear equations and functions
- engineering examples
- interpretations

Linear elastic structure

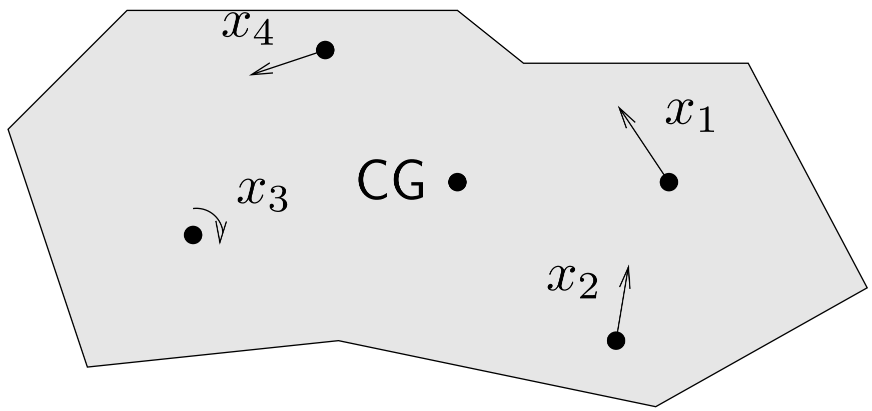
- x_j is external force applied at some node, in some fixed direction
- y_i is (small) deflection of some node, in some fixed direction



(provided x, y are small) we have $y \approx Ax$

- A is called the *compliance matrix*
- a_{ij} gives deflection i per unit force at j (in m/N)

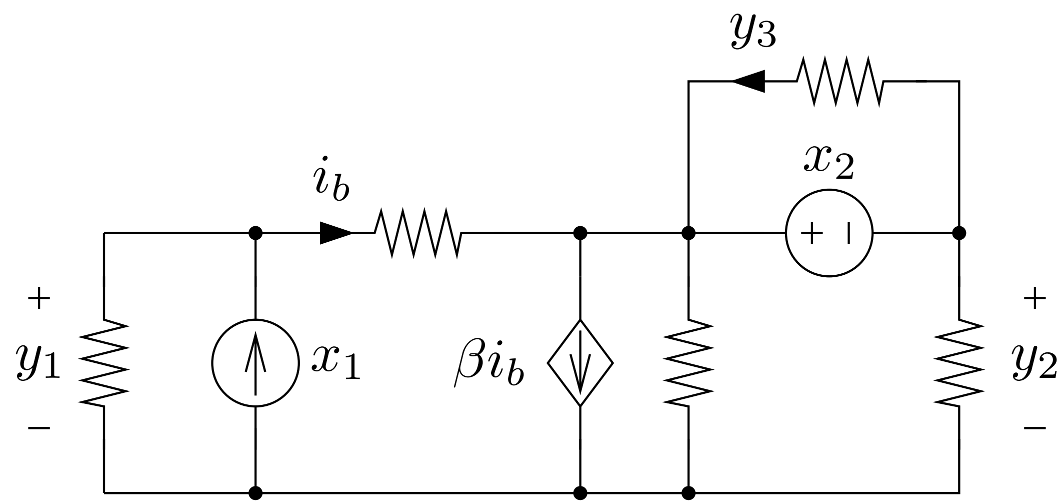
Total force/torque on rigid body



- x_j is external force/torque applied at some point/direction/axis
- $y \in \mathbf{R}^6$ is resulting total force & torque on body
  ($y_1,\ y_2,\ y_3$ are $\mathbf{x}$-, $\mathbf{y}$-, $\mathbf{z}$- components of total force,
  $y_4,\ y_5,\ y_6$ are $\mathbf{x}$-, $\mathbf{y}$-, $\mathbf{z}$- components of total torque)
- we have $y = Ax$
- A depends on geometry
  (of applied forces and torques with respect to center of gravity CG)
- jth column gives resulting force & torque for unit force/torque j

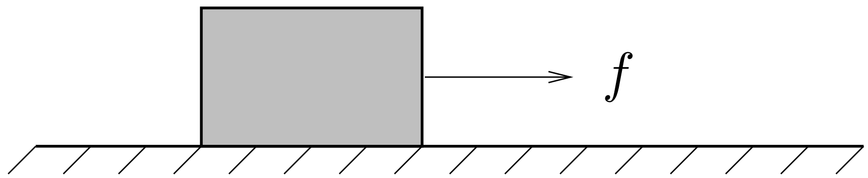
Linear static circuit

interconnection of resistors, linear dependent (controlled) sources, and independent sources



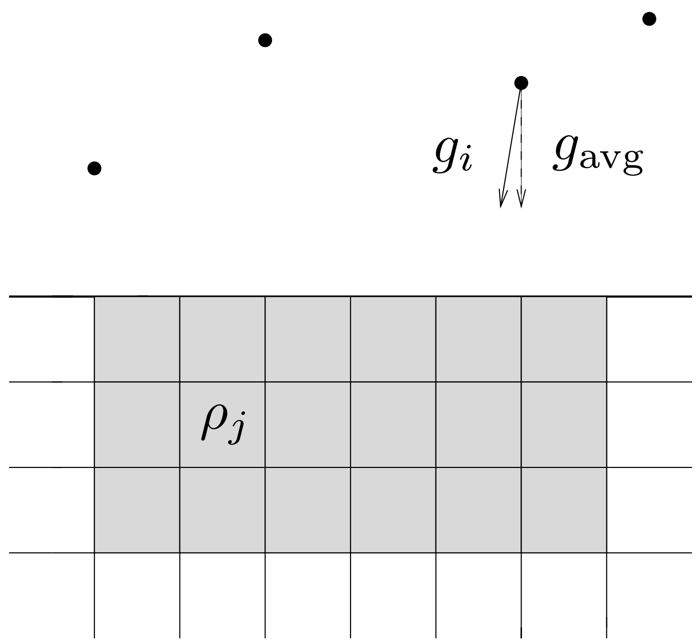
- x_j is value of independent source j
- y_i is some circuit variable (voltage, current)
- we have $y = Ax$
- if x_j are currents and y_i are voltages, A is called the *impedance* or *resistance* matrix

Final position/velocity of mass due to applied forces



- unit mass, zero position/velocity at $t = 0$, subject to force $f(t)$ for $0 \leq t \leq n$
- $f(t) = x_j$ for $j - 1 \leq t < j$, $j = 1, \dots, n$
  (x is the sequence of applied forces, constant in each interval)
- y_1, y_2 are final position and velocity ($i.e.$, at $t = n$)
- we have $y = Ax$
- a_{1j} gives influence of applied force during $j - 1 \leq t < j$ on final position
- a_{2j} gives influence of applied force during $j - 1 \leq t < j$ on final velocity

Gravimeter prospecting



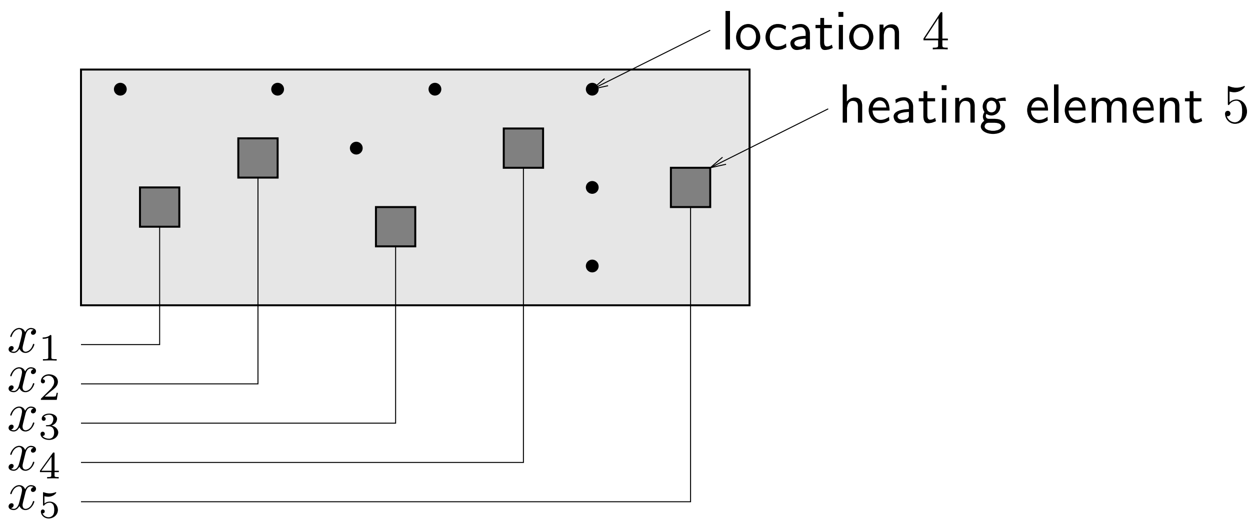
- $x_j = \rho_j - \rho_{\text{avg}}$ is (excess) mass density of earth in voxel j;
- y_i is measured *gravity anomaly* at location i, $i.e.$, some component (typically vertical) of $g_i - g_{\text{avg}}$
- $y = Ax$

- A comes from physics and geometry

- jth column of A shows sensor readings caused by unit density anomaly at voxel j

- ith row of A shows sensitivity pattern of sensor i

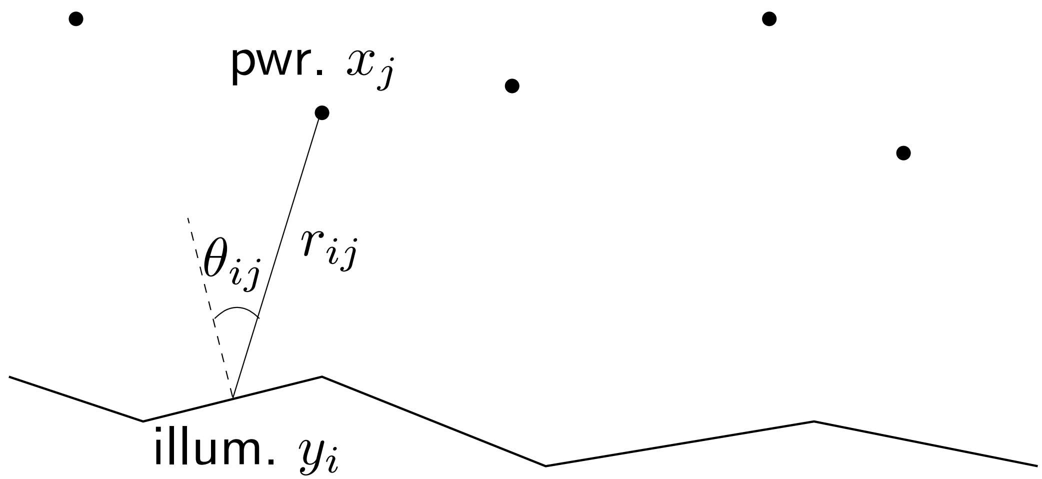
Thermal system



- x_j is power of jth heating element or heat source
- y_i is change in steady-state temperature at location i
- thermal transport via conduction
- $y = Ax$

- a_{ij} gives influence of heater j at location i (in °C/W)
- jth column of A gives pattern of steady-state temperature rise due to 1W at heater j
- ith row shows how heaters affect location i

Illumination with multiple lamps



- n lamps illuminating m (small, flat) patches, no shadows
- x_j is power of jth lamp; y_i is illumination level of patch i
- $y = Ax$, where $a_{ij} = r_{ij}^{-2} \max\{\cos\theta_{ij}, 0\}$

  ($\cos\theta_{ij} < 0$ means patch i is shaded from lamp j)
- jth column of A shows illumination pattern from lamp j

Broad categories of applications

linear model or function $y = Ax$

some broad categories of applications:

- estimation or inversion
- control or design
- mapping or transformation

(this list is not exclusive; can have combinations . . .)

