

# ENGSCI 741

## INVERSE PROBLEMS

*Oliver Maclare*  
*oliver.maclare@auckland.ac.nz*

## MODULE OVERVIEW

### 3. Applications [2-3 lectures/tutorials]

Deblurring, numerical differentiation, parameter estimation, tomography, remote sensing...(we'll see!)

1

3

## MODULE OVERVIEW

Inverse Problems (*Oliver Maclare*) [~9 lectures/tutorials]

### 1. Basic concepts [3 lectures]

Forward vs inverse problems. Well-posed vs ill-posed problems. Algebra of inverse problems (generalised inverses etc). Regularisation and trade-offs.

### 2. More regularisation [3-4 lectures/tutorials]

Higher-order Tikhonov regularisation, truncated singular value decompositions, iterative regularisation. Statistical view of inverse problems

## LECTURE 2: GENERALISED INVERSES

Topics:

- The algebra (and some calculus) of inverse problems
- Why generalised inverses aren't enough!

2

4

## EngSci 741 : Lecture 2.

- Algebra (& a little calculus)  
of inverse problems
  - ↳ Resolving lack of existence &/or uniqueness using generalised inverses
  - ↳ Formulating & solving as optimisation problems
- Two step vs simultaneous optimisation
  - Or, why the generalised inverse also needs regularisation

## Algebra (& some calc) 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, not be unique &/or not be stable

## Linear or nonlinear? Finite or infinite?

- We will discuss some basic algebra of the problem in the linear & finite dimensional setting  
→ i.e. using Linear Algebra in  $\mathbb{R}^n$
  - This can be considered as the 'algebra of linear mappings' in  $\mathbb{R}^n$
- An important question is:  
do these results carry over to the nonlinear & / or infinite-dimensional setting(s)?

Short answer: yes!

(key concepts &  
as long as careful)

## Linear or nonlinear? Finite or infinite?

- Algebra of arbitrary mappings?  
↳ category theory  
(see e.g. Nashed for generalised inverses in category theory setting  
→ existence = axiom of choice!)
  - Discontinuous infinite dimensional linear mappings ('ill-posed')  
↳ ill-conditioned finite dimensional matrices
- Theory is a bit beyond scope, but most of the key tools are applicable to nonlinear setting  
(we'll solve some of these!)

## Linear setting

$$\begin{matrix} \mathbb{R}^n \\ \textcircled{0} \end{matrix} \xrightarrow[A]{?} \begin{matrix} \mathbb{R}^m \\ \textcircled{0} \end{matrix}$$

Consider the system of equations

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

eg

$$\begin{matrix} n \\ m \end{matrix} \left\{ \begin{matrix} \text{rows: eqns} \\ \text{cols: unknowns} \end{matrix} \right. \quad \begin{matrix} n \\ n \end{matrix} = \begin{matrix} m \end{matrix}$$

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

$\rightarrow m > n$ , more rows than cols.

existence?

$$\begin{matrix} n \\ m \end{matrix} \left\{ \begin{array}{l} \rightarrow \text{eqns} > \text{unknowns} \\ \rightarrow \text{possibly inconsistent/overdetermined} \\ \rightarrow \text{'more data than parameters'} \end{array} \right.$$

$\rightarrow m < n$ , more cols than rows

uniqueness?

$$\begin{matrix} n \\ m \end{matrix} \left\{ \begin{array}{l} \rightarrow \text{unknowns} > \text{eqns} \\ \rightarrow \text{possibly many solns} \\ \rightarrow \text{'more param. than data'} \end{array} \right.$$

Tools developed to 'solve' each case

Case 1: possibly inconsistent: more equations than unknowns.

