ENGSCI 721

INVERSE PROBLEMS

Oliver Maclaren
oliver.maclaren@auckland.ac.nz

MODULE OVERVIEW

Inverse Problems (*Oliver Maclaren*) [~8 lectures/2-3 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 lectures]
   Higher-order Tikhonov regularisation, truncated singular value decompositions, iterative regularisation.

MODULE OVERVIEW

3. *Statistical view of inverse problems I* [2 lectures]
   Bayesians, Frequentists and all that. Basic frequentist analysis. Linearisation and covariance propagation.

LECTURE 2: GENERALISED INVERSES

Topics:

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

Eng Sci 721 : 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, might not be unique &/or might not be stable

Note: mappings, measurements & vectors

We might have an 'exact' model of the form

$\boxed{y = a + bx}$, for $\boxed{\text{scalars}}$ x, y

If we take a series of 'noisy' $\boxed{\text{measurements}}$ we get eg:

$$y_{obs,i} = (a + bx)_i + e_i$$

$$\Rightarrow y_{obs,i} = a + x_i + e_i$$

for $i = 1, \dots, n$.

These lead to $\boxed{\text{vector}}$ eqns in terms of the noisy observations/realisations:

$$\begin{bmatrix} y_{obs,1} \\ \vdots \\ y_{obs,n} \end{bmatrix} = a \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + b \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} e_1 \\ \vdots \\ e_n \end{bmatrix}$$

eg $\boxed{\bar{y}_{obs} \approx a\bar{1} + b\bar{x}}$ } $\bar{y}, \bar{x}, \bar{1}$ vectors

(I usually drop the explicit overbars on vectors)

Linear or nonlinear? Finite or Infinite?

- We will discuss some basic algebra of the problem in the linear & finite dimensional setting
  - → ie 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

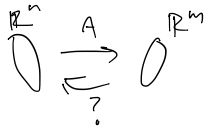
  (see eg Nashed / Maclaren & Nicholson for generalised inverses in category theory setting
  → existence ≡ axiom of choice!)

- 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

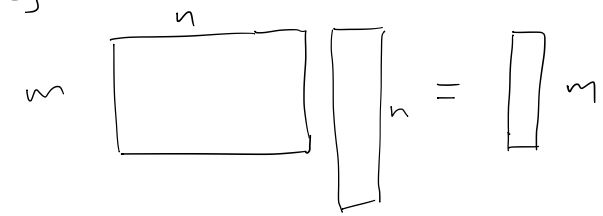


Consider the system of equations

$$Ax = y$$

- A is $m \times n$ matrix
- $x \in \mathbb{R}^n$ vector
- $y \in \mathbb{R}^m$ vector

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

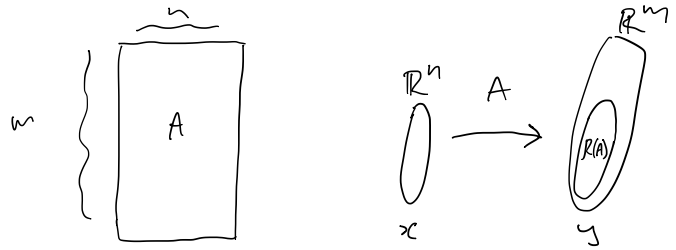
uniqueness?

→ unknowns > eq^{n}s
→ possibly many sol^{n}s
→ 'more param. than data'

Tools developed to 'solve' each case

Case 1: possibly inconsistent: more equations than unknowns.

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



→ Assume w.l.o.g. that all n columns are linearly independent

- $y \in R(A)$ $\Longleftrightarrow$ there is a unique solution

$R(A)$: range/col space/ image of A

→ see handout

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

→

1. Inconsistent equations: approximate solution

→ Define $r = y - Ax$ } residual 'error'

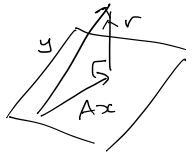
→ Measure size of error with a norm $\|\cdot\|$ (see handout for diff. types)

→ Typically assume $\|\cdot\|_2$

New problem:

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

- "best approximation"
- "closest approx"
- etc.