Estimation or inversion

$$y = Ax$$

- y_i is ith measurement or sensor reading (which we know)
- x_j is jth parameter to be estimated or determined
- a_{ij} is sensitivity of ith sensor to jth parameter

sample problems:

- find x, given y
- find all x's that result in y ($i.e.$, all x's consistent with measurements)
- if there is no x such that $y = Ax$, find x s.t. $y \approx Ax$ ($i.e.$, if the sensor readings are inconsistent, find x which is almost consistent)

Control or design

$$y = Ax$$

- x is vector of design parameters or inputs (which we can choose)
- y is vector of results, or outcomes
- A describes how input choices affect results

sample problems:

- find x so that $y = y_{\mathrm{des}}$
- find all x's that result in $y = y_{\mathrm{des}}$ (*i.e.*, find all designs that meet specifications)
- among x's that satisfy $y = y_{\mathrm{des}}$, find a small one (*i.e.*, find a small or efficient x that meets specifications)

Mapping or transformation

- x is mapped or transformed to y by linear function $y = Ax$

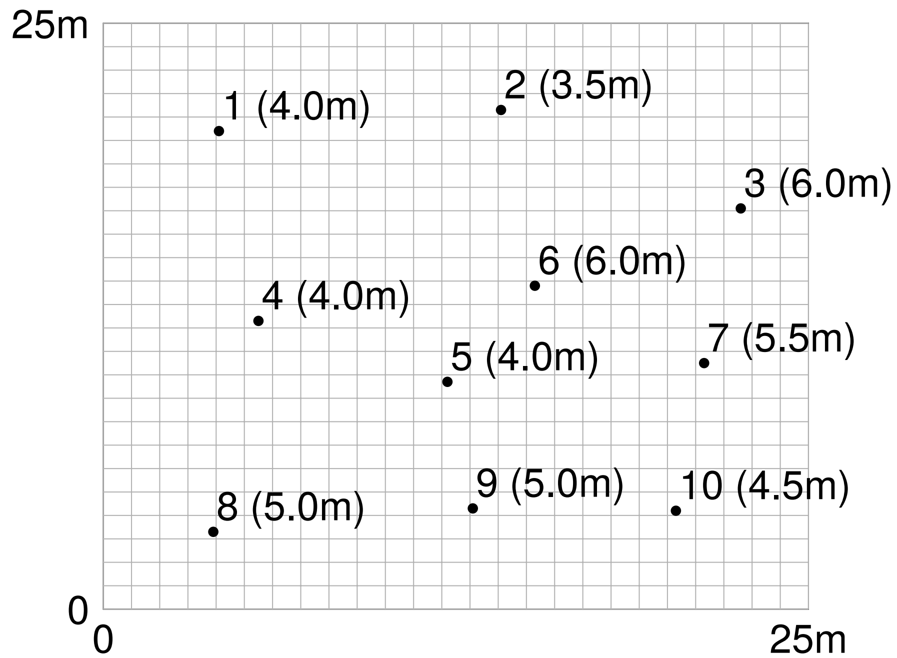
sample problems:

- determine if there is an x that maps to a given y
- (if possible) find *an* x that maps to y
- find *all* x's that map to a given y
- if there is only one x that maps to y, find it ($i.e.$, decode or undo the mapping)

Example: illumination

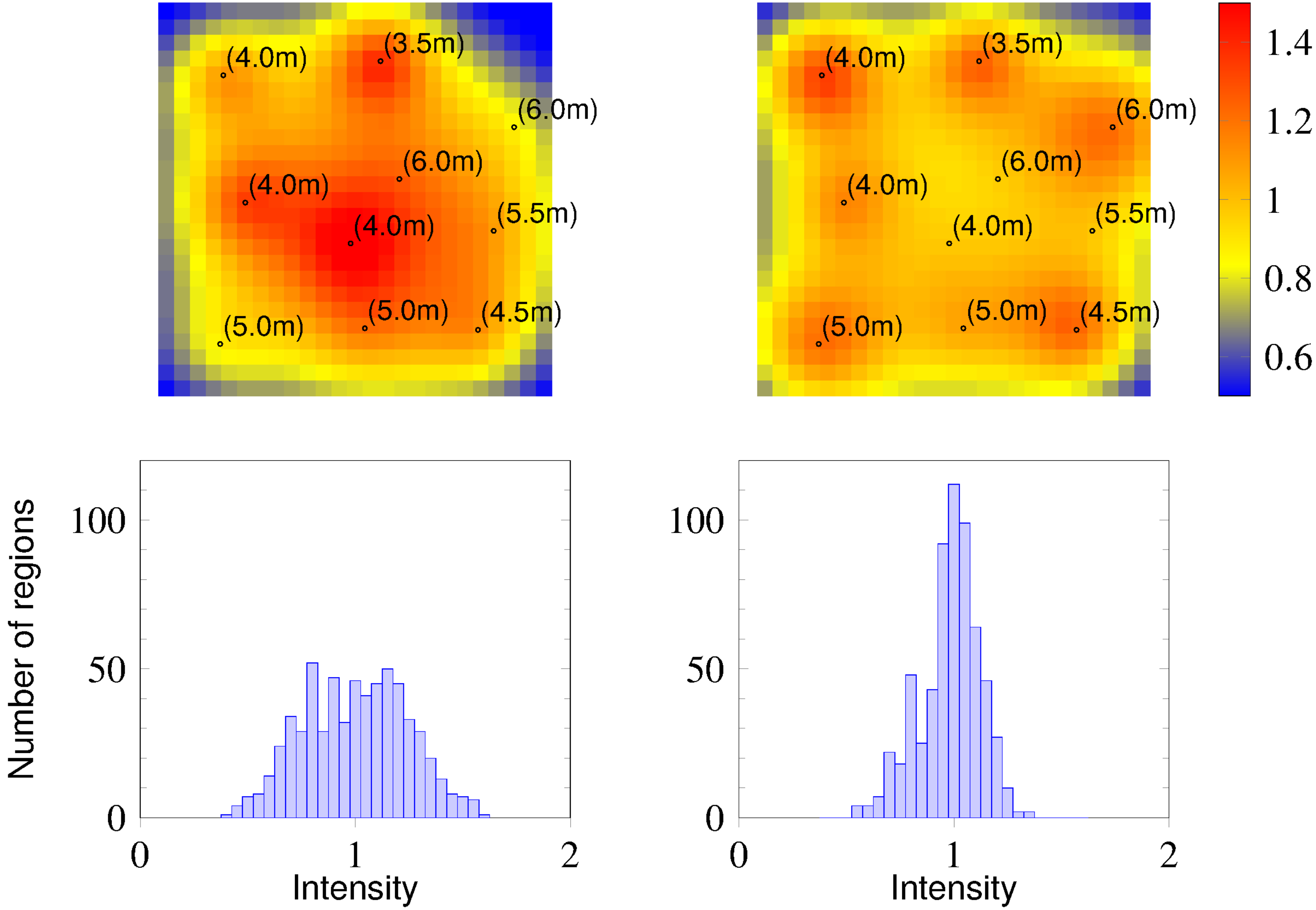
- n lamps at given positions above an area divided in m regions
- A_{ij} is illumination in region i if lamp j is on with power 1 and other lamps are off
- x_j is power of lamp j
- $(Ax)_i$ is illumination level at region i
- b_i is target illumination level at region i

Example: $m = 25^2$, $n = 10$; figure shows position and height of each lamp



Example: illumination

- left: illumination pattern for equal lamp powers ($x = \mathbf{1}$)
- right: illumination pattern for least squares solution $\hat{x}$, with $b = \mathbf{1}$



Linear-in-parameters model

we choose the model $\hat{f}(x)$ from a family of models

$$\hat{f}(x) = \theta_1 f_1(x) + \theta_2 f_2(x) + \cdots + \theta_p f_p(x)$$

- the functions f_i are scalar valued *basis functions* (chosen by us)
- the basis functions often include a constant function (typically, $f_1(x) = 1$)
- the coefficients $\theta_1, \ldots, \theta_p$ are the model *parameters*
- the model $\hat{f}(x)$ is linear in the parameters θ_i
- if $f_1(x) = 1$, this can be interpreted as a regression model

$$\hat{y} = \beta^T \tilde{x} + v$$

with parameters $v = \theta_1$, $\beta = \theta_{2:p}$ and new features $\tilde{x}$ generated from x:

$$\tilde{x}_1 = f_2(x), \quad \ldots, \quad \tilde{x}_p = f_p(x)$$

Least squares model fitting

- fit linear-in-parameters model to data set $(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})$
- residual for data sample i is

$$r^{(i)} = y^{(i)} - \hat{f}(x^{(i)}) = y^{(i)} - \theta_1 f_1(x^{(i)}) - \cdots - \theta_p f_p(x^{(i)})$$

- least squares model fitting: choose parameters θ by minimizing MSE

$$\frac{1}{N}\left((r^{(1)})^2 + (r^{(2)})^2 + \cdots + (r^{(N)})^2\right)$$

- this is a least squares problem: minimize $\|A\theta - y^{\mathrm{d}}\|^2$ with

$$A = \begin{bmatrix} f_1(x^{(1)}) & \cdots & f_p(x^{(1)}) \\ f_1(x^{(2)}) & \cdots & f_p(x^{(2)}) \\ \vdots & & \vdots \\ f_1(x^{(N)}) & \cdots & f_p(x^{(N)}) \end{bmatrix}, \qquad \theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_p \end{bmatrix}, \qquad y^{\mathrm{d}} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$$

Example: polynomial approximation

$$\hat{f}(x) = \theta_1 + \theta_2 x + \theta_3 x^2 + \cdots + \theta_p x^{p-1}$$

