MANCHESTER 1824

The University of Manchester

M\crNA
Manchester Numerical Analysis

MATH36061
Convex Optimization

Martin Lotz

School of Mathematics
The University of Manchester

Manchester, September 27, 2016

Outline

Outline

Organisation

Lecture	Tuesday 1-2 Thursday 9-10	ATB G.108 University Place 6.212
Tutorial	Thursday 10-11	University Place 6.212

Organisation

Lecture	Tuesday 1-2 Thursday 9-10	ATB G.108 University Place 6.212
Tutorial	Thursday 10-11	University Place 6.212

Course website: `http://www.math36061.org`

Organisation

Lecture	Tuesday 1-2 Thursday 9-10	ATB G.108 University Place 6.212
Tutorial	Thursday 10-11	University Place 6.212

Course website: `http://www.math36061.org`

- Tutorial class in week one will consist of an introduction to Python, Jupyter and SageMathCloud.

Organisation

Lecture	Tuesday 1-2 Thursday 9-10	ATB G.108 University Place 6.212
Tutorial	Thursday 10-11	University Place 6.212

Course website: `http://www.math36061.org`

- Tutorial class in week one will consist of an introduction to Python, Jupyter and SageMathCloud.
- Problem sets consist of two parts: problems to be worked on at home, and problems to be discussed in class.

Organisation

Lecture	Tuesday 1-2 Thursday 9-10	ATB G.108 University Place 6.212
Tutorial	Thursday 10-11	University Place 6.212

Course website: `http://www.math36061.org`

- Tutorial class in week one will consist of an introduction to Python, Jupyter and SageMathCloud.
- Problem sets consist of two parts: problems to be worked on at home, and problems to be discussed in class.
- All the material presented in the lecture is made available on the website and blackboard as the course progresses.

Organisation

Lecture	Tuesday 1-2 Thursday 9-10	ATB G.108 University Place 6.212
Tutorial	Thursday 10-11	University Place 6.212

Course website: `http://www.math36061.org`

- Tutorial class in week one will consist of an introduction to Python, Jupyter and SageMathCloud.
- Problem sets consist of two parts: problems to be worked on at home, and problems to be discussed in class.
- All the material presented in the lecture is made available on the website and blackboard as the course progresses.
- Contact: `martin.lotz@manchester.ac.uk`

Example classes

Example classes are there to deepen the understanding of the course material.

Example classes

Example classes are there to deepen the understanding of the course material.

- Problems sheets should ideally be looked at **before** the example class.

Example classes

Example classes are there to deepen the understanding of the course material.

- Problems sheets should ideally be looked at **before** the example class.
- In the example classes,
  - you will be given some time to work on the problems,
  - you should ask **questions** if some parts are not clear,
  - you will get **feedback** on your attempts at the problems,
  - we will go through a selection of problems and their solutions.

Python

Python

- Python is an indispensable tool for this course.

Python

- Python is an indispensable tool for this course.
- Python is one of the most popular programming languages, used among others by Google, Dropbox, NASA, etc.

Python

- Python is an indispensable tool for this course.
- Python is one of the most popular programming languages, used among others by Google, Dropbox, NASA, etc.
- Many Data science jobs require knowledge of Python.

Python

- Python is an indispensable tool for this course.
- Python is one of the most popular programming languages, used among others by Google, Dropbox, NASA, etc.
- Many Data science jobs require knowledge of Python.
- Availability for this course:
  - On SageMathCloud
  - By installing Anaconda Python on your computer

Python

- Python is an indispensable tool for this course.
- Python is one of the most popular programming languages, used among others by Google, Dropbox, NASA, etc.
- Many Data science jobs require knowledge of Python.
- Availability for this course:
  - On SageMathCloud
  - By installing Anaconda Python on your computer
- The course page contains links to Python documentation

Python

Example Python code for printing Fibonacci numbers below 25

```
def f(a,b):
    c = a + b
    return c

a, b = 0, 1
while b<25:
a, b = b, f(a,b)
print a
```

CVXPY

CVXPY

- CVXPY is a convex optimization package for Python.

CVXPY

- CVXPY is a convex optimization package for Python.
- CVXPY is freely available on

  `http://www.cvxpy.org/en/latest/`

CVXPY

- CVXPY is a convex optimization package for Python.
- CVXPY is freely available on

  `http://www.cvxpy.org/en/latest/`
- It is easy to install and the documentation provides some examples to get started. On SageMathCloud it is readily available.

CVXPY

- CVXPY is a convex optimization package for Python.
- CVXPY is freely available on

  `http://www.cvxpy.org/en/latest/`
- It is easy to install and the documentation provides some examples to get started. On SageMathCloud it is readily available.
- An overview of Python, Jupyter notebooks, and CVXPY is given during the first tutorial class on Thursday!

Mathematical prerequisites

The course will make extensive use of results from

- linear algebra (vectors and matrices in $\mathbb{R}^n$),
- multivariate calculus (gradients, Hessians).

A comprehensive overview of the mathematics needed can be found on the course website.