Minimiser of $\|\ \|$ & minimiser of $\|\ \|_2^2$ are same ($x \mapsto x^2$ is monotonic for $x \geq 0$)

⇒ Least squares approximation:

$$\underset{x}{\text{minimise}} \ \|y - Ax\|_2^2 \quad \text{(equiv. problem)}$$

Norms, products, summation etc

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

  $= x^T x$

  $= \sum_i x_i^2$

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

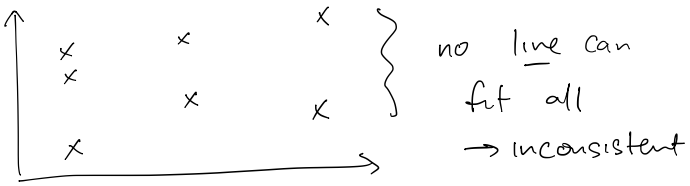
  $= A_{ij} x_j$ (Einstein summation convention → sum over repeated indices)

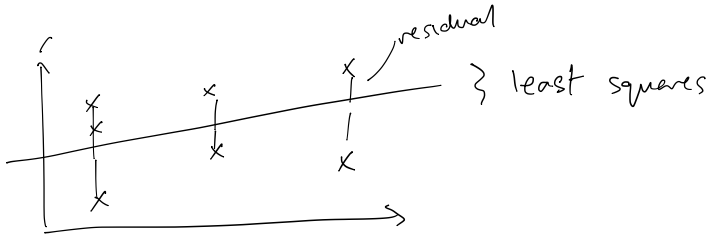
So:

- $y^T A x = y_i A_{ij} x_j$

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

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

same, different notation.

$$= \langle y, y \rangle - 2 \langle y, Ax \rangle + \langle Ax, Ax \rangle$$

$$= y^T y - 2 y^T A x + x^T A^T A x$$

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

etc

Differentiating vectors, matrices, tensors wrt vectors, matrices, tensors - - -

Requires:

Matrix calculus, Tensor calculus etc

→ multiple conventions / notation

(see eg Wiki page on Matrix Calculus)

I will sketch here, but then give you three key rules you can use instead of remembering details!

Conventions

→ I'll use $d_x f$ for derivative of f wrt x, regardless of whether f, x are scalar, vector etc.

→ use 'Jacobian layout', eg vector f, x:

$$[d_x f]_{ij} = \frac{\partial f_i}{\partial x_j} = \begin{array}{c} \\ f_1 \\ \vdots \\ f_m \end{array} \begin{array}{c} \begin{array}{ccc} x_1 & \cdots & x_n \end{array} \\ \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} \end{array}$$

Note, if f is scalar-valued, this implies $d_x f$ is a row vector:

$$d_x f = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_n} \end{bmatrix}$$

Hence this is sometimes written

$$d_x f = \frac{\partial f}{\partial x^T}$$

to indicate the 'layout' direction is 'row like' in x.

We can define the gradient of f as

$$\nabla_x f = (d_x f)^T = \frac{\partial f^T}{\partial x}$$

(often drop transpose on f)

which again indicates how to 'lay out' results.

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

Constant

$$d_x(a) = \frac{\partial}{\partial x^T}(a) = 0^T$$

Linear

$$d_x(Ax) = \frac{\partial}{\partial x^T}(Ax) = A$$

Quadratic

$$d_x(x^T A x) = \frac{\partial}{\partial x^T}(x^T A x)$$
$$= x^T(A + A^T)$$

Know these !

Extra :

Additional principles for 'deriving' results (see handout) :

- Derivative of scalar, possibly dep. on x :

$$d_x(a(x)) = d_x(a) = d_x(a^T)$$

- (Multivariable vector) chain rule

$$d_x(f(h(x), g(x)) = \frac{\partial}{\partial x^T} f(h, g)$$
$$= (d_g f)\cdot(d_x g) + (d_h f)\cdot(d_x h)$$
$$= \frac{\partial f}{\partial g^T}\frac{\partial g}{\partial x^T} + \frac{\partial f}{\partial h^T}\frac{\partial h}{\partial x^T}$$

Imply eg 'product rule' of form :

$$d_x(h^T g) = h^T d_x g + g^T d_x h$$
$$= h^T \frac{\partial g}{\partial x^T} + g^T \frac{\partial h}{\partial x^T}$$

Back to least squares!

$$\min_x f(x) = y^T y - 2y^T A x + x^T A^T A x$$

$\Rightarrow$ set $d_x f = 0^T$ (1) (row vector)

(or $\nabla_x f = 0$ ...)