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

$$\begin{matrix} n \\ m \end{matrix} \left\{ \begin{matrix} \text{rows: eqns} \\ \text{cols: unknowns} \end{matrix} \right. \quad \begin{matrix} n \\ \mathbb{R}^n \\ \textcircled{0} \end{matrix} \xrightarrow[A]{?} \begin{matrix} \mathbb{R}^m \\ \textcircled{0} \end{matrix}$$

$\rightarrow$  Assume w.l.o.g. that all n columns are linearly independent

- $y \in R(A) \Leftrightarrow$  there is a unique solution
- $R(A)$ : range/col space/  
image of  $A$   
 $\rightarrow$  see handout

- $y \notin R(A) \Rightarrow$  no (exact) solution



## I. Inconsistent equations: approximate solution

→ Define  $r = y - Ax$   $\left\{ \begin{array}{l} \text{residual 'error'} \\ \text{vector } r \end{array} \right.$

→ Measure size of error with a  
norm  $\|\cdot\|$  (see handout for  
 diff. types)

→ Typically assume  $\|\cdot\|_2$

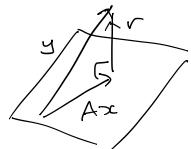
New problem:

minimise  $\|y - Ax\|$ ,  $A$  &  $y$  given  
 $x \in \mathbb{R}^n$

- "best approximation"

- "closest approx"

etc.



Minimiser of  $\|\cdot\|$  & minimiser of  $\|\cdot\|_2^2$

are same ( $x \mapsto x^2$  is monotonic for  $x \geq 0$ )

⇒ least squares approximation:

minimise  $\|y - Ax\|_2^2$  (equiv.  
 $x$  problem)

## Norms, products, summation etc

$$\circ \|x\|_2^2 = \langle x, x \rangle$$

$$= x^T x$$

$$= \sum_i x_i^2$$

$$\circ Ax = \sum A_{ij} x_j$$

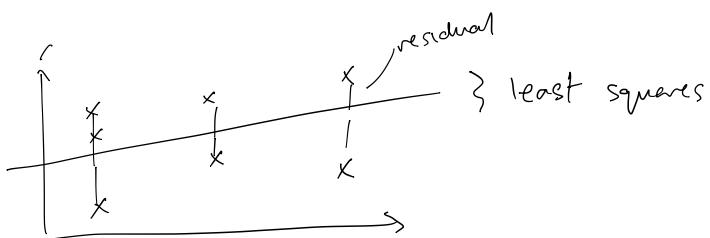
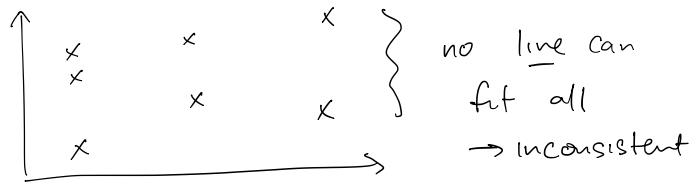
$= A_{ij} x_j$  (Einstein summation convention  
 → sum over repeated indices)

so:

$$\circ y^T Ax = y_i A_{ij} x_j$$

See handout (?) (we'll do some practice!)

### Illustration



Derivation via calc (can do geometrically too!)

$$\min_x \|y - Ax\|_2^2 = \min_x \langle y - Ax, y - Ax \rangle$$

$$= \min_x f(x)$$

where  $f(x) = \langle y - Ax, y - Ax \rangle$

$$[ = (y - Ax)^T (y - Ax) ]$$

$$\left. \begin{aligned} &= \langle y, y \rangle - 2 \langle y, Ax \rangle + \langle Ax, Ax \rangle \\ &= \|y\|_2^2 - 2y^T Ax + \|Ax\|_2^2 \\ &= y_i y_i - 2y_i A_{ij} x_j + (A_{ij} x_j)(A_{ik} x_k) \end{aligned} \right\}$$

etc

Differentiating matrices, vectors etc?

→ 'just' tensor calc!

