ENGSCI 721

INVERSE PROBLEMS

Oliver Maclaren
oliver.maclaren@auckland.ac.nz

MODULE OVERVIEW

Inverse Problems (*Oliver Maclaren*) [~8 lectures/2 tutorials]

1. *Basic concepts* [4 lectures]
   Forward vs inverse problems. Well-posed vs ill-posed problems. Algebra of inverse problems (matrix calculus, generalised inverses etc). Regularisation and trade-offs.
2. *More regularisation* [3 lectures]
   Higher-order Tikhonov regularisation, truncated singular value decompositions, iterative regularisation.

MODULE OVERVIEW

3. *Preview of the statistical view of inverse problems*
   [1 lectures] Bayesians, Frequentists and all that. Basic frequentist analysis.

LECTURE 3: INVERSES II

Topics:

- Tall and wide systems continued
- Least norm solutions for wide systems
- Types of algebraic inverses (left, right, generalised)
- Solving tall/wide systems with left/right inverses
- Unifying types of inverses with generalised inverses
- Why generalised inverses aren't enough!

EngSci 721 : Lecture 3 Inverses II

Resolving lack of existence (yesterday)
&/or $\boxed{\text{uniqueness}}$ (today)

↳ Formulating & solving as optimisation problems

↳ least squares & least norm solutions

- Algebraic characterisation of inverses & resolution operators
- Unification via generalised inverses
- Two step vs simultaneous optimisation
  → or, why the generalised inverse also needs regularisation

---

Recall

Algebra & Calculus of Inverse Problems

Generalised Inverses

Our basic problem can be defined as:

> 'solve', ie 'invert', equations like $F(x) = y$ for x, given y

where:

- x & y could be vectors, functions, images etc
- solutions might not exist, might not be unique &/or might not be stable

---

Recall

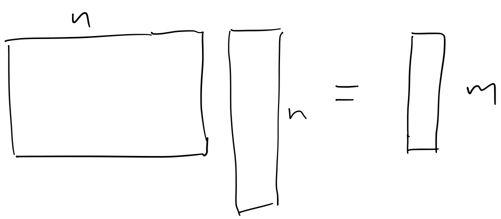
Linear setting

$$\mathbb{R}^n \underset{?}{\overset{A}{\rightleftarrows}} \mathbb{R}^m$$

Consider the system of equations

$$Ax = y \quad \begin{cases} - A \text{ is } m \times n \text{ matrix} \\ - x \in \mathbb{R}^n \text{ vector} \\ - y \in \mathbb{R}^m \text{ vector} \end{cases}$$

eg



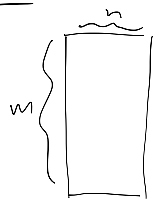
rows : eq^{n}s

cols : unknowns

How do we solve when $m \neq n$?

→ $m > n$, more rows than cols.

existence ?



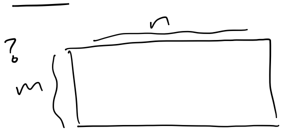
→ eq^{n}s > unknowns

→ possibly inconsistent / overdeterm.

→ 'more data than parameters'

→ $m < n$, more cols than rows

uniqueness ?



→ unknowns > eq^{n}s

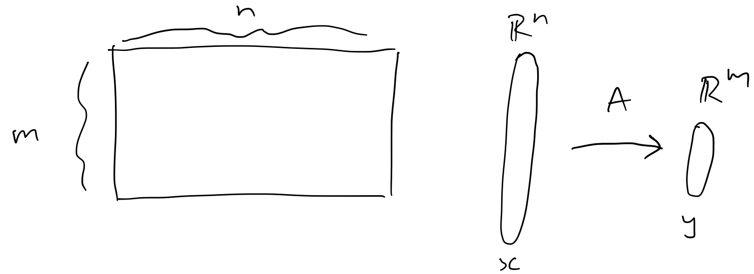
→ possibly many sol^{n}s

→ 'more param. than data'