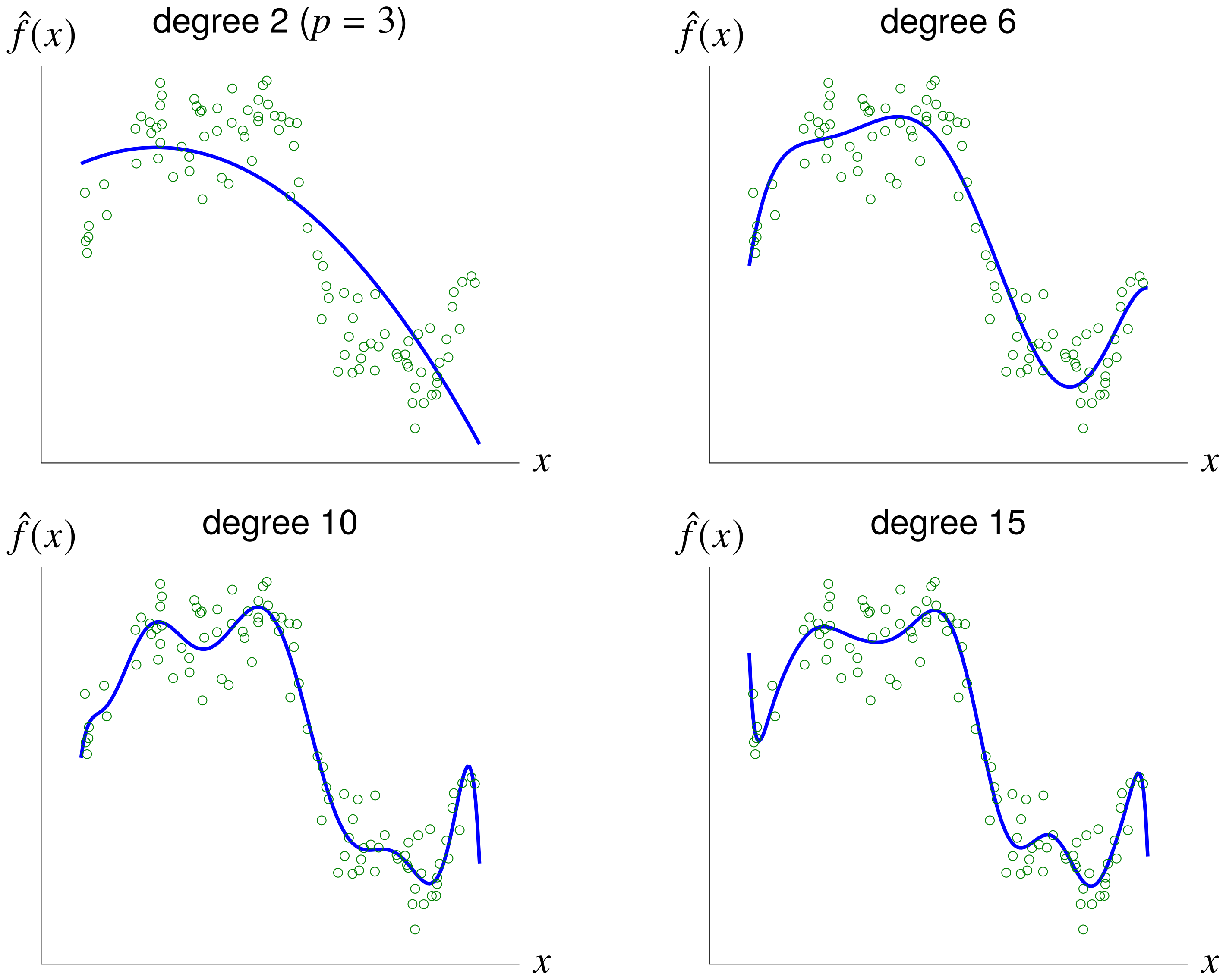
- a linear-in-parameters model with basis functions $1, x, \ldots, x^{p-1}$
- least squares model fitting: choose parameters θ by minimizing MSE

$$\frac{1}{N}\left((y^{(1)} - \hat{f}(x^{(1)}))^2 + (y^{(2)} - \hat{f}(x^{(2)}))^2 + \cdots + (y^{(N)} - \hat{f}(x^{(N)}))^2\right)$$

- in matrix notation: minimize $\|A\theta - y^{\mathrm{d}}\|^2$ with

$$A = \begin{bmatrix} 1 & x^{(1)} & (x^{(1)})^2 & \cdots & (x^{(1)})^{p-1} \\ 1 & x^{(2)} & (x^{(2)})^2 & \cdots & (x^{(2)})^{p-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x^{(N)} & (x^{(N)})^2 & \cdots & (x^{(N)})^{p-1} \end{bmatrix}, \qquad y^{\mathrm{d}} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$$

Example



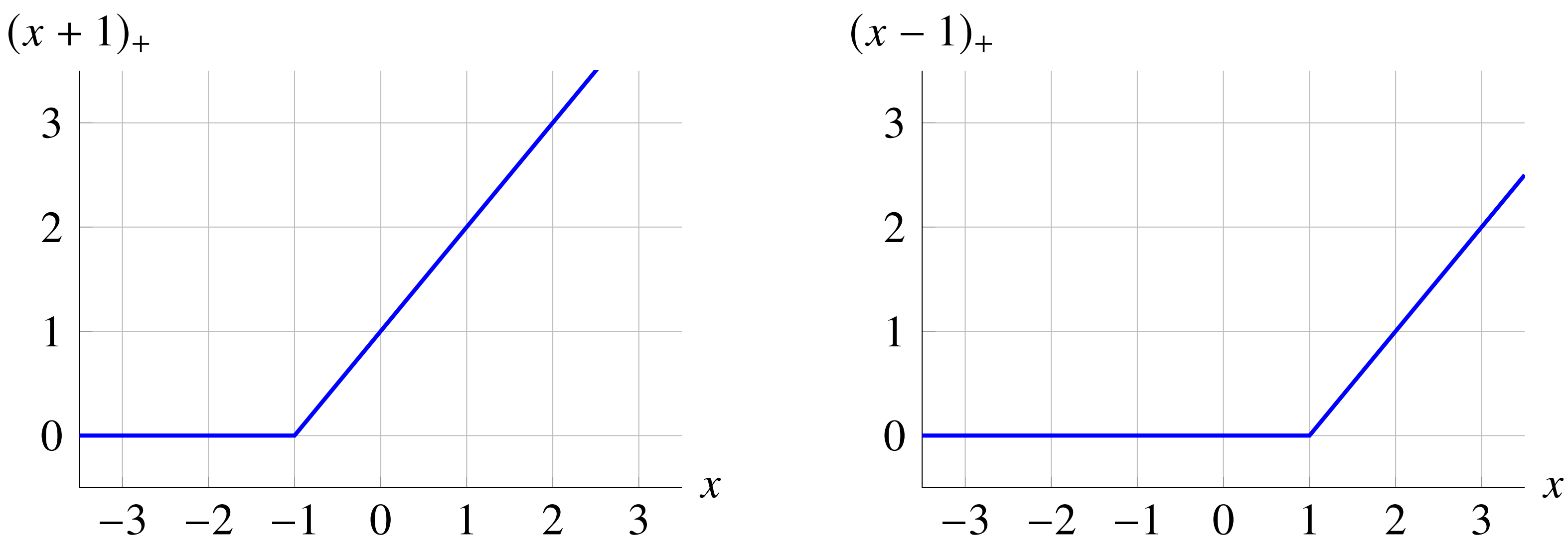
data set of 100 examples

Piecewise-affine function

- define *knot points* $a_1 < a_2 < \cdots < a_k$ on the real axis
- piecewise-affine function is continuous, and affine on each interval $[a_k, a_{k+1}]$
- piecewise-affine function with knot points $a_1, \ldots, a_k$ can be written as

$$\hat{f}(x) = \theta_1 + \theta_2 x + \theta_3 (x - a_1)_+ + \cdots + \theta_{2+k}(x - a_k)_+$$

where $u_+ = \max\{u, 0\}$

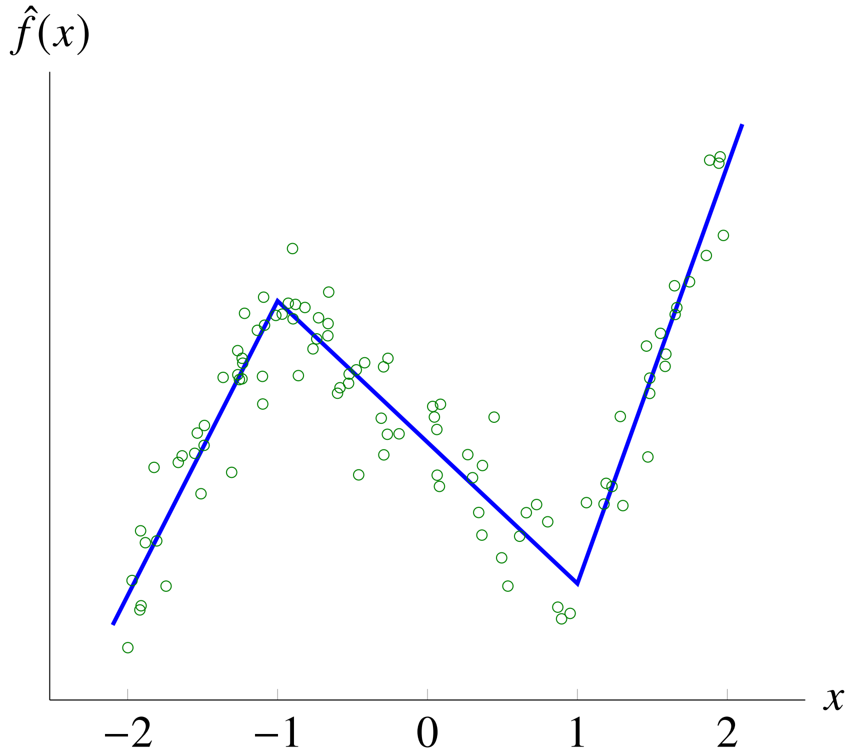


Piecewise-affine function fitting

piecewise-affine model is in linear in the parameters θ, with basis functions

$$f_1(x) = 1, \quad f_2(x) = x, \quad f_3(x) = (x - a_1)_+, \quad \ldots, \quad f_{k+2}(x) = (x - a_k)_+$$

Example: fit piecewise-affine function with knots $a_1 = -1$, $a_2 = 1$ to 100 points



Generalization and validation

Generalization ability: ability of model to predict outcomes for new, unseen data

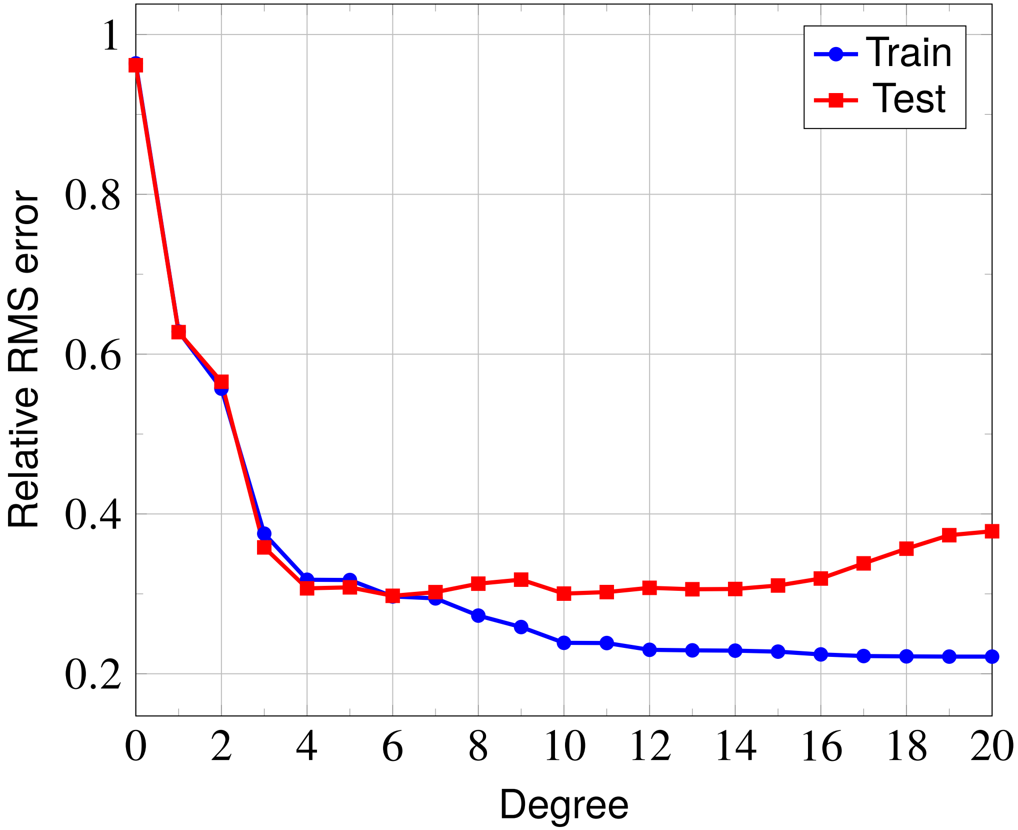
Model validation: to assess generalization ability,