→ can use index notation  
OR  
remember identities

]} see 3 key rules.

$$\min f(x) \Rightarrow \boxed{Df(x) = 0} \quad (\text{if diff.})$$

Here (convex, differentiable etc)

$$\min f(x) \Leftrightarrow Df(x) = 0$$

(note:  $f$  is scalar,  $Df$  is vector)

Detailed steps

Consider index form:

$$y_i y_i - 2y_i A_{ij} x_j + (A_{ij} x_j)(A_{ik} x_k)$$

$$\sim \sim \sim$$

$$\underbrace{\langle y, y \rangle}_{(A x)^T (A x)} + \underbrace{2 \langle y, Ax \rangle}_{x^T A^T A x} + \underbrace{\langle Ax, Ax \rangle}_{(A x)^T (A x)}$$

$$(A x)^T (A x)$$

$$x^T \overset{=} {A^T A} x$$

=

$$x_i A_{ij}^T A_{jk} x_k$$

$\{Df(x)\}_k$  for each term      ( $D_i$ : derivative  
 $\nabla$ : gradient)

f scalar

$\rightarrow Df$  is row vector

$\rightarrow \nabla f = (Df)^T$  is col vector

- $\frac{\partial}{\partial x_k} (y_i y_i) = 0$

- $\frac{\partial}{\partial x_k} (2 y_i A_{ij} x_j) = 2 y_i \underbrace{A_{ij}}_{\frac{\partial x_j}{\partial x_k}} x_{j,k}$

$$\frac{\partial x_j}{\partial x_k} = \begin{cases} 1 & \text{if } j=k \\ 0 & \text{if } j \neq k \end{cases}$$

$$= \delta_{jk}$$

$$\Rightarrow \frac{\partial}{\partial x_k} (\quad) = 2 y_i A_{ik}$$

$$= \underline{2 y^T A} \quad (\text{row vector})$$

Define  $\nabla f = [Df(x)]^T$  as col.

- $\frac{\partial}{\partial x_k} (x_i A_{ij}^T A_{jk} x_k) = \frac{\partial}{\partial x_k} (x_i P_{ij} x_j)$

where  $A_{ij}^T A_{jk} = P_{ik}$



so

- $\frac{\partial}{\partial x_k} (x_i P_{ij} x_j)$

$$= x_{i,k} P_{ij} x_j + x_i P_{ij} x_{j,k} \quad (\text{chain})$$

$$= \delta_{ik} P_{ij} x_j + x_i P_{ij} \delta_{jk}$$

$$= P_{kj} x_j + x_i P_{ik} \quad (\text{no } k \text{ sum})$$

$$= x_j P_{kj} + x_i P_{ik} \quad (\text{no } k \text{ sum})$$

$$= x_j P_{j,k}^T + x_i P_{ik} \quad (\text{no } k \text{ sum})$$

$$= x_i (P_{ik}^T + P_{ik}) \quad (\text{no } k \text{ sum})$$

$$= x^T (P^T + P)$$

But note:  $P = A^T A \Rightarrow P^T = A^T A = P$

so here, get

$$\underline{2 x^T A^T A} \quad (\text{row})$$

$$\nabla f = (Df)^T = 2 A^T A x \quad (\text{col})$$

Phew!

$$\text{so } f(x) = \|y\|_2^2 - 2y^T Ax + \|Ax\|_2^2$$

$$= y^T y - \underbrace{2y^T Ax}_{\text{scalar}} + \underbrace{x^T A^T A x}_{\text{scalar}}$$

minimised when:

$$\nabla f = -2y^T A + 2x^T A^T A = 0 \quad \left[ \begin{array}{l} \text{vector} \\ \text{eqns} \end{array} \right]$$

$$\text{or } \nabla f = -2A^T y + 2A^T A x = 0$$

$$\Leftrightarrow \boxed{A^T A x = A^T y}$$

$$\left[ \text{or } A^T(Ax - y) = 0, \text{ geometric condition} \right.$$

$\rightarrow \text{residual } \in N(A^T)$

3 Key rules: ( $A$ ,  $a$  independent of  $x$ )

- $\frac{\partial}{\partial x} (x^T A x) = x^T (A + A^T)$
- $\frac{\partial}{\partial x} (y^T x) = y^T$
- $\frac{\partial}{\partial x} (a) = 0$

→ know these!

## Least squares approximation

We have seen this leads to....