Tools developed to 'solve' each case : cont'd

Case 2 : possibly non-unique : more unknowns than equations

Consider $Ax = y$, A is $m \times n$ & $m < n$



→ Assume w.l.o.g. that all m rows are linearly independent

→ Assume $Ax = y$ has at least one solution, ie $y \in R(A)$ (range / image of A)

General solution of $Ax = y$:

$$\boxed{x = x^{+} + x_0}$$

where $x_0 \in N(A)$ & x^{+} is any sol^n of $Ax = y$

(Recall: Nullspace: $N(A) = \{ x \mid Ax = 0 \}$)

2. Underdetermined equations: least norm sol^{n}s

- If we want to 'pick out' a single sol^n (we don't always!), a 'natural' choice is to choose the 'smallest' (think: 'simplest' or 'most efficient' sol^n)

- Again, this is relative to a particular norm, eg $\|\cdot\|_2 = l_2$ norm.

$\Rightarrow$ Problem to solve:

$$\min_{x} \|x\| \quad \text{st.} \quad Ax = y$$

note: here assuming exactly solvable

Equivalent to:

$$\min_{x} \|x\|^2 \quad \text{st.} \quad Ax = y$$

'least squares sol^n' (cf least squares approximation)

Least norm problem ('Model reduction')

Using duality or lagrange multipliers (see eg →) you can show that the least-squares solution to the minimum norm problem requires solving the dual problem

$$\boxed{\begin{array}{l} AA^T v = -2y \quad \text{for } v^* \\ \text{\& then obtaining } x^* = -\frac{1}{2}A^T v^* \end{array}}$$

→ Since we assume the m rows of A are linearly independent then AA^T is invertible (see handouts) & so get unique minimum norm

$$\boxed{x^* = A^T(AA^T)^{-1}y}$$

(we will look at what happens if not LI soon!).

[sol^n sketch : Lagrange multipliers (beyond scope...?)

- $d_x\left(x^T x + \lambda^T (Ax - y) \right)$

$$= 2x^T + \lambda^T A = 0 \quad (1)$$

- $d_\lambda\left(x^T x + \lambda^T (Ax-y)\right) = d_\lambda\left(\lambda^T (Ax-y)\right)$

$$= d_\lambda\left((Ax-y)^T \lambda\right)$$

$$= (Ax-y)^T = 0^T \iff Ax - y = 0 \quad (2)$$

$$\lambda^T A = -2x^T \quad (1)$$

$$x = -\frac{1}{2} A^T \lambda$$

$$Ax = -\frac{1}{2} A A^T \lambda = y \quad (2)$$

$$\underline{AA^T \lambda = -2y}$$

$$\lambda = -2(AA^T)^{-1} y$$

$$x = -\frac{1}{2} A^T \left(-2(AA^T)^{-1} y\right)$$

$$= A^T (AA^T)^{-1} y$$]

Summary so far:

- over-determined : can find least squares approx.
- under-determined : can find least squares/norm sol^n

→ no proper 'full' inverse exists in each case, but each is either a left inverse or a right inverse

$$LA = I \quad \text{(left)}$$
$$AR = I \quad \text{(right)}$$

solve rectangular systems from 'one direction/side'.

Left inverses : Algebra

$\mathbb{R}^n \underset{L}{\overset{A}{\rightleftarrows}} \mathbb{R}^m$

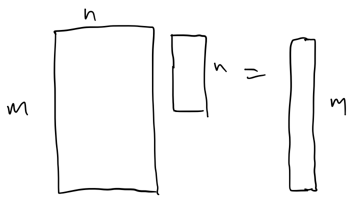
L is a left inverse (a retraction in category theory) for A if it satisfies

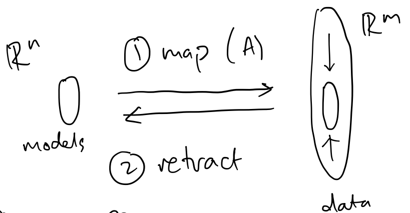
$$\boxed{LA = I} \quad \left\{ \begin{array}{l} A : m \times n \\ L : n \times m \\ I : n \times n \end{array} \right.$$