Outline

Mathematical optimization

A mathematical optimization problem is a problem of the form

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{x} \in \Omega \end{aligned}$$

where f is the objective function and $\Omega \subseteq \mathbb{R}^n$ is a set of constraints. Often the set Ω is defined by those $\boldsymbol{x} \in \mathbb{R}^n$ that satisfy some conditions

$$g_1(\boldsymbol{x}) \leq 0, \ldots, g_m(\boldsymbol{x}) \leq 0,$$

where the g_i are real-valued functions.
A vector $\boldsymbol{x}^*$ such that $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ for all other $\boldsymbol{x}$ satisfying the constraints is called a solution or minimizer of the problem.

Example: least squares

Assume a quantity Y depends linearly on predictors $X_1, \ldots, X_n$:

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon, \tag{3.1}$$

where ε is some random error.

Example: least squares

Assume a quantity Y depends linearly on predictors $X_1, \ldots, X_n$:

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon, \tag{3.1}$$

where ε is some random error.

- Example: Y could sales, and $X_1, \ldots, X_p$ advertisement budget in different media (TV, internet, radio, billboards, newspapers).

Example: least squares

Assume a quantity Y depends linearly on predictors $X_1, \ldots, X_n$:

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon, \tag{3.1}$$

where ε is some random error.

- Example: Y could sales, and $X_1, \ldots, X_p$ advertisement budget in different media (TV, internet, radio, billboards, newspapers).
- Goal: determine model parameters $\beta_0, \ldots, \beta_p$.

Example: least squares

Assume a quantity Y depends linearly on predictors $X_1, \ldots, X_n$:

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon, \tag{3.1}$$

where ε is some random error.

- Example: Y could sales, and $X_1, \ldots, X_p$ advertisement budget in different media (TV, internet, radio, billboards, newspapers).
- Goal: determine model parameters $\beta_0, \ldots, \beta_p$.
- What we know is data from $n \geq p$ observations or experiments:

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i, \quad 1 \leq i \leq n.$$

Example: least squares

What we know is data from $n \geq p$ observations or experiments:

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i, \quad 1 \leq i \leq n.$$

Example: least squares

What we know is data from $n \geq p$ observations or experiments:

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i, \quad 1 \leq i \leq n.$$

Collect the data in matrices and vectors:

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \boldsymbol{X} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ \vdots & \ddots & \vdots & \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix}, \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}, \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

Example: least squares

What we know is data from $n \geq p$ observations or experiments:

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i, \quad 1 \leq i \leq n.$$

Collect the data in matrices and vectors:

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \boldsymbol{X} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ \vdots & \ddots & \vdots & \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix}, \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}, \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

The relationship can then be written as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}.$$

Example: least squares

Based on the relationship

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

choose $\boldsymbol{\beta}$ that minimizes the squared 2-norm of the error,

$$\text{minimize} \quad \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2$$

Example: least squares

Based on the relationship

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

choose $\boldsymbol{\beta}$ that minimizes the squared 2-norm of the error,

$$\text{minimize} \quad \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2$$

- This is an optimization problem with a quadratic objective function and no constraints.

Example: least squares

Based on the relationship

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

choose $\boldsymbol{\beta}$ that minimizes the squared 2-norm of the error,

$$\text{minimize} \quad \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2$$

- This is an optimization problem with a quadratic objective function and no constraints.
- The problem has a closed form solution

$$\boldsymbol{\beta} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}.$$

Example: least squares

Based on the relationship

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

choose $\boldsymbol{\beta}$ that minimizes the squared 2-norm of the error,

$$\text{minimize} \quad \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2$$

- This is an optimization problem with a quadratic objective function and no constraints.
- The problem has a closed form solution

$$\boldsymbol{\beta} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}.$$

- There are more efficient methods available than using the closed form solution.

Application: linear regression

Observation: the log of the basal metabolic rate (energy expenditure) rate Y in mammals is related to log adult mass X as

$$Y = \beta_0 + \beta_1 X.$$

Application: linear regression

Observation: the log of the basal metabolic rate (energy expenditure) rate Y in mammals is related to log adult mass X as

$$Y = \beta_0 + \beta_1 X.$$

- Collect data from a database 573 mammal species and plot the relationship.

Application: linear regression

Observation: the log of the basal metabolic rate (energy expenditure) rate Y in mammals is related to log adult mass X as

$$Y = \beta_0 + \beta_1 X.$$

- Collect data from a database 573 mammal species and plot the relationship.
- Assemble the vector $\boldsymbol{y}$, matrix $\boldsymbol{X}$, and solve the problem