The normal equations:

$$\begin{cases} \text{normal?} \\ A^T(Ax - y) = 0 \end{cases}$$

$$\boxed{A^T A x = A^T y}$$

→ Since we assume the n cols of  $A$  are linearly independent then  $A^T A$  is invertible (see handout) & so get unique approximate soln

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

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

## Left inverses

$$\mathbb{R}^n \xrightleftharpoons[L]{A} \mathbb{R}^m$$

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

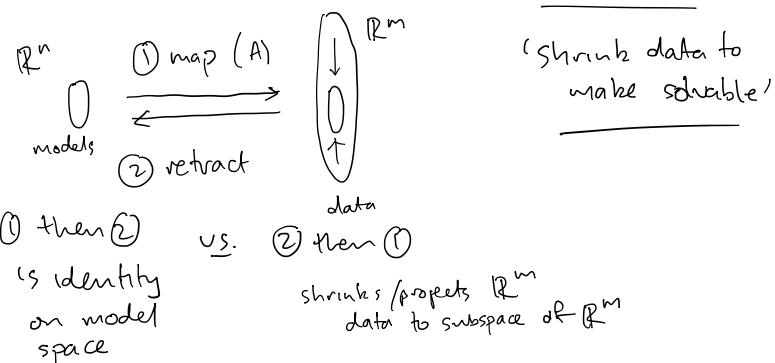
$$L A = I \quad \left\{ \begin{array}{l} A : m \times n \\ L : n \times m \\ I : n \times n \end{array} \right.$$

For least squares data approximation, we have

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

$A$  left inverse exists when ~~rows~~ rows  
 $\Rightarrow$  ~~cols~~ cols of  $A$  & the cols are LI

$$m \begin{array}{|c|} \hline n \\ \hline \end{array} \begin{array}{|c|} \hline n \\ \hline \end{array} = \begin{array}{|c|} \hline m \\ \hline \end{array}$$



## Tools developed to 'solve' each case

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

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

$$m \left\{ \begin{array}{|c|} \hline n \\ \hline \end{array} \right\} \xrightarrow[A]{\quad} \begin{array}{|c|} \hline \mathbb{R}^n \\ \hline \end{array} \xrightarrow[A]{\quad} \begin{array}{|c|} \hline \mathbb{R}^m \\ \hline \end{array} \xrightarrow{\quad} \begin{array}{|c|} \hline y \\ \hline \end{array}$$

$\rightarrow$  Assume w.l.o.g. that all  $m$  rows are linearly independent

$\rightarrow$  Assume  $Ax = y$  has at least one solution, i.e.  $y \in \mathbb{R}(A)$  (range / image of  $A$ )

General solution of  $Ax = y$ :

$$x = x^* + x_0$$

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 solns

- If we want to 'pick out' a single soln (we don't always!), a 'natural' choice is to choose the 'smallest' (think: 'simplest' or 'most efficient' soln)
- Again, this is relative to a particular norm, eg  $\|\cdot\|_2 = \text{l}_2 \text{ norm}$ .

⇒ Problem to solve:

$$\begin{array}{ll} \min_{x} & \|x\| \\ \text{st. } & Ax = y \end{array} \quad \left. \begin{array}{l} \text{note: here} \\ \text{assuming} \\ \text{exactly solvable} \end{array} \right\}$$

Equivalent to:

$$\begin{array}{ll} \min_{x} & \|x\|^2 \\ \text{st. } & Ax = y \end{array} \quad \left. \begin{array}{l} \text{'least squares} \\ \text{soln'} \\ (\text{cf least squares} \\ \text{approximation}) \end{array} \right\}$$

## Least norm problem ('Model reduction')

Using duality (assignment?)

you can show that the least-squares solution to the minimum norm problem requires solving the dual problem

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

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

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

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

## Right inverses

$$\mathbb{R}^n \xrightarrow[A]{R} \mathbb{R}^m$$

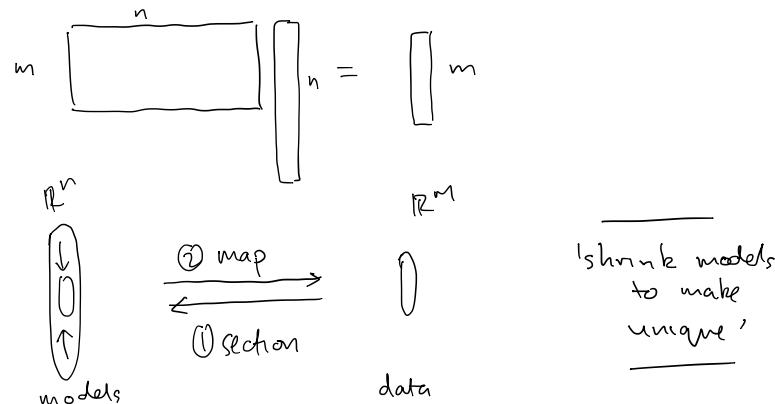
$R$  is a left inverse (a section in category theory) for  $A$  if it satisfies