- divide data in two sets: *training set* and *test (or validation)* set
- use training set to fit model
- use test set to get an idea of generalization ability
- this is also called *out-of-sample validation*

Over-fit model

- model with low prediction error on training set, bad generalization ability
- prediction error on training set is much smaller than on test set

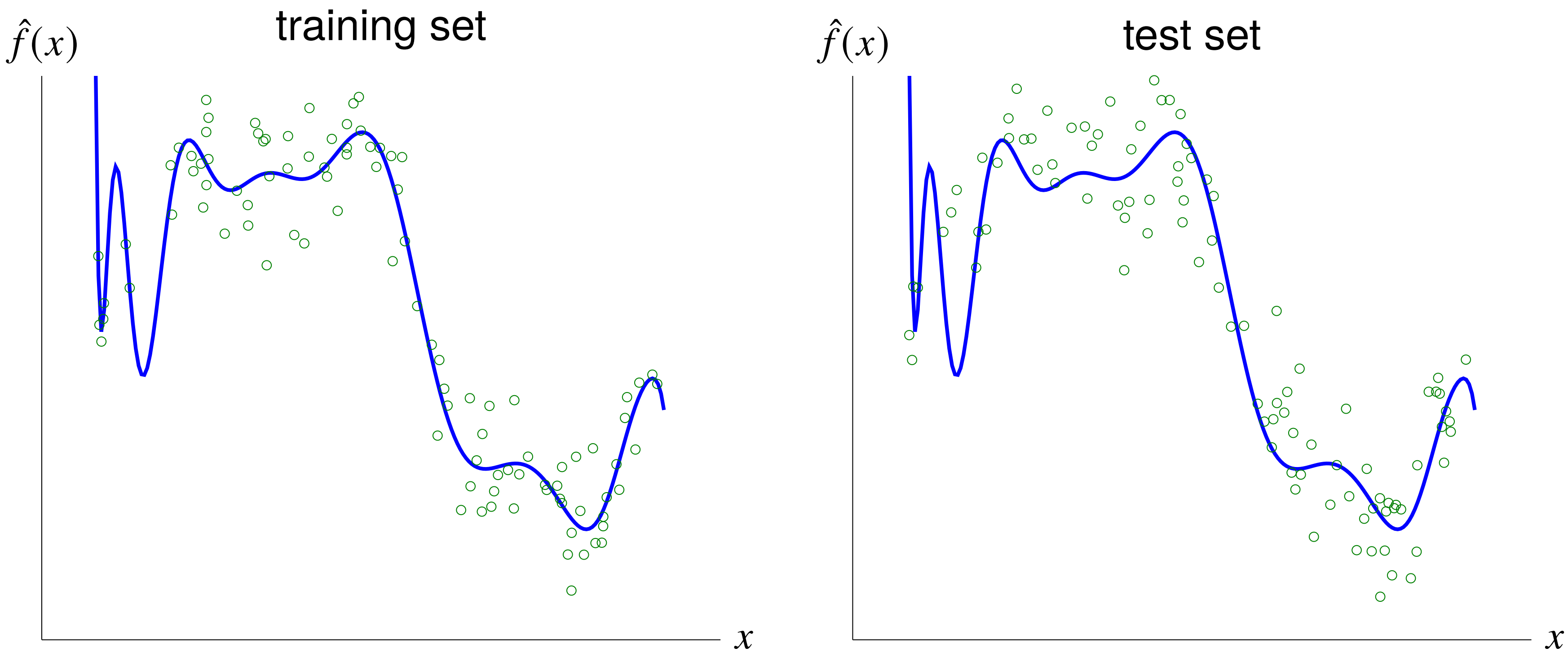
Example: polynomial fitting



- training set is data set of 100 points used on page 9.11
- test set is a similar set of 100 points
- plot suggests using degree 6

Over-fitting

polynomial of degree 20 on training and test set



over-fitting is evident at the left end of the interval

Auto-regressive (AR) time series model

$$\hat{z}_{t+1} = \beta_1 z_t + \cdots + \beta_M z_{t-M+1}, \qquad t = M, M+1, \ldots$$

- $z_1, z_2, \ldots$ is a time series
- $\hat{z}_{t+1}$ is a prediction of z_{t+1}, made at time t
- prediction $\hat{z}_{t+1}$ is a linear function of previous M values $z_t, \ldots, z_{t-M+1}$
- M is the *memory* of the model

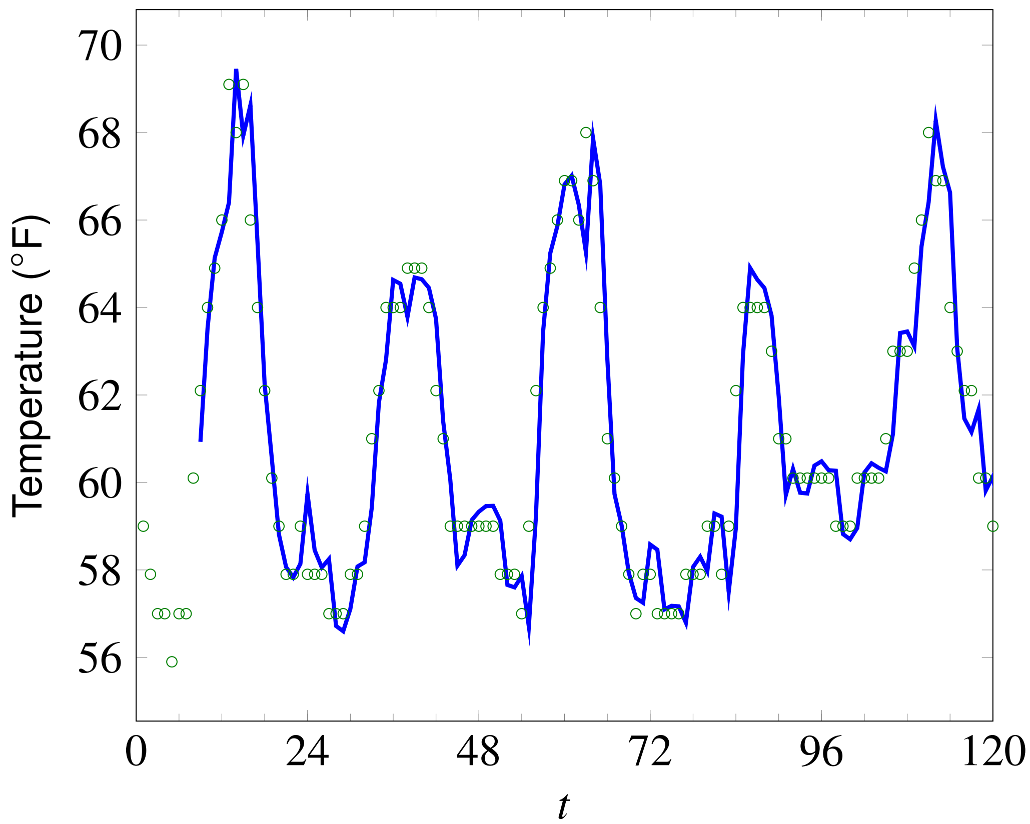
Least squares fitting of AR model: given oberved data $z_1, \ldots, z_T$, minimize

$$(z_{M+1} - \hat{z}_{M+1})^2 + (z_{M+2} - \hat{z}_{M+2})^2 + \cdots + (z_T - \hat{z}_T)^2$$

this is a least squares problem: minimize $\|A\beta - y^{\mathrm{d}}\|^2$ with

$$A = \begin{bmatrix} z_M & z_{M-1} & \cdots & z_1 \\ z_{M+1} & z_M & \cdots & z_2 \\ \vdots & \vdots & & \vdots \\ z_{T-1} & z_{T-2} & \cdots & z_{T-M} \end{bmatrix}, \qquad \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_M \end{bmatrix}, \qquad y^{\mathrm{d}} = \begin{bmatrix} z_{M+1} \\ z_{M+2} \\ \vdots \\ z_T \end{bmatrix}$$

Example: hourly temperature at LAX



- blue line shows prediction by AR model of memory $M = 8$
- model was fit on time series of length $T = 744$ (May 1–31, 2016)
- plot shows first five days

10. Multi-objective least squares

- multi-objective least squares
- regularized data fitting
- control
- estimation and inversion

Multi-objective least squares

we have several objectives

$$J_1 = \|A_1 x - b_1\|^2, \qquad \ldots, \qquad J_k = \|A_k x - b_k\|^2$$

- A_i is an $m_i \times n$ matrix, b_i is an m_i-vector
- we seek *one* x that makes all k objectives small
- usually there is a trade-off: no single x minimizes all objectives simultaneously

Weighted least squares formulation: find x that minimizes

$$\lambda_1 \|A_1 x - b_1\|^2 + \cdots + \lambda_k \|A_k x - b_k\|^2$$

- coefficients $\lambda_1, \ldots, \lambda_k$ are positive weights
- weights λ_i express relative importance of different objectives
- without loss of generality, we can choose $\lambda_1 = 1$

Solution of weighted least squares

- weighted least squares is equivalent to a standard least squares problem

$$\text{minimize} \quad \left\| \begin{bmatrix} \sqrt{\lambda_1} A_1 \\ \sqrt{\lambda_2} A_2 \\ \vdots \\ \sqrt{\lambda_k} A_k \end{bmatrix} x - \begin{bmatrix} \sqrt{\lambda_1} b_1 \\ \sqrt{\lambda_2} b_2 \\ \vdots \\ \sqrt{\lambda_k} b_k \end{bmatrix} \right\|^2$$

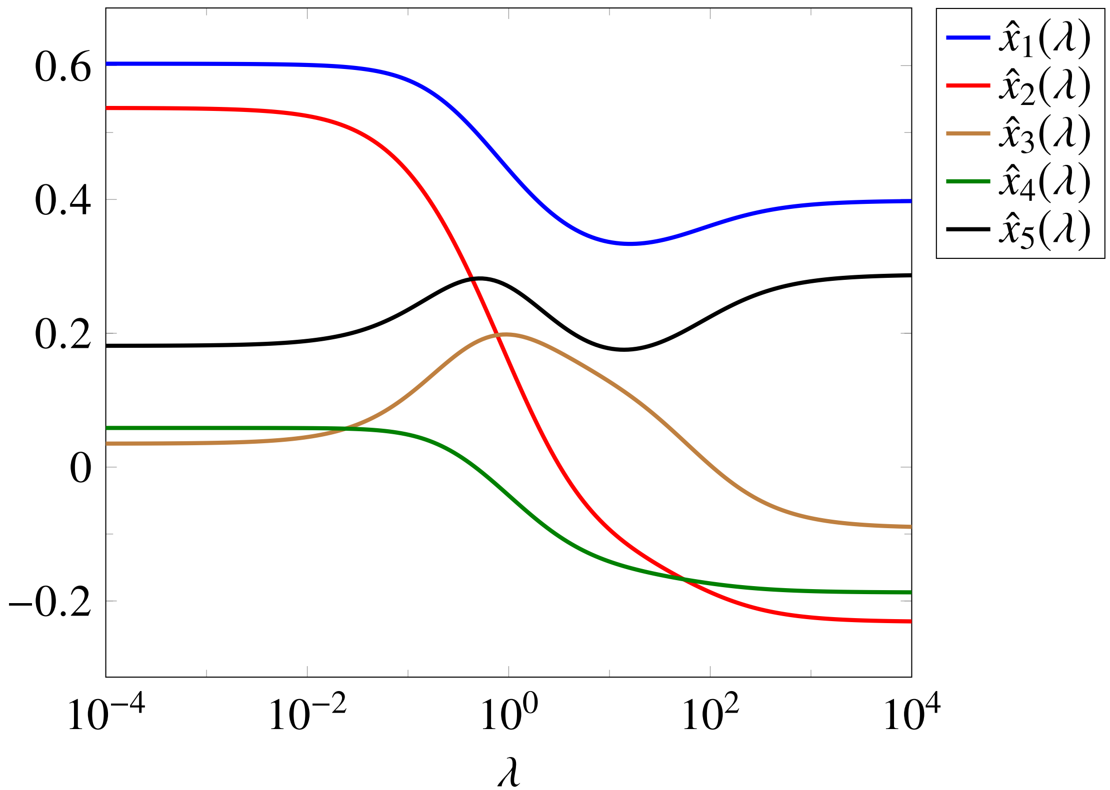
- solution is unique if the *stacked matrix* has linearly independent columns
- each matrix A_i may have linearly dependent columns (or be a wide matrix)
- it the stacked matrix has linearly independent columns, the solution is

$$\hat{x} = \left(\lambda_1 A_1^T A_1 + \cdots + \lambda_k A_k^T A_k\right)^{-1} \left(\lambda_1 A_1^T b_1 + \cdots + \lambda_k A_k^T b_k\right)$$