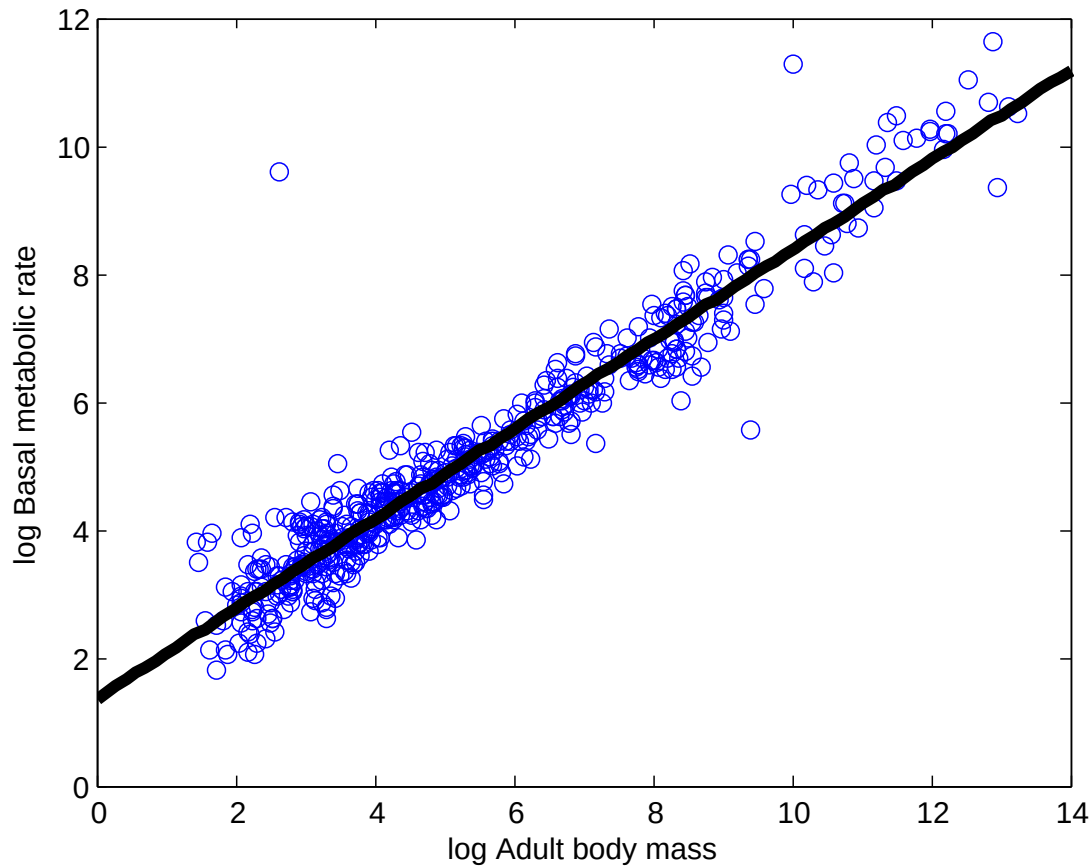
$$\text{minimize} \quad \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2$$

Application: linear regression

```
# Create a p+1 vector of variables
Beta = Variable(p+1)

# Create sum-of-squares objective function
objective=Minimize(sum_entries(square(X*Beta - y)))

# Create problem and solve it
prob = Problem(objective)
prob.solve()

plt.plot(bmr[:,0],bmr[:,1],'o')

xx = np.linspace(0,14,100)
bmr = plt.plot(xx, Beta[0].value+Beta[1].value*xx,\
   color='red', linewidth=2)
plt.show()
```

Application: linear regression

$$Y = \beta_0 + \beta_1 X.$$



Application: linear regression

$$Y = \beta_0 + \beta_1 X.$$

Figure: Episode Size Matters from TV series Wonders of Life

Example: linear programming

Linear programming refers to problems of the form

$$\begin{aligned} \text{maximize} \quad & c_1 x_1 + \cdots + c_n x_n \\ \text{subject to} \quad & a_{11} x_1 + \cdots + a_{1n} x_n \leq b_1 \\ & \cdots \\ & a_{m1} x_1 + \cdots + a_{mn} x_n \leq b_m \\ & x_1 \geq 0, \ldots, x_n \geq 0. \end{aligned}$$

Example: linear programming

Linear programming refers to problems of the form

$$\begin{aligned} \text{maximize} \quad & c_1 x_1 + \cdots + c_n x_n \\ \text{subject to} \quad & a_{11} x_1 + \cdots + a_{1n} x_n \leq b_1 \\ & \cdots \\ & a_{m1} x_1 + \cdots + a_{mn} x_n \leq b_m \\ & x_1 \geq 0, \ldots, x_n \geq 0. \end{aligned}$$

- Can be rewritten in standard form shown at the beginning!