$$AR = I \quad \begin{cases} A: m \times n \\ R: n \times m \\ I: m \times m \end{cases}$$

For least norm problems, we have

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

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



② then ① while

'shrinks' / projects  
models to  
subspace

① then ② is

identity on  
data space

'shrink models  
to make  
unique'

## Unification: generalised inverses

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

→ This will solve both existence & uniqueness issues at the same time!

↳ spoiler alert: but not stability issues!

| we will see an alternative way

| to combine/unity the solutions when we discuss regularisation | )

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

⇒ i.e. it

$$\boxed{\text{solves } \min ||y - Ax|| \text{ AND } \min ||x||}$$

(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: minimise  $\underset{x}{\|y - Ax\|}$  or  $\|y - Ax\|^2$

Then

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

The result is a matrix  $A^+$  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  $A^T A$  is invertible

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

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

$$\begin{aligned} A^+ A &= (\underbrace{A^T A}_{M})^{-1} \underbrace{A^T A}_{M} \\ &= M^{-1} M \\ &= I \quad \left. \right\} \text{is a left inverse too.} \end{aligned}$$

• now suppose  $A A^T$  is invertible  
⇒ least norm right inverse  $R = A^T (A A^T)^{-1}$  exists

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

$$\begin{aligned} A A^+ &= (A A^T) (A A^T)^{-1} \\ &= I \quad \left. \right\} \text{is a right inverse too} \end{aligned}$$

## Generalised vs pseudo-inverses

We don't have to take

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

in each case.

Generalised inverses defined via (1)

are not unique

- add extra conditions
  - depends on goals
- eg different norms etc  
diff. inverses

Pseudo?

→ (& will assume unless otherwise)

Eg what we've been using (least sq/norm)  
is the Moore-Penrose Pseudo-inverse

This satisfies

$$\boxed{\begin{array}{l} A A^+ A = A \\ A^+ A A^+ = A^+ \\ (A^+ A)^T = A^+ A \\ (A A^+)^T = A A^+ \end{array}}$$

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

## Model resolution, data resolution operators

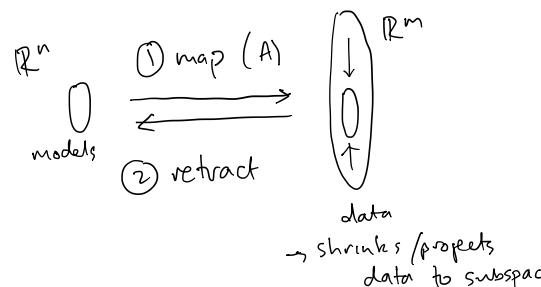
$$\boxed{R_D = A A^+} \quad \left. \right\} \text{ how much data is 'shrunk' or smeared}$$

$$\boxed{R_M = A^+ A} \quad \left. \right\} \text{ how much model is 'shrunk' or smeared}$$

least squares data approx

$$\begin{matrix} n \\ m \end{matrix} \xrightarrow{} \begin{matrix} n \\ n \end{matrix} = \begin{matrix} m \\ m \end{matrix}$$