Example with two objectives

$$\text{minimize} \quad \|A_1 x - b_1\|^2 + \lambda \|A_2 x - b_2\|^2$$

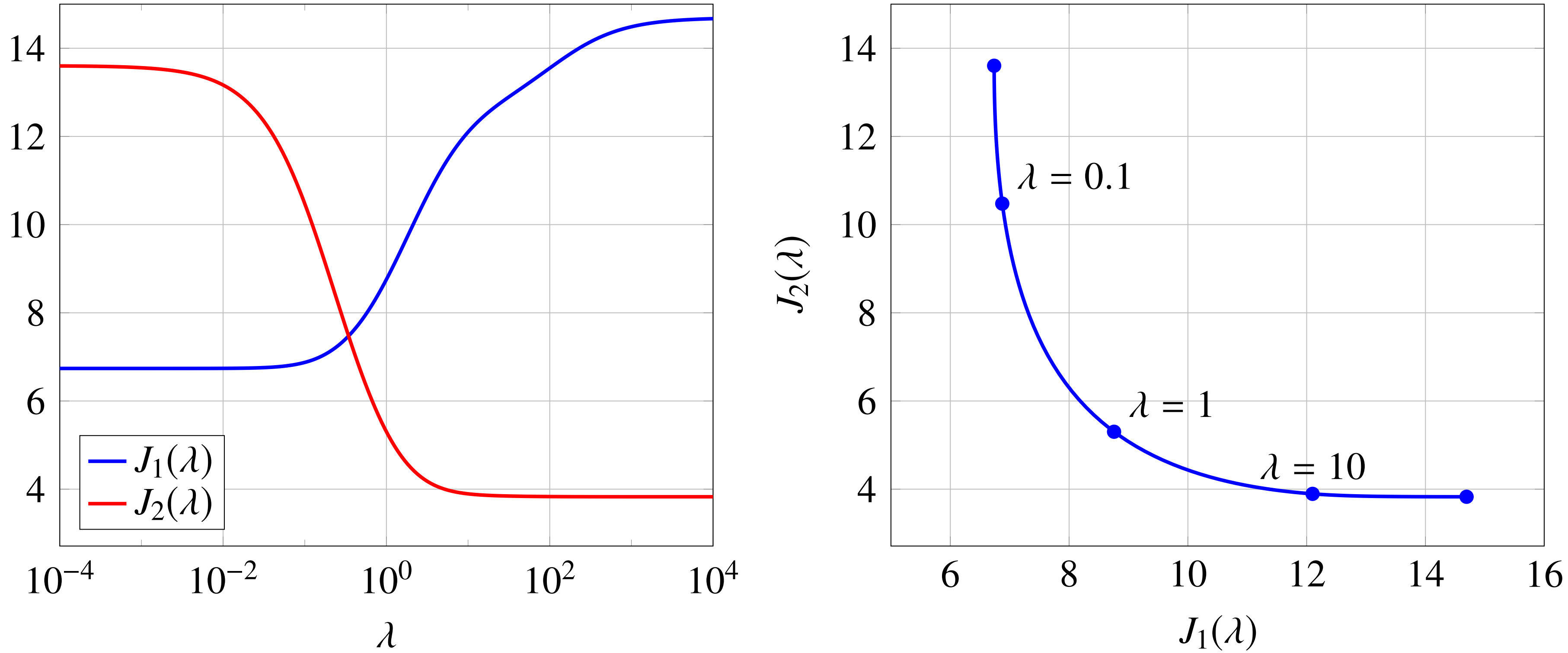
A_1 and A_2 are 10×5



plot shows weighted least squares solution $\hat{x}(\lambda)$ as function of weight λ

Example with two objectives

$$\text{minimize} \quad \|A_1x - b_1\|^2 + \lambda\|A_2x - b_2\|^2$$



- left figure shows $J_1(\lambda) = \|A_1\hat{x}(\lambda) - b_1\|^2$ and $J_2(\lambda) = \|A_2\hat{x}(\lambda) - b_2\|^2$
- right figure shows optimal trade-off curve of $J_2(\lambda)$ versus $J_1(\lambda)$

Outline

- multi-objective least squares
- **regularized data fitting**
- control
- estimation and inversion

Motivation

- consider linear-in-parameters model

$$\hat{f}(x) = \theta_1 f_1(x) + \cdots + \theta_p f_p(x)$$

  we assume $f_1(x)$ is the constant function 1

- we fit the model $\hat{f}(x)$ to examples $(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})$
- large coefficient θ_i makes model more sensitive to changes in $f_i(x)$
- keeping $\theta_2, \ldots, \theta_p$ small helps avoid over-fitting
- this leads to two objectives:

$$J_1(\theta) = \sum_{k=1}^{N} (\hat{f}(x^{(k)}) - y^{(k)})^2, \qquad J_2(\theta) = \sum_{j=2}^{p} \theta_j^2$$

  primary objective $J_1(\theta)$ is sum of squares of prediction errors

Weighted least squares formulation

$$\text{minimize} \quad J_1(\theta) + \lambda J_2(\theta) = \sum_{k=1}^{N} (\hat{f}(x^{(k)}) - y^{(k)})^2 + \lambda \sum_{j=2}^{p} \theta_j^2$$

- λ is positive *regularization parameter*
- equivalent to least squares problem: minimize

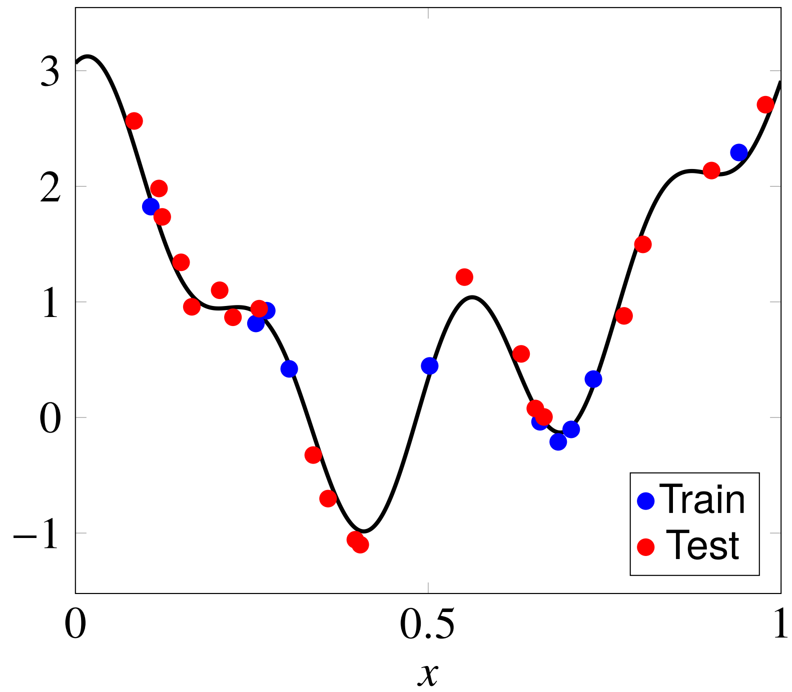
$$\left\| \begin{bmatrix} A_1 \\ \sqrt{\lambda} A_2 \end{bmatrix} \theta - \begin{bmatrix} y^{\mathrm{d}} \\ 0 \end{bmatrix} \right\|^2$$

with $y^{\mathrm{d}} = (y^{(1)}, \ldots, y^{(N)})$,

$$A_1 = \begin{bmatrix} 1 & f_2(x^{(1)}) & \cdots & f_p(x^{(1)}) \\ 1 & f_2(x^{(2)}) & \cdots & f_p(x^{(2)}) \\ \vdots & \vdots & & \vdots \\ 1 & f_2(x^{(N)}) & \cdots & f_p(x^{(N)}) \end{bmatrix}, \qquad A_2 = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

- stacked matrix has linearly independent columns (for positive λ)
- value of λ can be chosen by out-of-sample validation or cross-validation