3 Rules:

$$d_x (y^T y) = 0^T$$

$$d_x (-2y^T A x) = -2y^T A$$

$$d_x (x^T A^T A x) = x^T (A^T A + (A^T A)^T)$$
$$= 2x^T A^T A$$

(1) $\Rightarrow$ $-2y^T A + 2x^T A^T A = 0^T$

$\Rightarrow$ $-A^T y + A^T A x = 0$

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

Least squares approximation

We have seen this leads to....

The normal equations:

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

[normal? $A^T(Ax - y) = 0$ geometric]

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

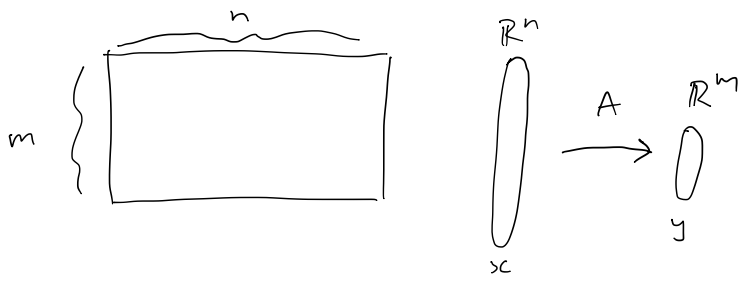
$$\boxed{x^* = (A^T A)^{-1} A^T y}$$

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

Tools developed to 'solve' each case

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

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



→ 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 $Ax = y$:

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

where $x_0 \in N(A)$ & x^* is any soln of $Ax = y$

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

2. Underdetermined equations: least norm soln's

- 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 $\| \cdot \|_2 = l_2$ norm.

⇒ Problem to solve:

$$\min_x \|x\| \quad \text{s.t.} \quad Ax = y$$

} note: here assuming exactly solvable

Equivalent to:

$$\min_x \|x\|^2 \quad \text{s.t.} \quad Ax = y$$

} 'least squares soln' (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

$$AA^T v = -2y \quad \text{for } v^*$$

& then obtaining $x^* = -\frac{1}{2} A^T v^*$

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

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

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

[soln sketch : Lagrange multipliers (beyond scope...?)

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

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

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

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

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

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

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

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

$$AA^T \lambda = -2y$$

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

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

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

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

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

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

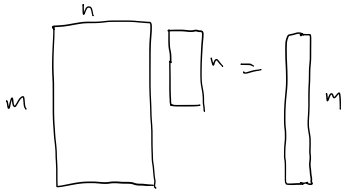
$$\boxed{LA = I}$$

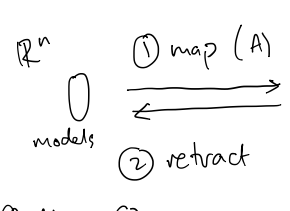
$A: m \times n$
$L: n \times m$
$I: n \times n$

For least squares data approximation, we have

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

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





① then ② is identity on model space

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

'shrink data to make solvable'

Right inverses

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

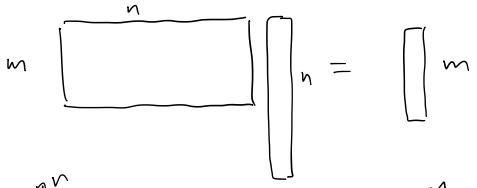
R is a right inverse (a section in category theory) for A if it satisfies

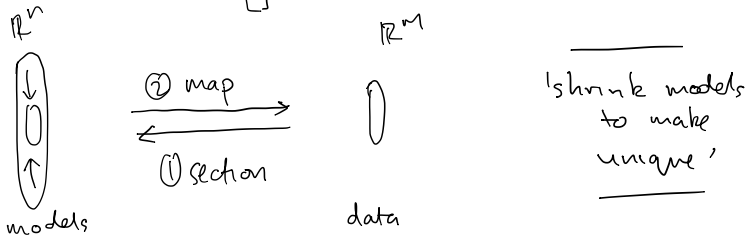
$$\boxed{AR = I} \quad \begin{cases} A: m \times n \\ R: n \times m \\ I: m \times m \end{cases}$$

For least norm problems, we have

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

A right inverse exists when # cols ≥ # 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 $A^\dagger$ ← 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 $A^\dagger$ is just