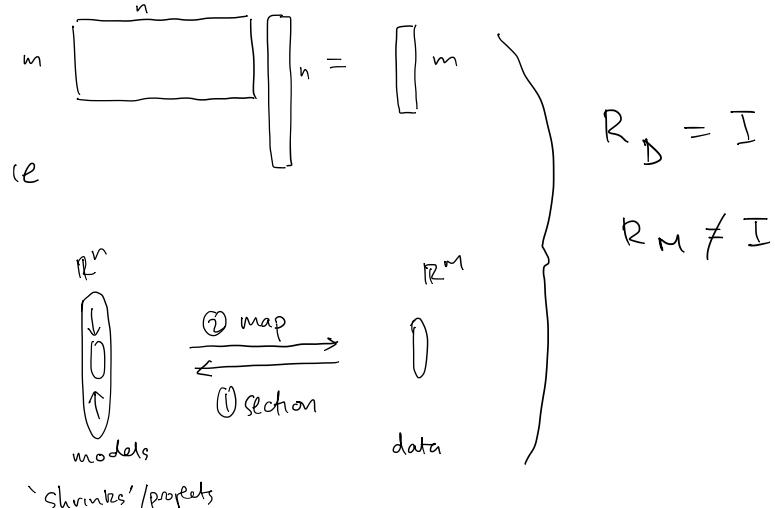
i.e.



$$R_D \neq I$$

$$R_M = I$$

## least squares model reduction



'shrink's' / projects  
models to  
subspace

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)

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

→ estimate  $a, b, c$  given we observe at  $m$  times:

$$\text{def } \theta = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

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

$$y_1 = a + b t_1 - 0.5 t_1^2$$

$$y_2 = a + b t_2 - 0.5 t_2^2$$

$$\vdots$$

$$y_m = a + b t_m - 0.5 t_m^2$$

i.e.

$$\begin{bmatrix} 1 & t_1 & -0.5 t_1^2 \\ 1 & t_2 & -0.5 t_2^2 \\ \vdots & \vdots & \vdots \\ 1 & t_m & -0.5 t_m^2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

i.e.

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

[ Note: linear in parameters! ]

Example : least squares ( param < data)

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

```
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])

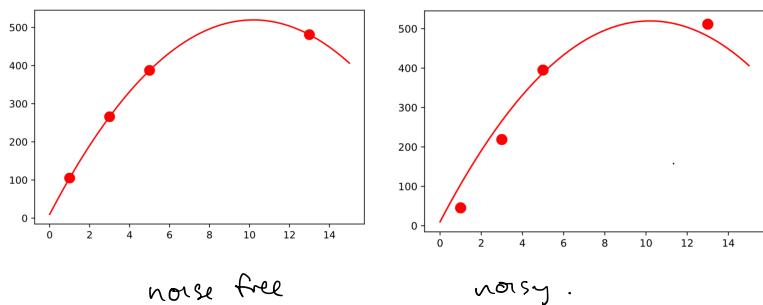
#fine time grid
t = np.linspace(0, 15, 1000)

#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))

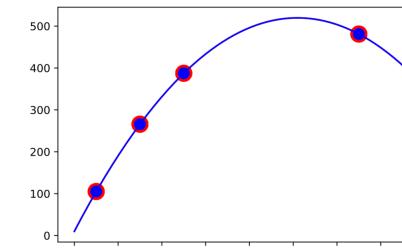
#plots
plt.plot(tobs, yobs, 'ro', markersize=10)
plt.plot(t, x, 'r')
plt.show()
```



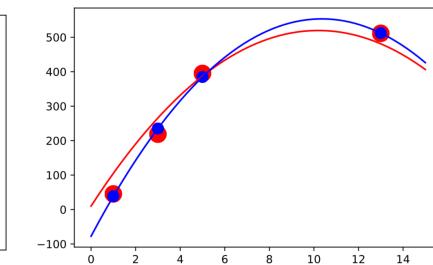
Invert :

Apinv = np.linalg.pinv(Aobs)

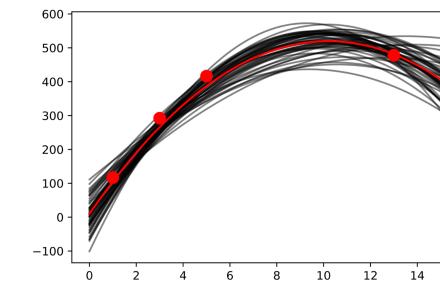
noise free



noisy



repeated sampling



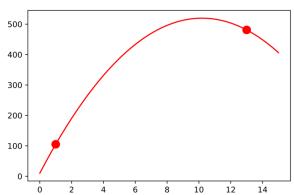
Exercise : explore what the parameters are doing compared to true values (this & next)