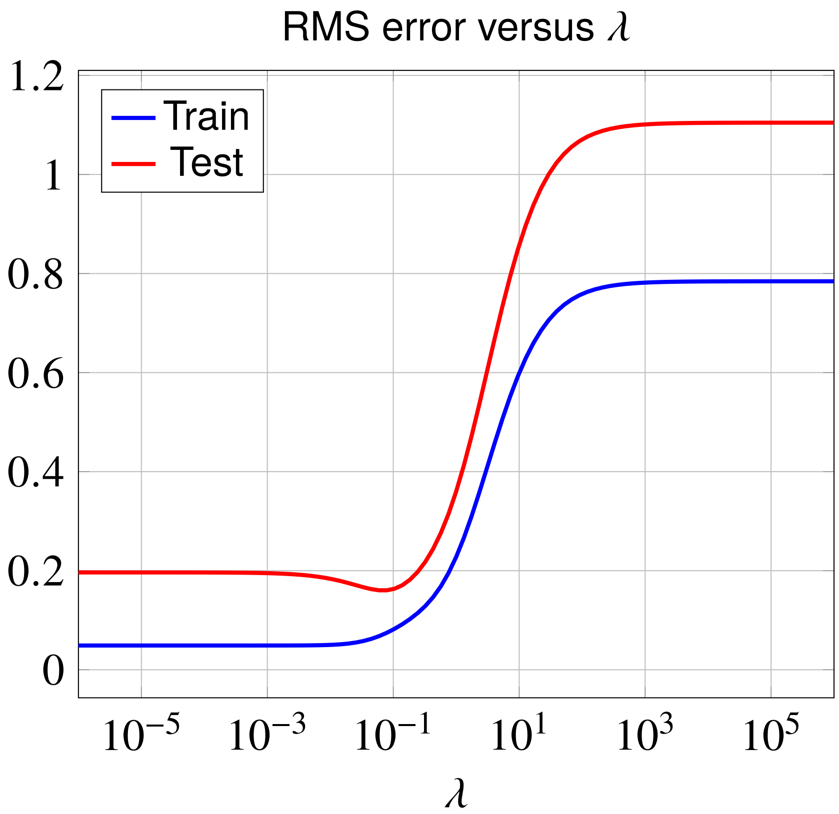
Example

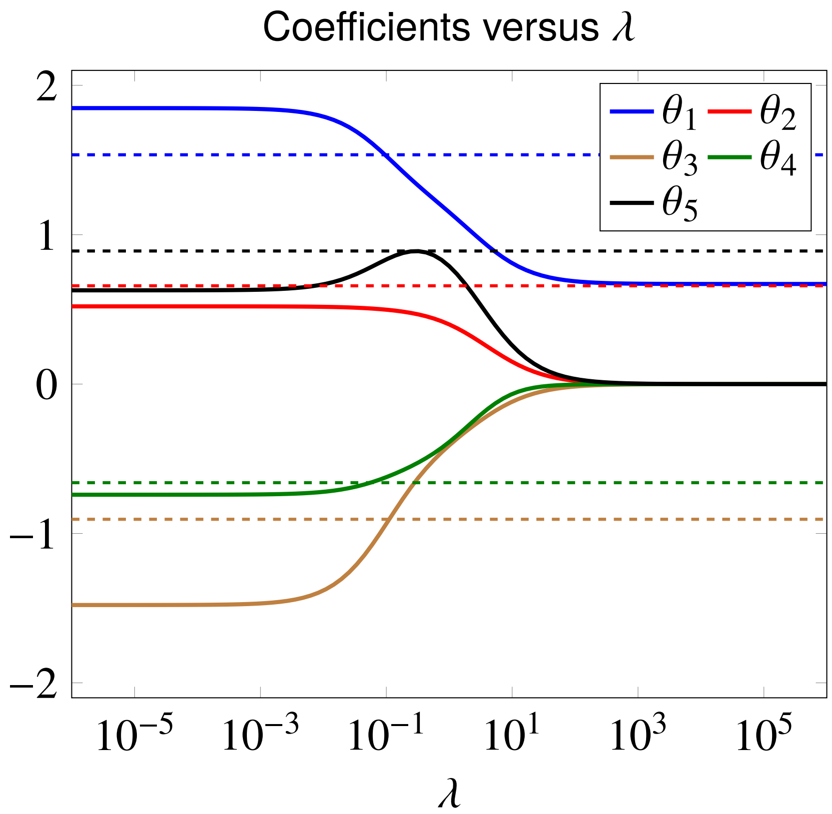


- solid line is signal used to generate synthetic (simulated) data
- 10 blue points are used as training set; 20 red points are used as test set
- we fit a model with five parameters $\theta_1, \ldots, \theta_5$:

$$\hat{f}(x) = \theta_1 + \sum_{k=1}^{4} \theta_{k+1} \sin(\omega_k x + \phi_k) \qquad \text{(with given } \omega_k, \phi_k\text{)}$$

Result of regularized least squares fit





- minimum test RMS error is for λ around 0.08
- increasing λ "shrinks" the coefficients $\theta_2, \ldots, \theta_5$
- dashed lines show coefficients used to generate the data
- for λ near 0.08, estimated coefficients are close to these "true" values

Outline

- multi-objective least squares
- regularized data fitting
- **control**
- estimation and inversion

Control

$$y = Ax + b$$

- x is n-vector of *actions* or *inputs*
- y is m-vector of *results* or *outputs*
- relation between inputs and outputs is a known affine function

the goal is to choose inputs x to optimize different objectives on x and y

Optimal input design

Linear dynamical system

$$y(t) = h_0 u(t) + h_1 u(t-1) + h_2 u(t-2) + \cdots + h_t u(0)$$

- output $y(t)$ and input $u(t)$ are scalar
- we assume input $u(t)$ is zero for $t < 0$
- coefficients $h_0, h_1, \ldots$ are the *impulse response coefficients*
- output is convolution of input with impulse response

Optimal input design

- optimization variable is the input sequence $x = (u(0), u(1), \ldots, u(N))$
- goal is to track a desired output using a small and slowly varying input

Input design objectives

$$\text{minimize} \quad J_{\mathrm{t}}(x) + \lambda_{\mathrm{v}} J_{\mathrm{v}}(x) + \lambda_{\mathrm{m}} J_{\mathrm{m}}(x)$$

- primary objective: track desired output y_{des} over an interval $[0, N]$:

$$J_{\mathrm{t}}(x) = \sum_{t=0}^{N} (y(t) - y_{\mathrm{des}}(t))^2$$

- secondary objectives: use a small and slowly varying input signal:

$$J_{\mathrm{m}}(x) = \sum_{t=0}^{N} u(t)^2, \qquad J_{\mathrm{v}}(x) = \sum_{t=0}^{N-1} (u(t+1) - u(t))^2$$

Tracking error

$$
\begin{aligned}
J_{\mathrm{t}}(x) &= \sum_{t=0}^{N} (y(t) - y_{\mathrm{des}}(t))^2 \\
&= \|A_{\mathrm{t}} x - b_{\mathrm{t}}\|^2
\end{aligned}
$$

with

$$
A_{\mathrm{t}} = \begin{bmatrix}
h_0 & 0 & 0 & \cdots & 0 & 0 \\
h_1 & h_0 & 0 & \cdots & 0 & 0 \\
h_2 & h_1 & h_0 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
h_{N-1} & h_{N-2} & h_{N-3} & \cdots & h_0 & 0 \\
h_N & h_{N-1} & h_{N-2} & \cdots & h_1 & h_0
\end{bmatrix}, \qquad
b_{\mathrm{t}} = \begin{bmatrix}
y_{\mathrm{des}}(0) \\
y_{\mathrm{des}}(1) \\
y_{\mathrm{des}}(2) \\
\vdots \\
y_{\mathrm{des}}(N-1) \\
y_{\mathrm{des}}(N)
\end{bmatrix}
$$

Input variation and magnitude

Input variation

$$J_{\mathrm{v}}(x) = \sum_{t=0}^{N-1} (u(t+1) - u(t))^2 = \|Dx\|^2$$

with D the $N \times (N+1)$ matrix

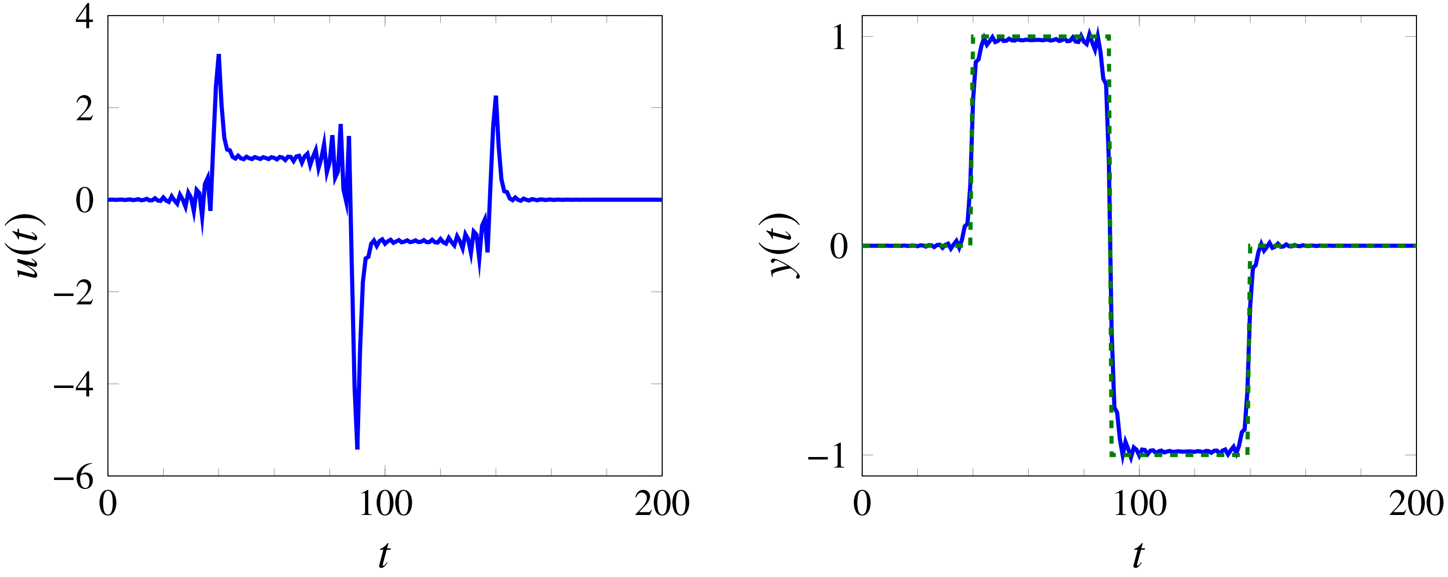
$$D = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 1 \end{bmatrix}$$

Input magnitude

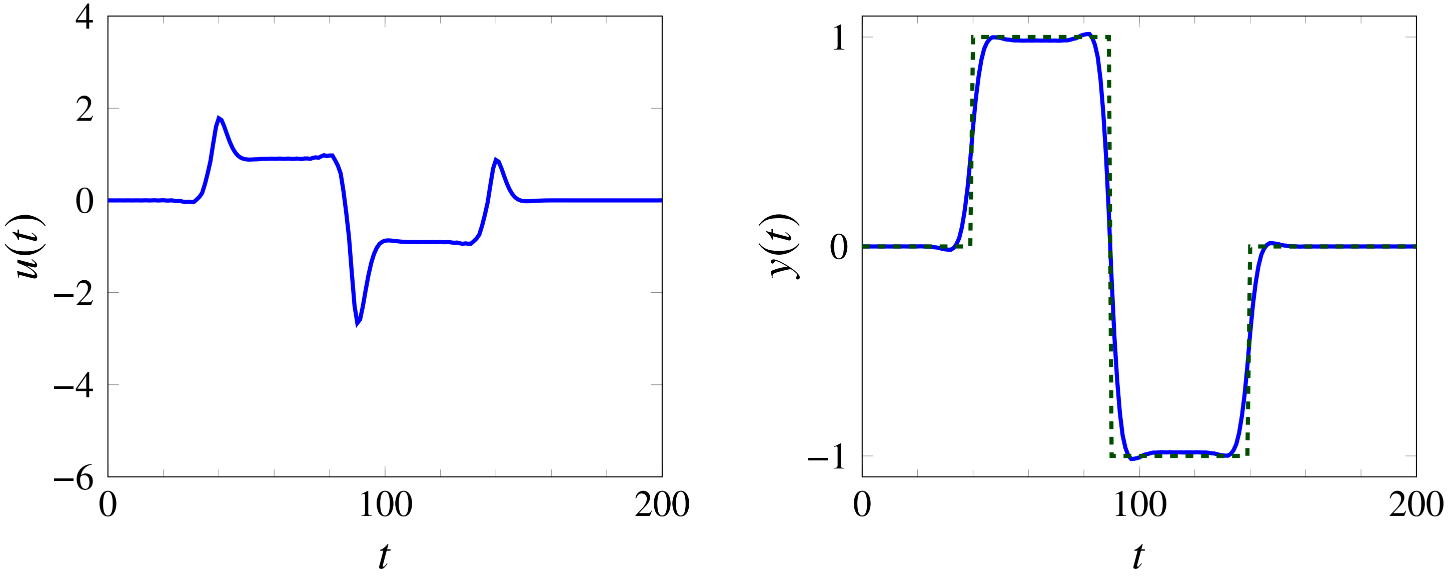
$$J_{\mathrm{m}}(x) = \sum_{t=0}^{N} u(t)^2 = \|x\|^2$$

Example

$\lambda_{\text{v}} = 0$,
small λ_{m}



larger λ_{v}
larger λ_{m}



Outline

- multi-objective least squares
- regularized data fitting
- control
- **estimation and inversion**