Example: linear programming

Linear programming refers to problems of the form

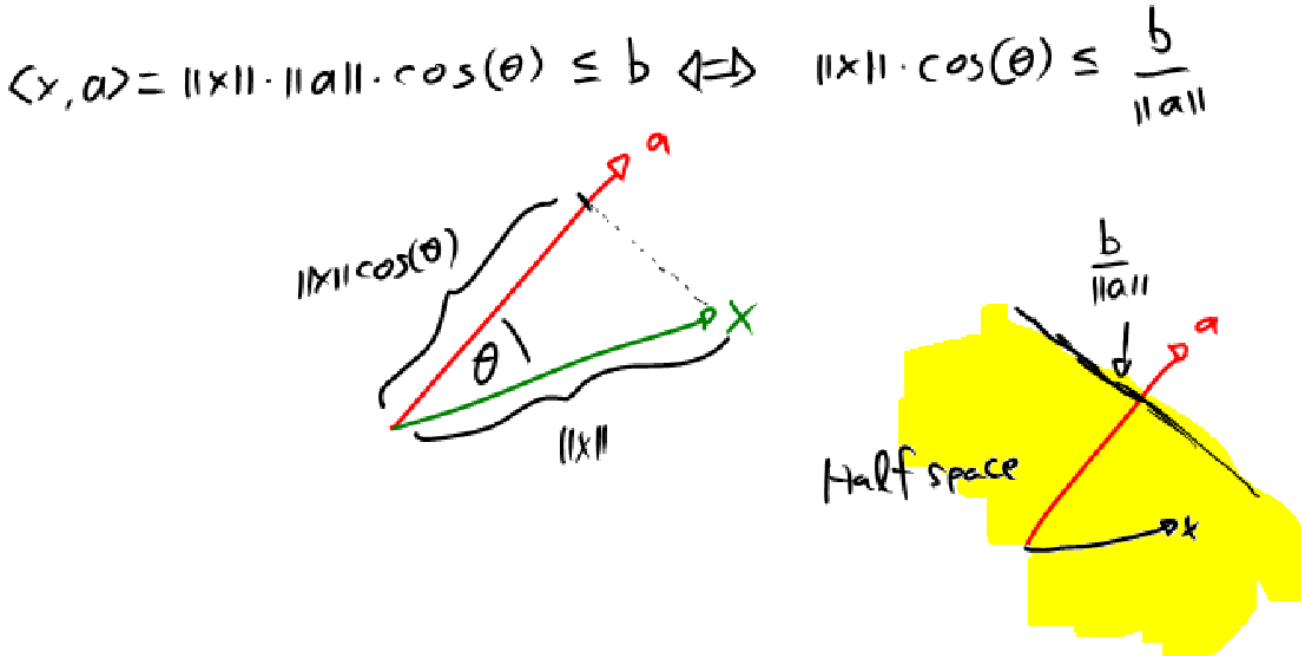
$$\begin{aligned} \text{maximize} \quad & c_1 x_1 + \cdots + c_n x_n \\ \text{subject to} \quad & a_{11} x_1 + \cdots + a_{1n} x_n \leq b_1 \\ & \cdots \\ & a_{m1} x_1 + \cdots + a_{mn} x_n \leq b_m \\ & x_1 \geq 0, \ldots, x_n \geq 0. \end{aligned}$$

- Can be rewritten in standard form shown at the beginning!
- Using matrices and vectors:

$$\begin{aligned} \text{maximize} \quad & \langle \boldsymbol{c}, \ \boldsymbol{x} \rangle \, (= \boldsymbol{c}^\top \boldsymbol{x}) \\ \text{subject to} \quad & A\boldsymbol{x} \leq \boldsymbol{b} \quad \text{(inequality componentwise)} \\ & \boldsymbol{x} \geq 0 \end{aligned}$$