For least squares data approximation, we have

$$\boxed{L = (A^T A)^{-1} A^T} \quad \& \quad \boxed{x = Ly}$$

A left inverse exists when # rows $\geq$ # cols of A & the cols are LI (think 1-1 map into bigger space)





'shrink data to make solvable'

① then ② is identity on model space vs. ② then ① shrinks/projects $\mathbb{R}^m$ data to subspace of $\mathbb{R}^m$

Right inverses : Algebra

$\mathbb{R}^n \underset{R}{\overset{A}{\rightleftarrows}} \mathbb{R}^m$

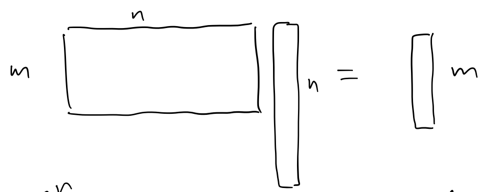
R is a right inverse (a section in category theory) for A if it satisfies

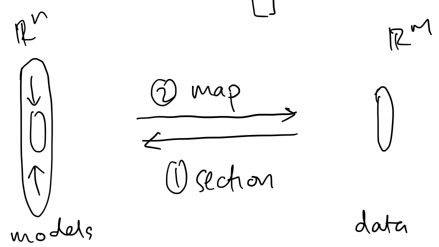
$$\boxed{AR = I} \quad \left\{ \begin{array}{l} A : m \times n \\ R : n \times m \\ I : m \times m \end{array} \right.$$

For least norm problems, we have

$$\boxed{R = A^T (A A^T)^{-1}} \quad \& \quad \boxed{x^* = Ry}$$

A right inverse exists when # cols $\geq$ # rows of A & the rows are LI





'shrink models to make unique'

② then ① 'shrinks'/projects models to subspace while ① then ② is identity on data space

Unification: generalised inverses

→ We can unify the solution of these problems with 'generalised inverses'

→ This will solve both existence & uniquess issues at the same time!
 ∟ spoiler alert: but not stability issues!

! We will see an alternative way to combine/unify the solutions when we discuss regularisation!)

The (actually, a - see later) generalised inverse can be charaterised as solving the minimum norm approximation problem,

⇒ ie it solves

$$\boxed{\begin{array}{l} \min \| y - Ax \| \\ \text{AND} \quad \min \| x \| \end{array}}$$

(in a particular way)

Unification: generalised inverses

- Generalised (or pseudo, for the special cases we look at) inverses solve the two step optimisation prob:

- Stage 1: $\underset{x}{\text{minimise}} \; \| y - Ax \|$ or $\| y - Ax \|^2$

Then

- Stage 2: minimise $\| x \|$ or $\| x \|^2$ among all solutions to stage 1.

The result is a matrix A^+ ← dagger symbol that

→ is a left inverse if one exists

→ is a right inverse if one exists

→ 'close to' left/right inverse if neither exist.

Generalised vs pseudo-inverses

The most general algebraic characterisation of a generalised inverse A^+ is just

(1) $\boxed{AA^+A = A}$

Note: • suppose A^TA is invertible

$\Rightarrow$ least sq. left inverse $L = (A^TA)^{-1}A^T$ exists

$A^T \cdot (1) \rightarrow A^TAA^+A = A^TA$

$$A^+A = (A^TA)^{-1}A^TA$$
$$= (M)^{-1}M$$
$$= I \quad \} \; A^+ \text{ is a left inverse too}$$

• now suppose AA^T is invertible

$\Rightarrow$ least norm right inverse $R = A^T(AA^T)^{-1}$ exists

$(1) \cdot A^T \rightarrow AA^+AA^T = AA^T$

$$AA^+ = (AA^T)(AA^T)^{-1}$$
$$= I \quad \} \; A^+ \text{ is a right inverse too}$$

Generalised vs pseudo-inverses

We don't have to take

- $A^+ = L = (A^TA)^{-1}A^T$ = least squares
- $A^+ = R = A^T(AA^T)^{-1}$ = least norm

in each case.