Estimation

Linear measurement model

$$y = Ax_{\text{ex}} + v$$

- n-vector x_{ex} contains parameters that we want to estimate
- m-vector v is unknown measurement error or noise
- m-vector y contains measurements
- $m \times n$ matrix A relates measurements and parameters

Least squares estimate: use as estimate of x_{ex} the solution $\hat{x}$ of

$$\text{minimize} \quad \|Ax - y\|^2$$

Regularized estimation

add other terms to $\|Ax - y\|^2$ to include information about parameters

Example: Tikhonov regularization

$$\text{minimize} \quad \|Ax - y\|^2 + \lambda \|x\|^2$$

- goal is to make $\|Ax - y\|$ small with small x
- equivalent to solving

$$(A^T A + \lambda I)x = A^T y$$

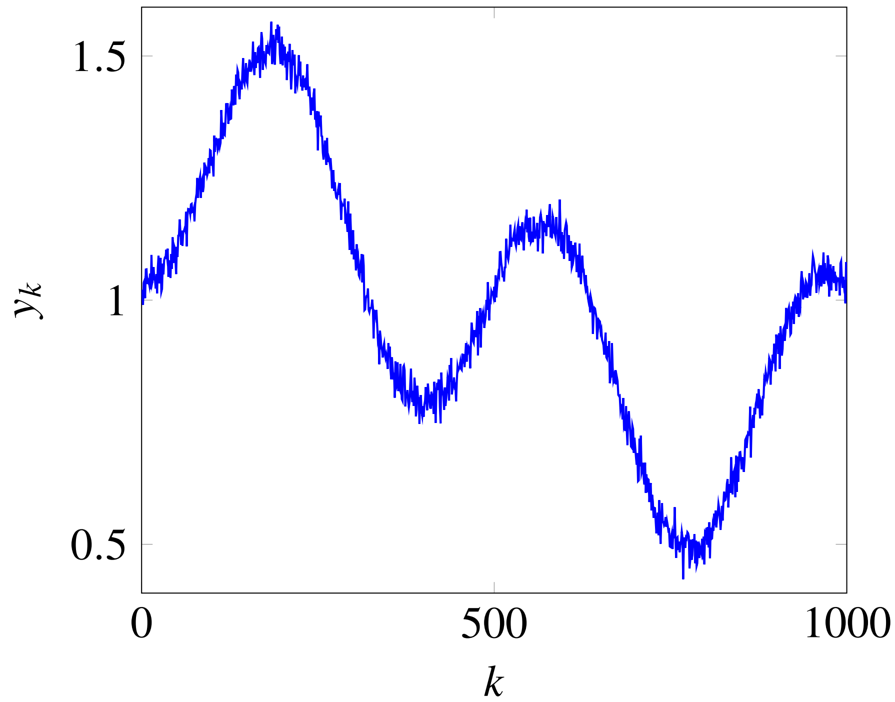
- solution is unique (if $\lambda > 0$) even when A has linearly dependent columns

Signal denoising

- observed signal y is n-vector

$$y = x_{\text{ex}} + v$$

- x_{ex} is unknown signal
- v is noise



Least squares denoising: find estimate $\hat{x}$ by solving

$$\text{minimize} \quad \|x - y\|^2 + \lambda \sum_{i=1}^{n-1} (x_{i+1} - x_i)^2$$

goal is to find slowly varying signal $\hat{x}$, close to observed signal y

Matrix formulation

$$\text{minimize} \quad \left\| \begin{bmatrix} I \\ \sqrt{\lambda} D \end{bmatrix} x - \begin{bmatrix} y \\ 0 \end{bmatrix} \right\|^2$$

- D is $(n-1) \times n$ finite difference matrix

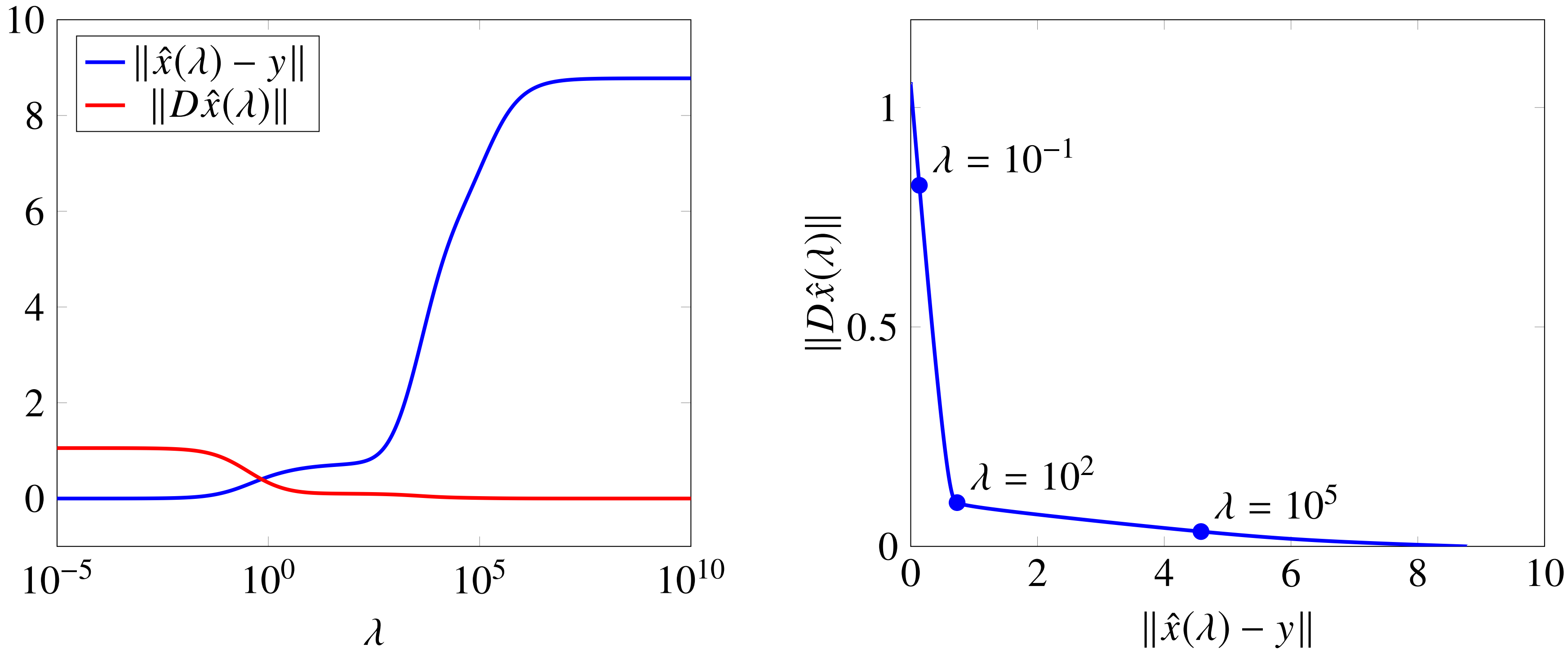
$$D = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 1 \end{bmatrix}$$

- equivalent to linear equation

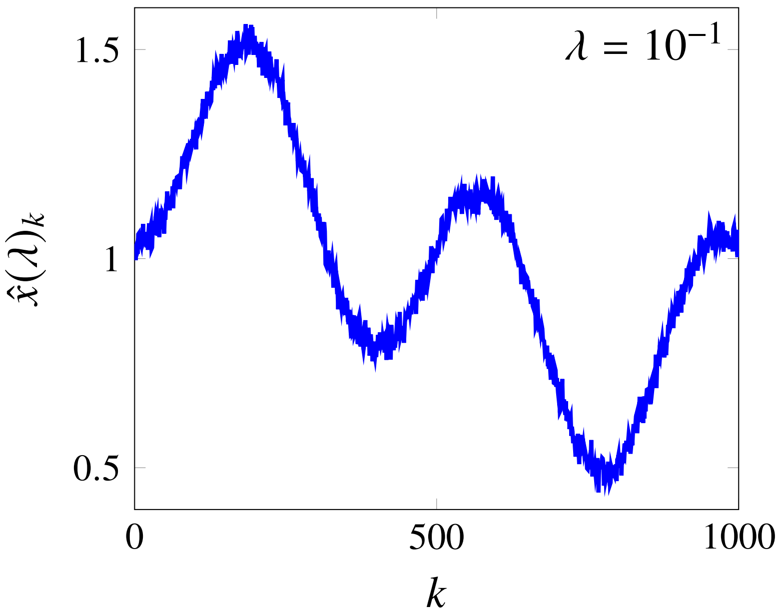
$$(I + \lambda D^T D)x = y$$

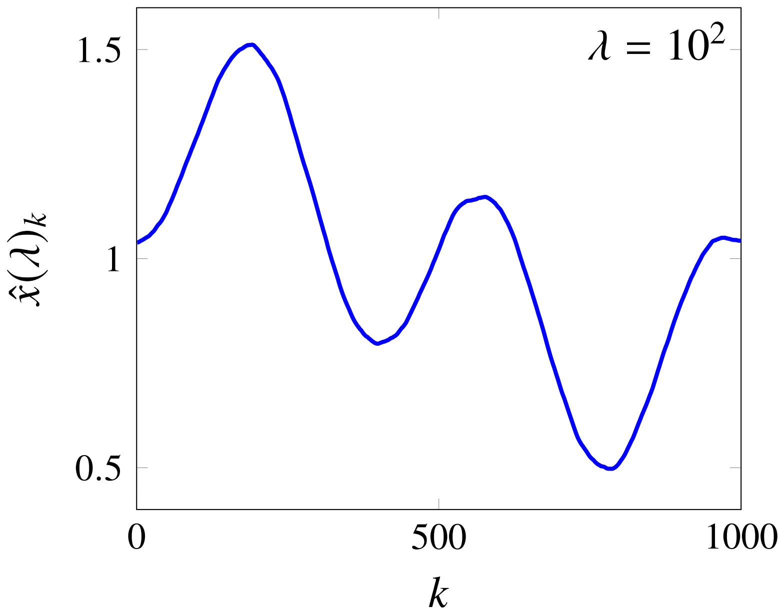
Trade-off

the two objectives $\|\hat{x}(\lambda) - y\|$ and $\|D\hat{x}(\lambda)\|$ for varying λ



Three solutions





- $\hat{x}(\lambda) \to y$ for $\lambda \to 0$
- $\hat{x}(\lambda) \to \mathbf{avg}(y)\mathbf{1}$ for $\lambda \to \infty$
- $\lambda \approx 10^2$ is good compromise

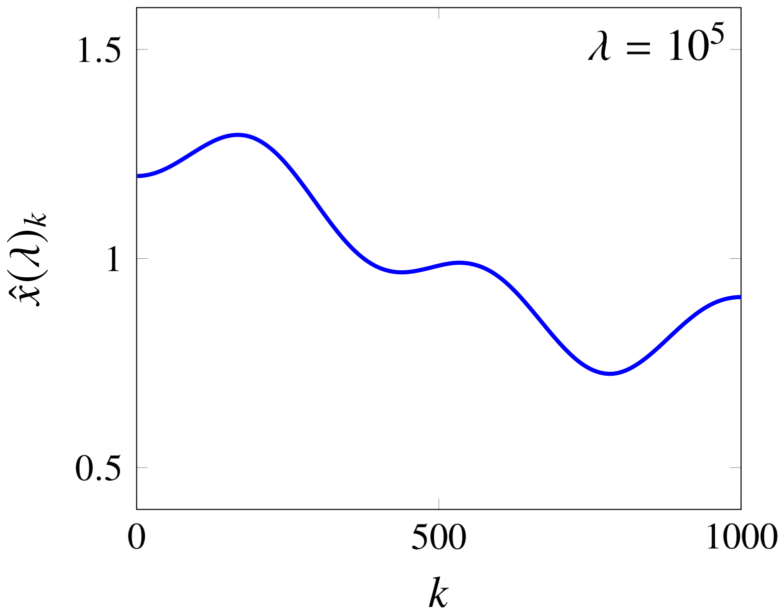


Image deblurring

$$y = Ax_{\text{ex}} + v$$

- x_{ex} is unknown image, y is observed image
- A is (known) blurring matrix, v is (unknown) noise
- images are $M \times N$, stored as MN-vectors

blurred, noisy image y

deblurred image $\hat{x}$

Least squares deblurring

$$\text{minimize} \quad \|Ax - y\|^2 + \lambda(\|D_\text{v}x\|^2 + \|D_\text{h}x\|^2)$$