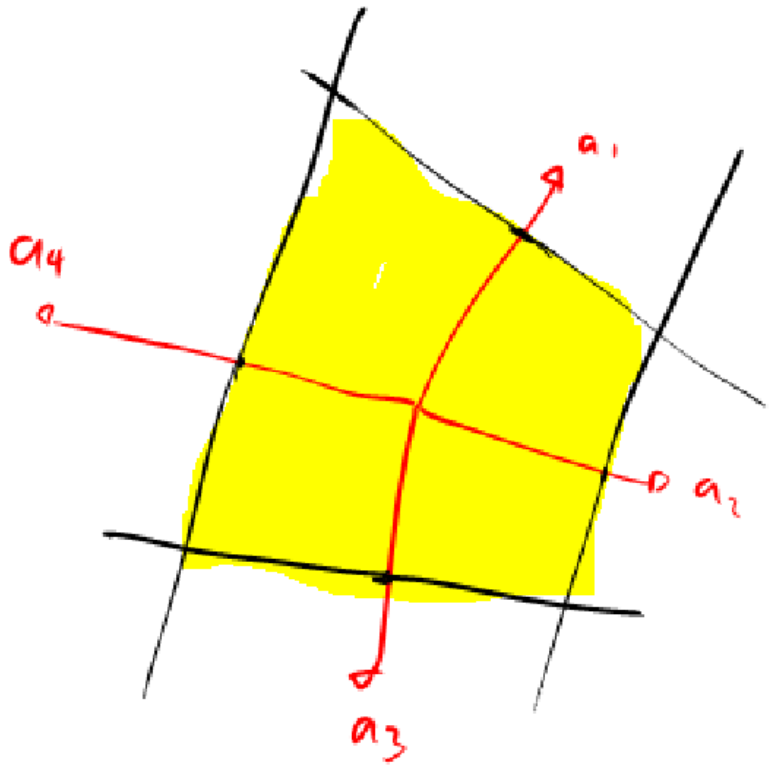
Linear programming: geometry

$$\{\boldsymbol{x} : \langle \boldsymbol{a},\ \boldsymbol{x} \rangle \leq b\}.$$



Linear programming: geometry

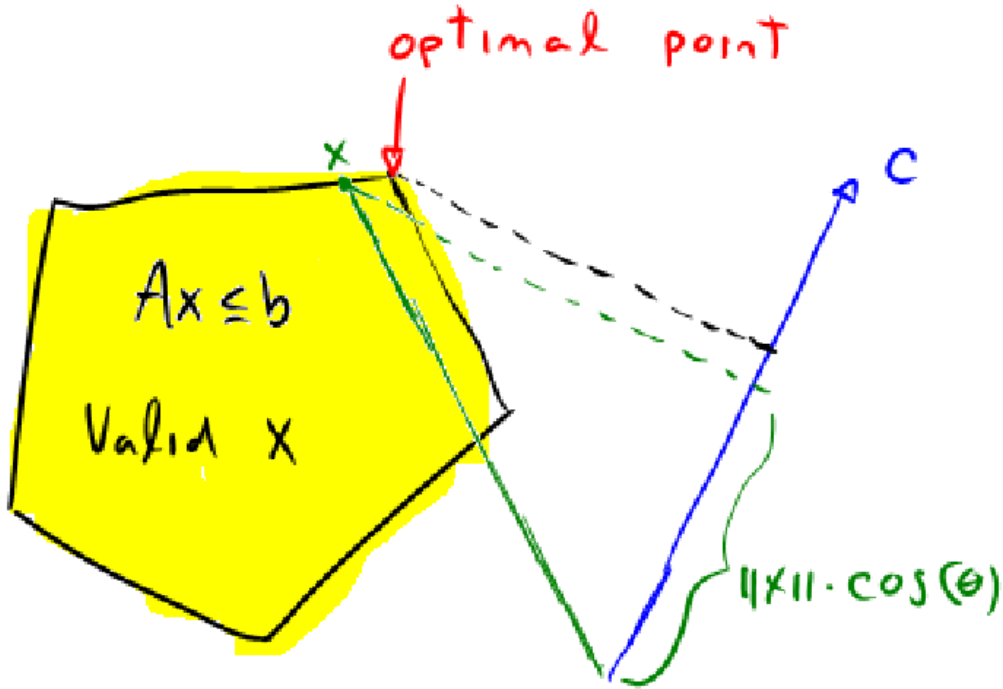
$$\{\boldsymbol{x} : \langle \boldsymbol{a}_1,\ \boldsymbol{x}\rangle \le b_1, \ldots, \langle \boldsymbol{a}_m,\ \boldsymbol{x}\rangle \le b_m\} = \{\boldsymbol{x} : \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}\}.$$



Polyhedron

Linear programming: geometry

maximize $\langle \boldsymbol{c},\ \boldsymbol{x} \rangle$ subject to $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}$.



x s.t. $\langle x, c \rangle = \|x\| \cdot \|c\| \cdot \cos(\theta)$ max.

Linear programming: application

A cargo plane has two compartments with capacities $C_1 = 35$ and $C_2 = 40$ tonnes and volumes $V_1 = 250$ and $V_2 = 400$ cubic metres.

Linear programming: application

A cargo plane has two compartments with capacities $C_1 = 35$ and $C_2 = 40$ tonnes and volumes $V_1 = 250$ and $V_2 = 400$ cubic metres.

- Three types of cargo:

	Volume	Weight	Profit (£/ tonne)
Cargo 1	8	25	£300
Cargo 2	10	32	£350
Cargo 3	7	28	£270

Linear programming: application

A cargo plane has two compartments with capacities $C_1 = 35$ and $C_2 = 40$ tonnes and volumes $V_1 = 250$ and $V_2 = 400$ cubic metres.

- Three types of cargo:

	Volume	Weight	Profit (£/ tonne)
Cargo 1	8	25	£300
Cargo 2	10	32	£350
Cargo 3	7	28	£270

- How to distribute the cargo to maximize profit?

Linear programming: application

x_{ij}: amount of cargo i in compartment j

Objective function:

$$f(\boldsymbol{x}) = 300 \cdot (x_{11} + x_{12}) + 350 \cdot (x_{21} + x_{22}) + 270 \cdot (x_{31} + x_{32}).$$

Linear programming: application

x_{ij}: amount of cargo i in compartment j

Objective function:

$$f(\boldsymbol{x}) = 300 \cdot (x_{11} + x_{12}) + 350 \cdot (x_{21} + x_{22}) + 270 \cdot (x_{31} + x_{32}).$$

Constraints:

$$\begin{aligned}
x_{11} + x_{12} &\leq 25 && \text{(total amount of cargo 1)} \\
x_{21} + x_{22} &\leq 32 && \text{(total amount of cargo 2)} \\
x_{31} + x_{32} &\leq 28 && \text{(total amount of cargo 3)} \\
x_{11} + x_{21} + x_{31} &\leq 35 && \text{(weight constraint on comp. 1)} \\
x_{12} + x_{22} + x_{32} &\leq 40 && \text{(weight constraint on comp. 2)} \\
8x_{11} + 10x_{21} + 7x_{31} &\leq 250 && \text{(volume constraint on comp. 1)} \\
8x_{12} + 10x_{22} + 7x_{32} &\leq 400 && \text{(volume constraint on comp. 2)} \\
(x_{11} + x_{21} + x_{31})/35 - (x_{12} + x_{22} + x_{32})/40 &= 0 && \text{(maintain balance of weight ratio)} \\
x_{ij} &\geq 0 && \text{(cargo can't have negative weight)}
\end{aligned}$$