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

Note: • suppose A^TA is invertible

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

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

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

• now suppose AA^T is invertible

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

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

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

Generalised vs pseudo-inverses

We don't have to take

- $A^\dagger = L = (A^T A)^{-1} A^T$ = least squares
- $A^\dagger = R = A^T (AA^T)^{-1}$ = 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

$$AA^\dagger A = A$$
$$A^\dagger A A^\dagger = A^\dagger$$
$$(A^\dagger A)^T = A^\dagger A$$
$$(AA^\dagger)^T = AA^\dagger$$

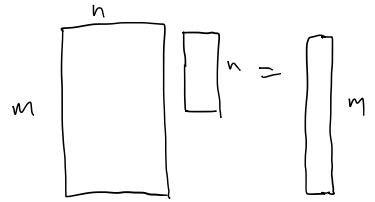
least squares, least norm generalised inverse = 'pseudo-inverse'

Model resolution, data resolution operators

$$R_D = AA^\dagger$$ } how much data is 'shrunk' or smeared

$$R_M = A^\dagger A$$ } how much model is 'shrunk' or smeared

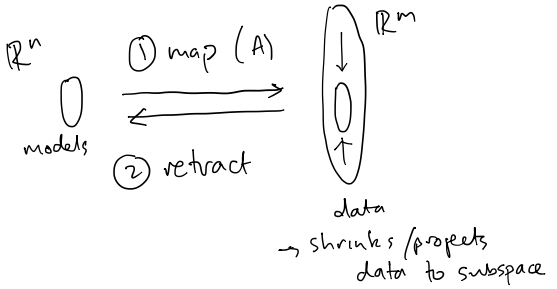
Least squares data approx ($A^\dagger$ is left inverse)



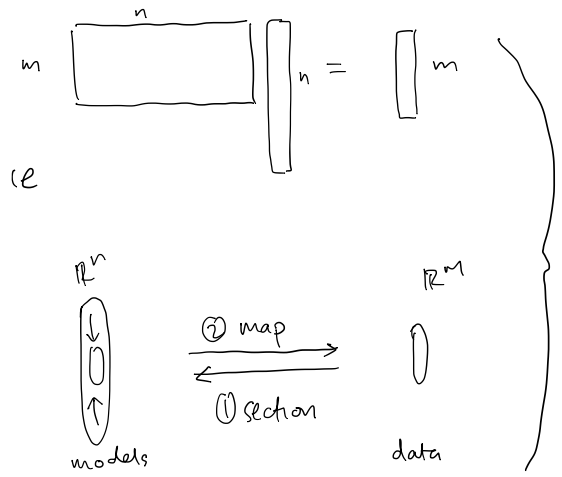
ie

$R_D \neq I$

$R_M = I$



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)

$$y(t) = a + bt - 0.5ct^2$$

→ estimate a, b, c given we observe at m times: } def $\theta = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$, $\theta \in \mathbb{R}^n = \mathbb{R}^3$

$$\begin{aligned} y_1 &= a + bt_1 - 0.5t_1^2 \\ y_2 &= a + bt_2 - 0.5t_2^2 \\ &\vdots \\ y_m &= a + bt_m - 0.5t_m^2 \end{aligned}$$

ie

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

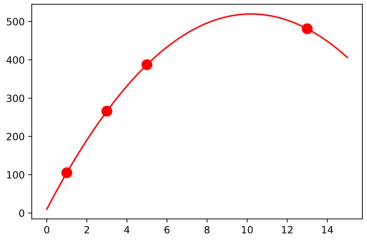
ie $\boxed{F \cdot \theta = y}$

[Note: linear in parameters!]

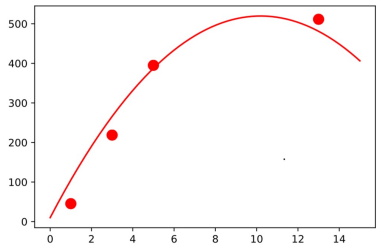
Example : Least squares (param < data)