Example : least norm (param > data)

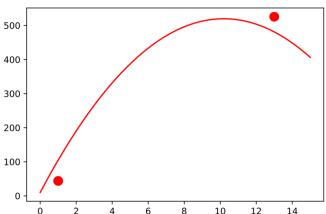
→ less than three observations

Solutions

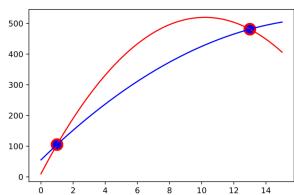
noise free data



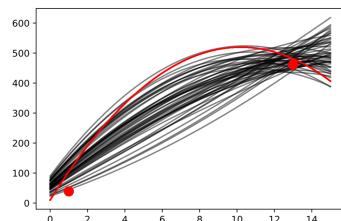
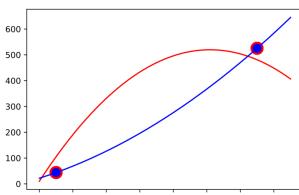
noisy data



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?

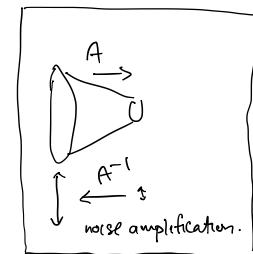
Recall : two stage def<sup>n</sup> of generalised inverse

- First minimise data fit
- Then minimise model

Inverse problems:

Typically

- underdetermined
- also have noise



So: generalised inverse will  
exactly fit noise!

→ unstable (just like  
 usual inverse)

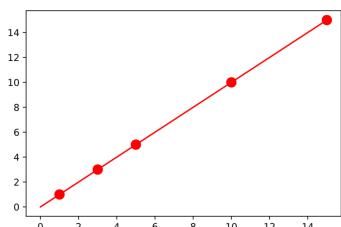
Example: instability despite model 'reduction'

- Eg - high degree polynomial  $\Rightarrow$  possibly non-unique.  
- plus observation noise  $\Rightarrow$  possibly inconsistent

$$y = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n \quad \text{class}$$

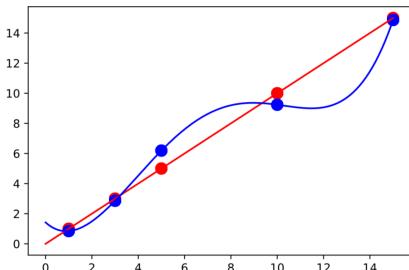
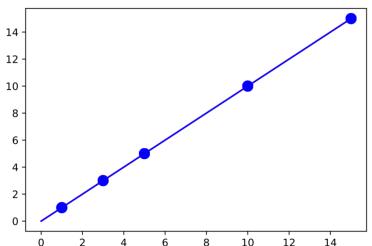
But true model:  $a_i = 0$  if  $i \neq 1$   
 $a_1 = 1$

'True' data: (5 observations)



noise  
free  
recovery

add noise &  
recover



unstable! Not reduced  
enough!

### Trade-offs & regularisation

$\rightarrow$  Just as we saw in L1 that having an inverse in principle is not enough,

having a generalised inverse is not enough

$\rightarrow$  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

o Instead of two-step:

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

Then

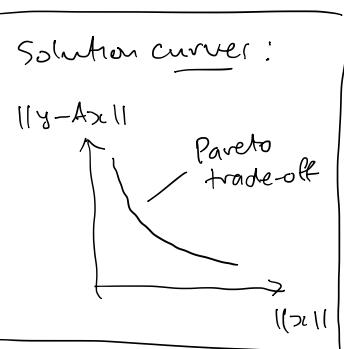
o Stage 2: minimise  $\|x\|$  or  $\|x\|^2$  among  
all solutions to stage 1.

o Try simultaneous minimisation:

vector/multi objective problem

$$\min_x (\|y - Ax\|, \|x\|)$$

simultaneously



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