Linear programming: application

As complicated as it looks, it can be solved easily with CVXPY.

```
 # Define all the matrices and vectors involved
c = np.array([300,300,350,350,270,270])
A = np.array([[1, 1, 0, 0, 0, 0],
              [0, 0, 1, 1, 0, 0],
              [0, 0, 0, 0, 1, 1],
              [1, 0, 1, 0, 1, 0],
              [0, 1, 0, 1, 0, 1],
              [8, 0, 10, 0, 7, 0],
              [0, 8, 0, 10, 0, 7]])
b = np.array([25,32,28,35,40,250,400]);
B = np.array([1/35, -1/40, 1/35, -1/40, 1/35, -1/40]);
d = np.zeros(1)

# Create variables, objective and constraints
x = Variable(6)
constraints = [A*x <= b, B*x == d, x >= 0]
objective = Maximize(c*x)

# Create a problem using the objective and constraints and solve it
prob = Problem(objective, constraints)
prob.solve()
```

Linear programming: application

As complicated as it looks, it can be solved easily with CVXPY.

```
 # Define all the matrices and vectors involved
c = np.array([300,300,350,350,270,270])
A = np.array([[1, 1, 0, 0, 0, 0],
              [0, 0, 1, 1, 0, 0],
              [0, 0, 0, 0, 1, 1],
              [1, 0, 1, 0, 1, 0],
              [0, 1, 0, 1, 0, 1],
              [8, 0, 10, 0, 7, 0],
              [0, 8, 0, 10, 0, 7]])
b = np.array([25,32,28,35,40,250,400]);
B = np.array([1/35, -1/40, 1/35, -1/40, 1/35, -1/40]);
d = np.zeros(1)

# Create variables, objective and constraints
x = Variable(6)
constraints = [A*x <= b, B*x == d, x >= 0]
objective = Maximize(c*x)

# Create a problem using the objective and constraints and solve it
prob = Problem(objective, constraints)
prob.solve()
```

$$x_{11} = 6.7500, x_{12} = 7.7143, x_{21} = 0, x_{22} = 32, x_{31} = 28, x_{32} = 0.$$

Example: total variation inpainting

Optimization methods play an increasingly important role in image and signal processing.

Example: total variation inpainting

Optimization methods play an increasingly important role in image and signal processing.

- An image is an $m \times n$ matrix $\boldsymbol{U}$, with each entry u_{ij} corresponding to a light intensity or colour.

Example: total variation inpainting

Optimization methods play an increasingly important role in image and signal processing.

- An image is an $m \times n$ matrix $\boldsymbol{U}$, with each entry u_{ij} corresponding to a light intensity or colour.
- In the image inpainting problem, one aims to *guess* the true value of missing or corrupted entries of an image.

Example: total variation inpainting

Optimization methods play an increasingly important role in image and signal processing.

- An image is an $m \times n$ matrix $\boldsymbol{U}$, with each entry u_{ij} corresponding to a light intensity or colour.
- In the image inpainting problem, one aims to *guess* the true value of missing or corrupted entries of an image.

A conceptually simple approach is to replace the image with the *closest* image among a set of images satisfying typical properties.

Example: total variation inpainting

What are typical properties of a typical image?

- Images tend to have large homogeneous areas in which the colour doesn't change much;
- Images have approximately low rank, when interpreted as matrices.

The total variation or TV-norm is the sum of the norm of the horizontal and vertical differences,

$$\|\boldsymbol{U}\|_{\mathrm{TV}} = \sum_{i=1}^{m}\sum_{j=1}^{n}\sqrt{(u_{i+1,j}-u_{i,j})^2+(u_{i,j+1}-u_{i,j})^2},$$