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

```
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

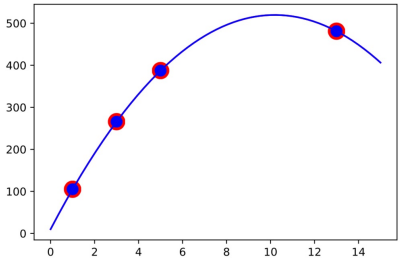


noisy.

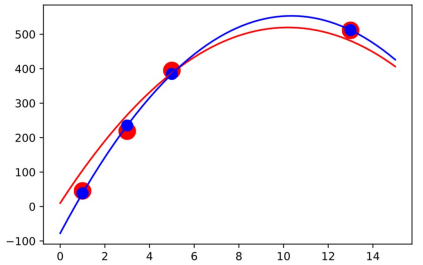
Invert:

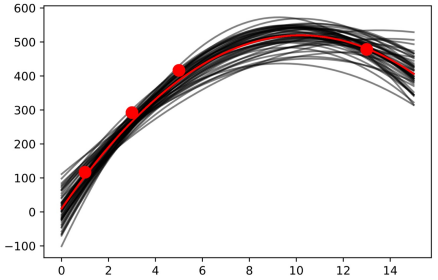
```
Apinv = np.linalg.pinv(Aobs)
```

noise free



noisy





repeated sampling

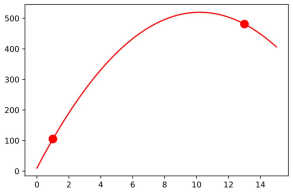
Exercise: explore what the parameters are doing compared to true values (this & next)

Example : least norm (param > data)

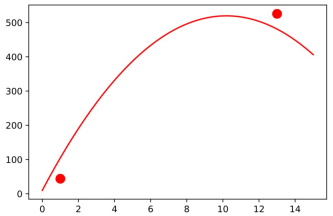
→ less than three observations

Solutions

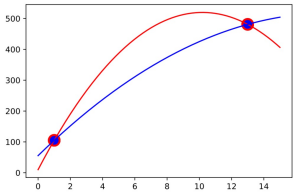
noise free data



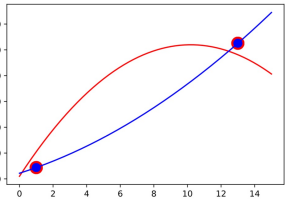
noisy data

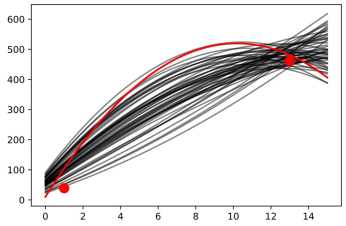


recovered (blue)



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

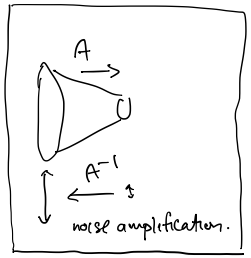
→ First minimise data fit

→ Then minimise model

Inverse problems:

Typically

- undetermined
- also have noise



So: generalised inverse will exactly fit noise!

→ unstable (just like usual inverse)

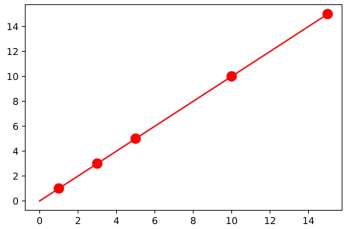
Example : instability despite model 'reduction'

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

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

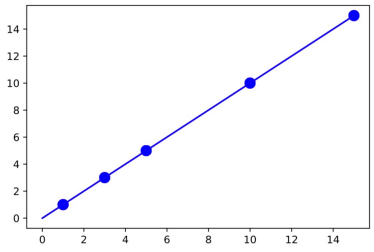
But true model : $a_i = 0$ if $i \neq 1$
$a_1 = 1$

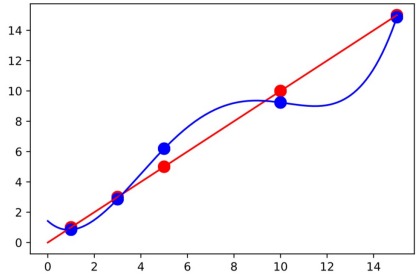
'True' data : (5 observations)



noise free recovery

add noise & recover





unstable ! Not reduced enough !

Trade-offs & regularisation

→ Just as we saw in LI 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: $\underset{x}{\text{minimise}} \; \|y - Ax\|$ or $\|y - Ax\|^2$

Then

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

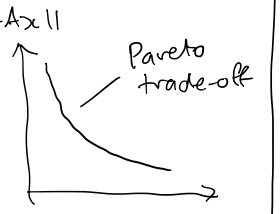
- Try simultaneous minimisation:

vector/multi objective problem

$$\min_x \left(\|y - Ax\|, \|x(1)\| \right) \rightarrow$$

simultaneously

Solution curve:



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