Generalised inverses defined via (1) are not unique

$\rightarrow$ add extra conditions

$\Rightarrow$ depends on goals

} eg different norms etc diff. inverses

Pseudo?

Eg what we've been using (least sq/norm) (& will assume unless otherwise) is the Moore-Penrose | Pseudo-inverse |

This satisfies

$$\boxed{\begin{aligned} AA^+A &= A \\ A^+AA^+ &= A^+ \\ (A^+A)^T &= A^+A \\ (AA^+)^T &= AA^+ \end{aligned}}$$

least squares, least norm generalised inverse = 'pseudo-inverse'

Model resolution, data resolution operators:

$\boxed{R_D = AA^+}$ } how much data is 'shrunk' or smeared

$\boxed{R_M = A^+A}$ } how much model is 'shrunk' or smeared

see below

Not I in gen. but something 'similar'

→ Note $I^2 = I$ ('idempotent')

Projection operators P characterised by

$\boxed{P^2 = P}$ ('idempotent')

→ one application of P gives 'maximum' effect

1. Suppose $A^+A = I$ but $AA^+ \neq I$ (left inverse only)

$$\Rightarrow R_D R_D = AA^+AA^+ = AA^+ = R_D$$

⇒ R_D is a projection on data space

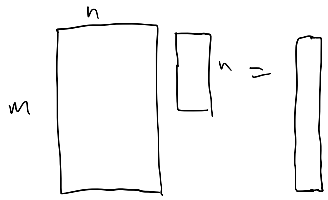
2. Suppose $AA^+ = I$ but $A^+A \neq I$ (right inverse only)

$$R_M R_M = A^+AA^+A = A^+A = R_M$$

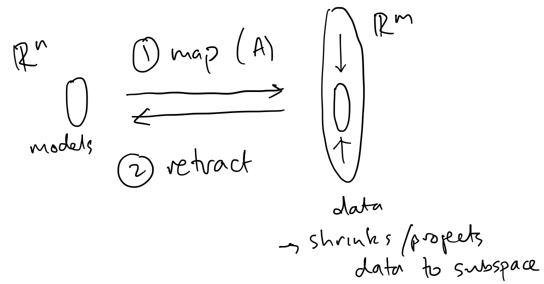
⇒ R_M is a projection on model space.

Visualisation:

Least squares data approx (A^+ is left inverse)



ie



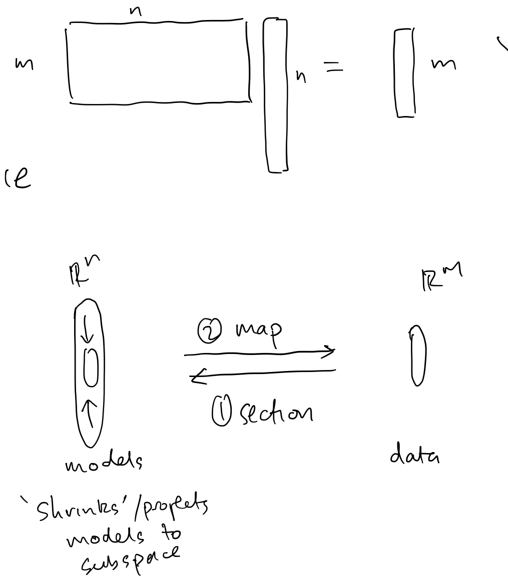
$R_D \neq I$

$R_M = I$

→

Visualisation:

Least squares model reduction (A^{+} is right inverse)



$R_D = I$

$R_M \neq I$

Subtle point (see later):

we actually want $R_D \neq I$ in above, if we have noise!