Example: total variation inpainting

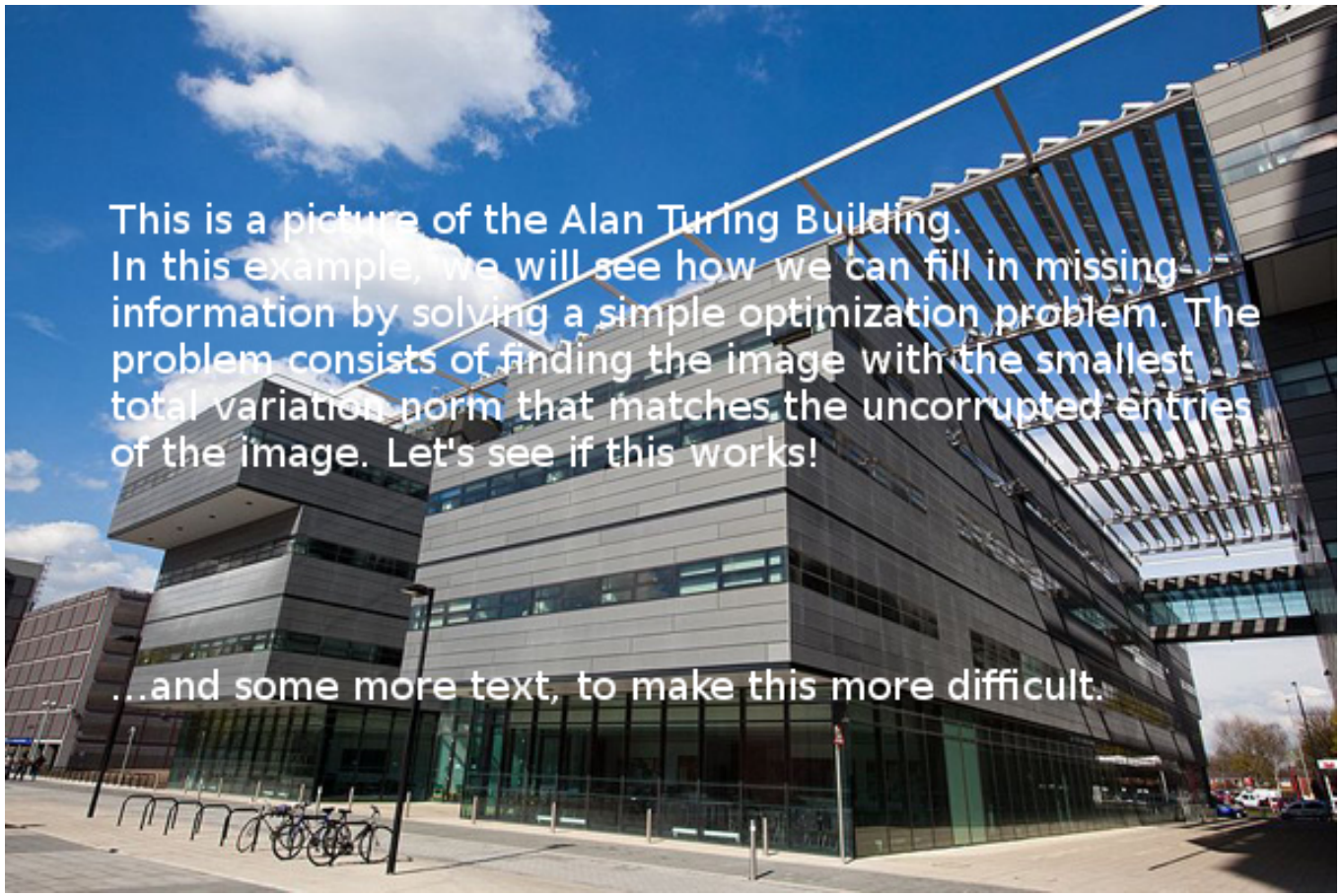
The TV-norm naturally increases with increased variation or sharp edges in an image.

$$\boldsymbol{U}_1 = \begin{pmatrix} 0 & 17 & 3 \\ 7 & 32 & 0 \\ 2 & 9 & 27 \end{pmatrix}, \quad \boldsymbol{U}_2 \begin{pmatrix} 1 & 1 & 3 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \end{pmatrix}$$

- The left matrix has TV-norm $\|\boldsymbol{U}_1\|_{\mathrm{TV}} = 200.637$, while the right one has TV-norm $\|\boldsymbol{U}_2\|_{\mathrm{TV}} = 14.721$.
- Intuitively, we would expect a natural image with artifacts added to it to have a higher TV norm.

Example: total variation inpainting

Consider the following corrupted image.



Example: total variation inpainting

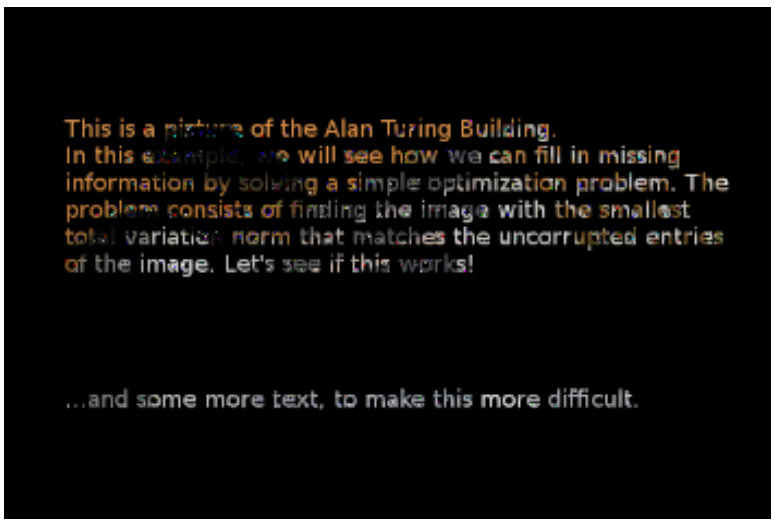
Applying the optimization problem

$$\text{minimize} \quad \|\boldsymbol{X}\|_{\text{TV}} \quad \text{subject to} \quad x_{ij} = u_{ij}, \ (i,j) \in \Omega.$$

we recover an approximation to the true image:

Inpainted Image

Difference Image



Example: machine learning

In machine learning, the goal is to automatically learn a function

$$f(\boldsymbol{x}) = \boldsymbol{y}$$

from a training set of pairs $(\boldsymbol{x}_i, \boldsymbol{y}_i)$.