- 1st term is *"data fidelity"* term: ensures $A\hat{x} \approx y$
- 2nd term penalizes differences between values at neighboring pixels

$$\|D_\text{h}x\|^2 + \|D_\text{v}x\|^2 = \sum_{i=1}^{M} \sum_{j=1}^{N-1} (X_{i,j+1} - X_{ij})^2 + \sum_{i=1}^{M-1} \sum_{j=1}^{N} (X_{i+1,j} - X_{ij})^2$$

if X is the $M \times N$ image stored in the MN-vector x

Differencing operations in matrix notation

suppose x is the $M \times N$ image X, stored column-wise as MN-vector

$$x = \left(X_{1:M,1}, X_{1:M,2}, \ldots, X_{1:M,N}\right)$$

- horizontal differencing: $(N-1) \times N$ block matrix with $M \times M$ blocks

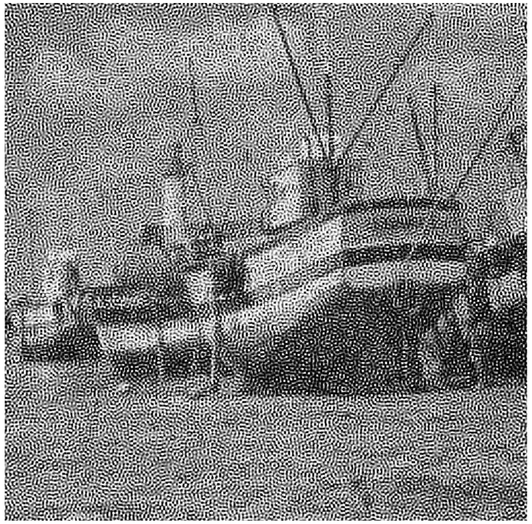
$$D_{\mathrm{h}} = \begin{bmatrix} -I & I & 0 & \cdots & 0 & 0 & 0 \\ 0 & -I & I & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & -I & I \end{bmatrix}$$

- vertical differencing: $N \times N$ block matrix with $(M-1) \times M$ blocks

$$D_{\mathrm{v}} = \begin{bmatrix} D & 0 & \cdots & 0 \\ 0 & D & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & D \end{bmatrix}, \qquad D = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 1 \end{bmatrix}$$

Deblurred images

$\lambda = 10^{-6}$



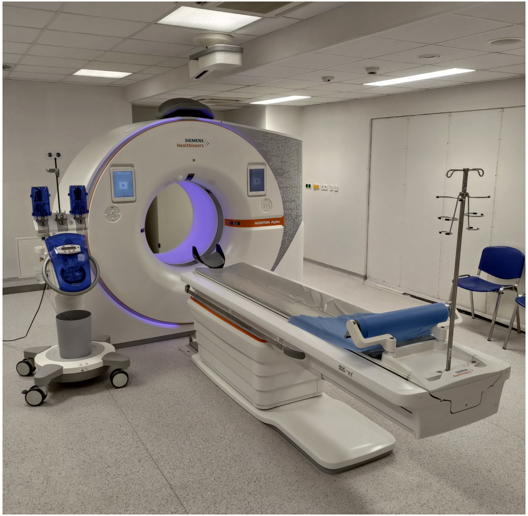
$\lambda = 10^{-4}$

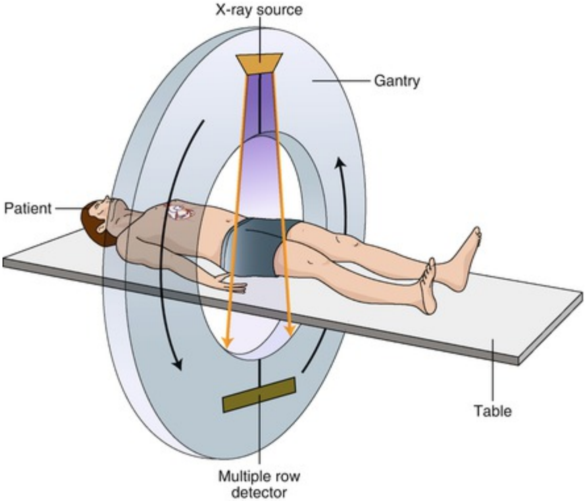
$\lambda = 10^{-2}$

$\lambda = 1$

Tomography

- goal is to reconstruct or estimate a function $d : \mathbf{R}^2 \to \mathbf{R}$ from (possibly noisy) line integral measurements
- d is often (but not always) some kind of density
- we'll focus on 2-D case, but it can be extended to 3-D
- used in medicine, manufacturing, networking, geology
- best known application: CAT (computer-aided tomography) scan


SIEMENS
Healthineers


X-ray source
Gantry
Patient
Table
Multiple row
detector

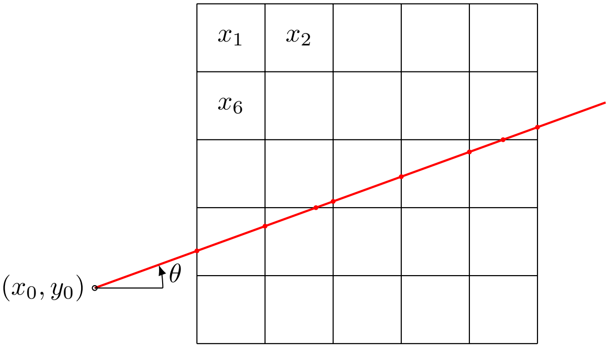
Discretization of d

- we d is constant on n pixels, numbered 1 to n
- represent (discretized) density function d by n-vector x
- x_i is value of d in pixel i
- line integral measurement y_i has form

$$y_i = \sum_{j=1}^{n} A_{ij} x_j + v_i$$

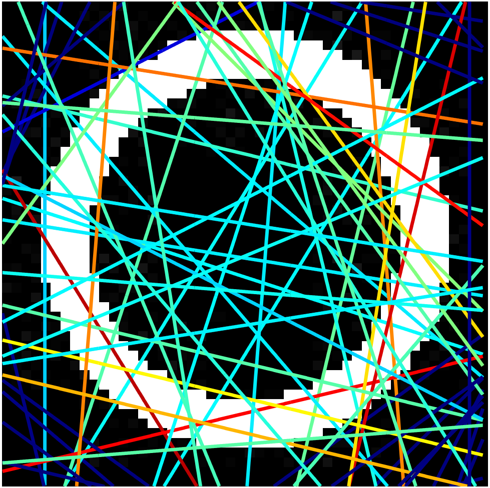
- A_{ij} is length of line ℓ_i in pixel j
- in matrix-vector form, we have $y = Ax + v$

Illustration

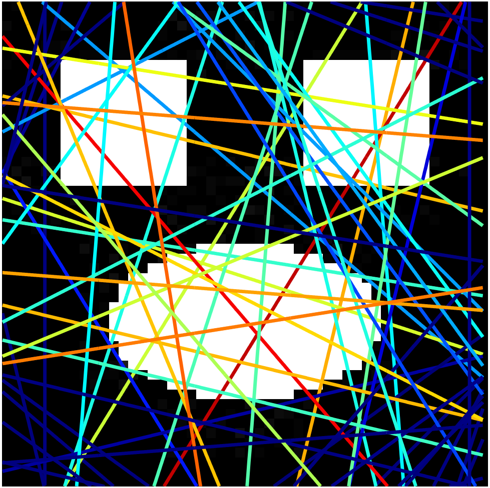


$$y = 1.06x_{16} + 0.80x_{17} + 0.27x_{12} + 1.06x_{13} + 1.06x_{14} + 0.53x_{15} + 0.54x_{10} + v$$

Example



Another example



Smoothness prior

- we assume that image is not too rough, as measured by (Laplacian)

$$\|D_{\mathrm{v}}x\|^2 + \|D_{\mathrm{h}}x\|^2$$

  - $D_h x$ gives first order difference in horizontal direction
  - $D_v x$ gives first order difference in vertical direction

- roughness measure is sum of squares of first order differences
- it is zero only when x is constant

Least-squares reconstruction

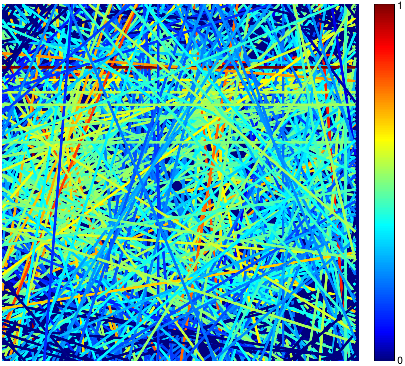
- choose $\hat{x}$ to minimize

$$\|Ax - y\|^2 + \lambda(\|D_\mathrm{v}\hat{x}\|^2 + \|D_\mathrm{h}\hat{x}\|^2)$$

  - first term is $\|v\|^2$, or deviation between what we observed (y) and what we would have observed without noise (Ax)
  - second term is roughness measure
- regularization parameter $\lambda > 0$ trades off measurement fit versus roughness of recovered image

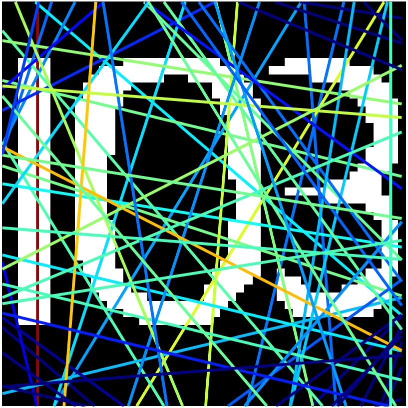
Example

- 50×50 pixels ($n = 2500$)
- 40 angles, 40 offsets ($m = 1600$ lines)
- 600 lines shown
- small measurement noise



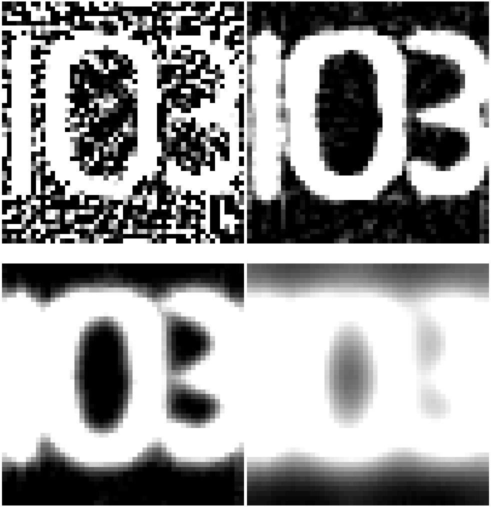
Reconstruction

reconstruction with $\lambda = 10$



Reconstruction

reconstructions with $\lambda = 10^{-6}, 20, 230, 2600$



Varying the number of line integrals

reconstruct with $m = 100, 400, 2500, 6400$ lines (with $\lambda = 10, 15, 25, 30$)