Example

Projectile motion (Aster et al Ex. 1.1)

$$\boxed{y(t) = a + bt - 0.5ct^2}$$

→ estimate a, b, c given we observe at m times: } def $\theta = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$

$\theta \in \mathbb{R}^n = \mathbb{R}^3$

$$y_1 = a + bt_1 - 0.5t_1^2$$
$$y_2 = a + bt_2 - 0.5t_2^2$$
$$\vdots$$
$$y_m = a + bt_m - 0.5t_m^2$$

ie
$$\begin{bmatrix} 1 & t_1 & -0.5t_1^2 \\ 1 & t_2 & -0.5t_2^2 \\ \vdots & & \\ 1 & t_m & -0.5t_2^2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}$$

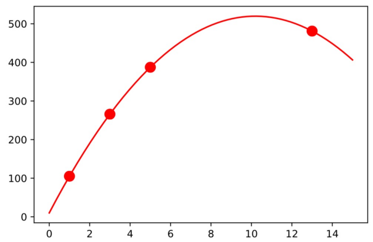
ie $\boxed{F \cdot \theta = y}$

[Note: linear in parameters!]

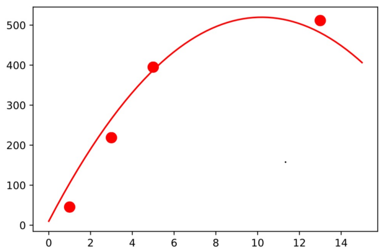
Example : Least squares (param < data)

→ more than three observations
eg 4 obs., 3 param

```python
def fmap(tobs):
    A = np.zeros((len(tobs),3))
    for i, ti in enumerate(tobs):
        A[i,:] = np.array([1,ti,-0.5*ti**2])
    return A

#true parameters
theta_true = np.array([10,100,9.81])

#fine time grid
t = np.linspace(0,15,1000)

#observation times
#tobs = np.array([1,13]) #under-determined
tobs = np.array([1,3,5,13]) #over-determined

#forward map
Aobs = fmap(tobs)

#observed data
yobs = np.dot(Aobs,theta_true) #noise-free
#yobs = np.dot(Aobs,theta_true) + np.random.normal(0,30,size=len(tobs))

#plots
plt.plot(tobs,yobs,'ro',markersize=10)
plt.plot(t,x,'r')
plt.show()
```



noise free

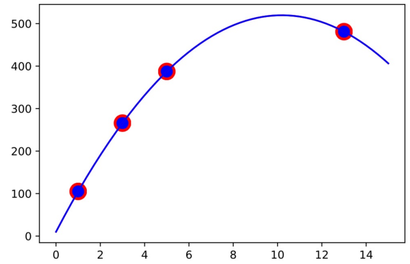


noisy.

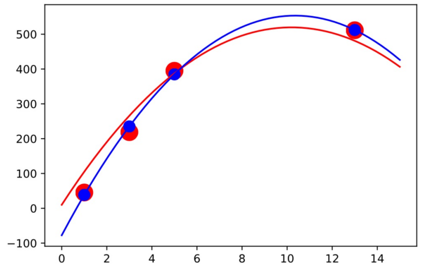
Invert :

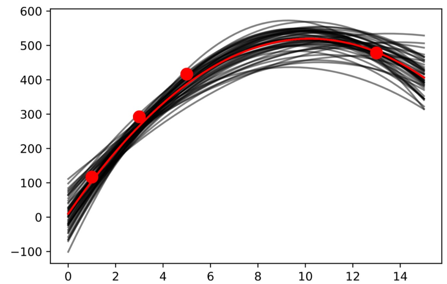
```python
Apinv = np.linalg.pinv(Aobs)
```

noise free



noisy





repeated sampling

Exercise: explore what the parameters are doing compared to true values (this & next)

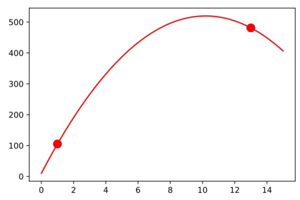
Example : least norm (param > data)