Example: machine learning

In machine learning, the goal is to automatically learn a function

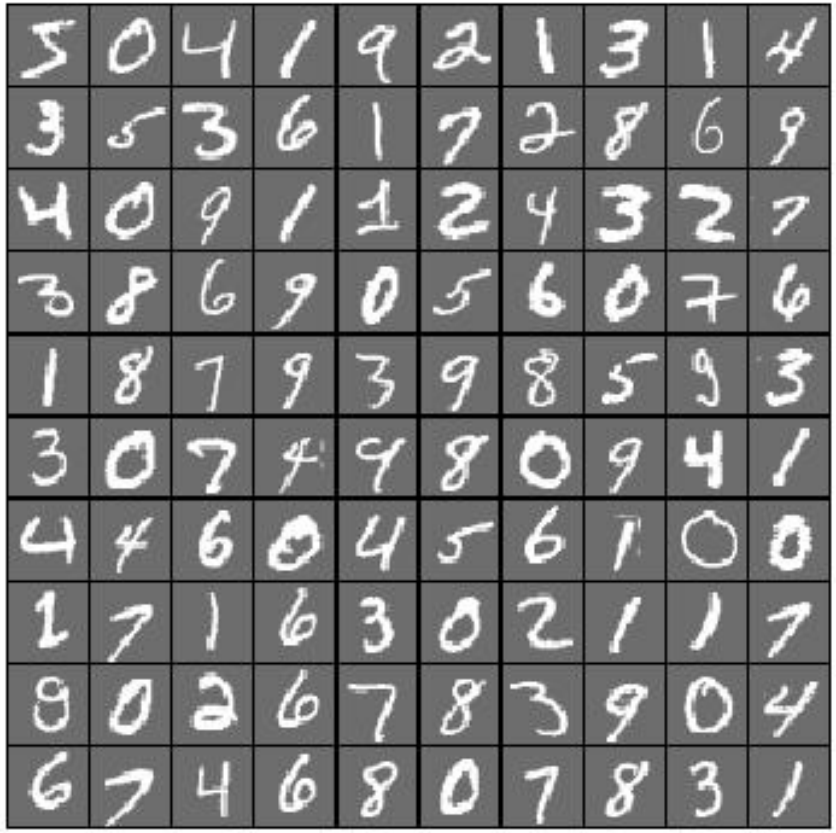
$$f(\boldsymbol{x}) = \boldsymbol{y}$$

from a training set of pairs $(\boldsymbol{x}_i, \boldsymbol{y}_i)$.
The problem can be posed as an optimization problem

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{m} (f(\boldsymbol{x}_i) - \boldsymbol{y}_i)^2,$$

where $\mathcal{F}$ is a class of functions. Examples include regression and classification.

Application: letter recognition



- By solving an optimization problem (a multi-layer neural network), the computer learns a function that assigns to an image the closest handwritten letter.

Applications of machine learning

- Handwritten character recognition;
- Classification of gene expression data;
- Computer games;
- Online recommender systems;
- Sentiment analysis, text mining;
- Medical diagnosis;
- Computer vision;
- ...

Developments driven by Google DeepMind, Facebook AI, OpenAI, etc.

Outline

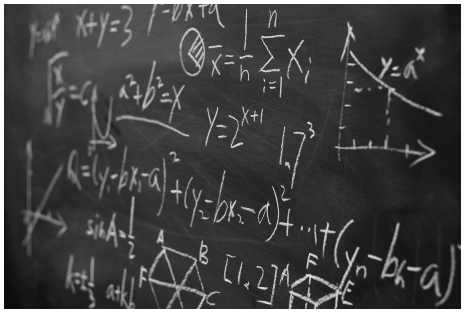
Overview of the course

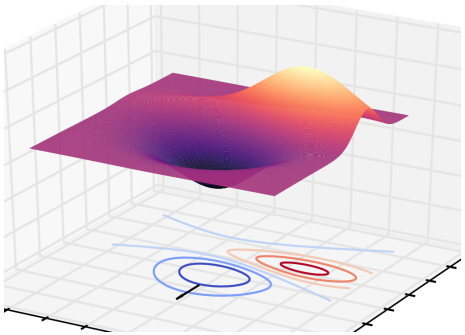
The course will try to balance three points of view:

- Theory. Optimality conditions, duality theory, convex analysis, geometry of linear, quadratic and semidefinite programming.
- Algorithms. Descent algorithms, interior-point methods, projected gradient methods, Lagrangian
- Applications. Logistics and scheduling, finance, signal processing, convex regularization, semidefinite relaxations of hard combinatorial problems, machine learning, compressive sensing.

The course will be...

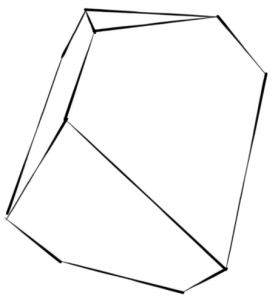
Rigorous. We prove Theorems and the occasional Lemma.





Computational. We implement examples and see optimization at work.

Visual. Convex optimization is a geometric theory, so expect to see lots of pictures.



Overview of the course

Next class (Thursday 9-11, Univ. Place 6.212):

- Unconstrained optimization: gradient descent;
- Introduction to the computational tools of this course.

See you next time!

Caged parrot

Free parrot

minimize $\|\boldsymbol{X}\|_{\mathrm{TV}}$ subject to $x_{ij} = u_{ij},\ (i,j) \in \Omega.$