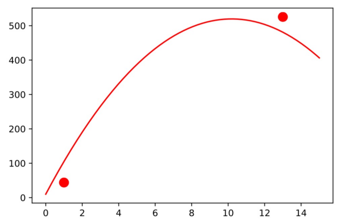
→ less than three observations

Solutions

noise free data

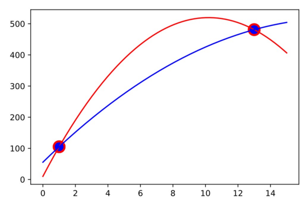
noisy data

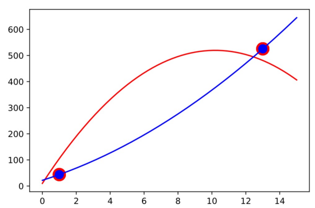


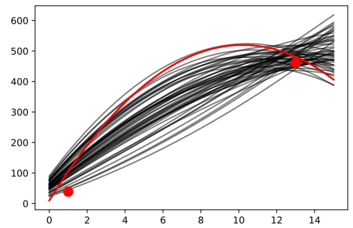


recovered (blue)

recovered (blue)







repeated sampling

Note: we've used the naive model norm $\|\theta\|_2$ including the constant term → here curved actually 'smaller' than straight (see later)

So are we done ?

Recall: two stage def^n of generalised inverse

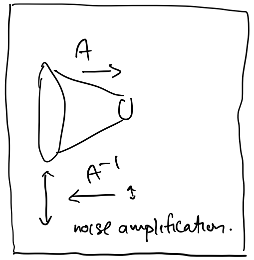
→ First minimise data fit

→ Then minimise model

Inverse problems:

Typically

- undetermined
- also have noise



So: generalised inverse will exactly fit noise !

→ unstable (just like usual inverse)

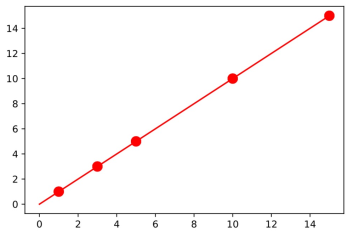
Example: instability despite model 'reduction'

Eg – high degree polynomial } possibly non unique.
– plus observation noise } possibly inconsistent

$$\boxed{y = a_0 + a_1 x + a_2 x + \dots a_n x^n}$$ class

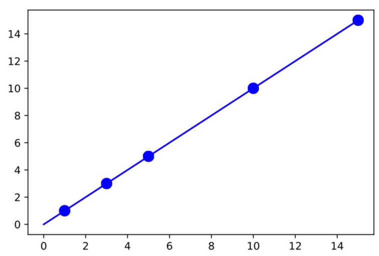
But true model: $a_i = 0$ if $i \neq 1$
$a_1 = 1$

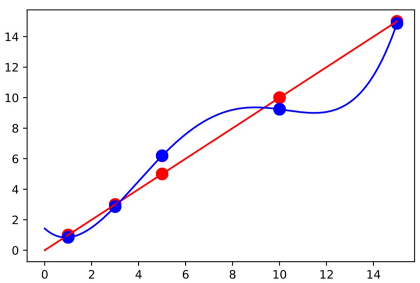
'True' data: (5 observations)



noise free recovery

add noise & recover





unstable! Not reduced enough!

Trade-offs & regularisation

→ Just as we saw in L1 that having an inverse in principle is not enough,

having a generalised inverse is not enough

→ Stability is still a key issue.

However, the 'extremal' or variational characterisation of generalised inverses provides a as to how to control stability as well!

Trade-offs & regularisation

- Instead of two-step:

  - Stage 1: $\min_x \|y - Ax\|$ or $\|y-Ax\|^2$

  Then

  - Stage 2: minimise $\|x\|$ or $\|x\|^2$ among all solutions to stage 1.

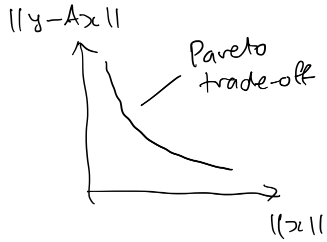
- Try simultaneous minimisation:

  vector/multiobjective problem

$$\boxed{\min_x \left(\|y-Ax\|, \|x\|\right)} \rightarrow$$

simultaneously

Solution curves:



→ allows us to filter noise in underdetermined case, while still 'shrinking' or reducing models to get 'simple' solutions