MRC | Cognition and Brain Sciences Unit

General Linear Models:

Linear equations and matrix inversion

Saskia Helbling

Introduction to Signal Analysis in Matlab - MRC CBSU

Saskia.Helbling@mrc-cbu.cam.ac.uk

GLMs = multivariate linear regression

- *Simple linear regression*: $y = x * \beta + \varepsilon$, for example $34 = \beta_0 + 4 * \beta_1 + \varepsilon$
- *Multiple linear regression*: $\mathbf{y} = \mathbf{X} * \boldsymbol{\beta} + \boldsymbol{\varepsilon}$, for example $34 = \beta_0 + 4 * \beta_1 + 2 * \beta_2 + \varepsilon$

regression coefficients

$$\mathbf{y} = \begin{pmatrix} y_1 \\ \dots \\ y_j \\ \dots \\ y_R \end{pmatrix} = \mathbf{X}\beta = \begin{pmatrix} X_{11} & \dots & X_{1j} & \dots & X_{1C} \\ \dots & \dots & \dots & \dots & \dots \\ X_{i1} & \dots & X_{ij} & \dots & X_{iC} \\ \dots & \dots & \dots & \dots & \dots \\ X_{R1} & \dots & X_{Rj} & \dots & X_{RC} \end{pmatrix} \begin{pmatrix} \beta_1 \\ \dots \\ \beta_j \\ \dots \\ \beta_R \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \dots \\ \varepsilon_j \\ \dots \\ \varepsilon_R \end{pmatrix}$$

Data (dependent variable)

Error or noise

Design matrix with regressors (independent variables)

GLMs = multivariate linear regression

- *Multivariate linear regression*: $\mathbf{Y} = \mathbf{X} * \boldsymbol{\beta} + \mathbf{E}$ ($\mathbf{Y}$, $\boldsymbol{\beta}$ and $\mathbf{E}$ are now matrices)

- GLMs can be used to quantify the effect of different experimental variables on our BOLD signal or reaction times or oscillatory activity or …

- GLMs incorporate a number of different statistical models (e.g. ANOVA, ANCOVA,MANOVA, t-tests…)

Linear equations

Simple linear regression

2*a=10
What's a?

2*a + 5=10
What's a?

[2]*a=[10]

[2]*a=[5]
What's a?

Multiple linear regression

2*a + 5 = b
b=10
What are a and b?

$$\begin{pmatrix} 2 & -1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} -5 \\ 10 \end{pmatrix}$$

In Matlab:

[2 -1; 0 1]*[a b]' = [-5 10]'

Problem:

- We have an equation of the form **Mx**=**y**
- We know **M** and **y**
- We want to know **x**

Inverse Matrices

If only we had a matrix $\mathbf{M}^{-1}$ with the property

$$\mathbf{M}^{-1}\mathbf{M} = \mathbf{I}$$

(just like (1/3)*3 = 1)

$$\mathbf{I} = \begin{matrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{matrix}$$

because then we could solve an linear equation:

$$\mathbf{M} * \mathbf{x} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

by multiplying both sides of the equation with M^{-1}

$$\underbrace{\mathbf{M}^{-1}\mathbf{M}}_{identity} * \mathbf{x} = \mathbf{M}^{-1}\begin{pmatrix} 1 \\ 1 \end{pmatrix} => \mathbf{x} = \mathbf{M}^{-1}\begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$\mathbf{M}^{-1}$ is the “inverse matrix” of **M.** In Matlab: **inv**

Linear equations

Multiple linear regression

2*a + 5 = b
b=10
What are a and b?

[2 -1; 0 1]*[a b]' = [-5 10]'

[a b]' = inv([2 -1; 0 1])[-5 10]'

Solve the following linear equation!

a+2*y = 1
a-b = 2
What are a and b?

```
% define your matrix M
M = [2 -1;0 1];

% and your dependent variable
x = [-5 10]'
% solve the GLM
sol = inv(M)*x

% check if it's really a solution
M*sol

% check the inverse
inv(M)*M

% Back slash operator
solBs = M\x;  % faster and more robust computation

% Matrix inversion is not element-wise division!
inv(M)
M.^-1

% A singular matrix is not invertible
singM = [2 -1; -4 2];
inv(singM)
```

Basis functions

In our last example we were solving the equation

$$\begin{pmatrix} 2 & -1 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} -5 \\ 10 \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} 2 \\ 0 \end{pmatrix} a + \begin{pmatrix} -1 \\ 1 \end{pmatrix} b = \begin{pmatrix} -5 \\ 10 \end{pmatrix}$$

In general, we have our data y and a set of basis functions (columns of M) and we want to know how much do the different basis functions contribute to our measured data

This may mean:

- y: measured fMRI time course per voxel
- basis functions ([1 -1 0] etc.): your predicted fMRI time courses for different conditions
- a/b/c: the "betas" for different conditions => see next week's *Introduction to Neuroimaging* lecture

- y: measured reaction times for all stimuli
- basis functions: predictor variables, one value per stimulus (length, frequency…)
- a/b/c: regression coefficients for different conditions

y is a linear combination of the columns of **M.**

$$\mathbf{y} = \begin{pmatrix} y_1 \\ \cdots \\ y_j \\ \cdots \\ y_R \end{pmatrix} = \mathbf{Mx} = \begin{pmatrix} M_{11} & \cdots & M_{1j} & \cdots & M_{1C} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ M_{i1} & \cdots & M_{ij} & \cdots & M_{iC} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ M_{R1} & \cdots & M_{Rj} & \cdots & M_{RC} \end{pmatrix} \begin{pmatrix} x_1 \\ \cdots \\ x_j \\ \cdots \\ x_R \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^{C} x_j * M_{1j} \\ \cdots \\ \sum_{j=1}^{C} x_j * M_{ij} \\ \cdots \\ \sum_{j=1}^{C} x_j * M_{Rj} \end{pmatrix} = \sum_{j=1}^{C} x_j * \begin{pmatrix} M_{1j} \\ \cdots \\ M_{ij} \\ \cdots \\ M_{Rj} \end{pmatrix} = \sum_{j=1}^{C} x_j \mathbf{M}_{.j}$$

$\mathbf{M}_{.j}$ stands for the j-th column of $\mathbf{M}$.

Each column of **M** is weighted by the corresponding element in **x**.

$$a * \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} + b * \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix} + c * \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ -1 & 1 & 1 \\ 0 & -1 & 1 \end{pmatrix} * \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

You could call **M** the “design matrix”.

Basis functions

Orthonormal: Orthogonal and of unit norm/length

For example:
[1 0] and [0 1] are orthonormal basis functions

$$\alpha\begin{bmatrix}1\\0\end{bmatrix}+\beta\begin{bmatrix}0\\1\end{bmatrix}=\begin{pmatrix}1&0\\0&1\end{pmatrix}\begin{pmatrix}\alpha\\\beta\end{pmatrix}=\begin{pmatrix}1\\1\end{pmatrix}$$

$$\Downarrow$$

$$\alpha=\begin{pmatrix}1&0\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}=1$$

$$\beta=\begin{pmatrix}0&1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}=1$$

No “inversion” necessary – just multiply basis functions to your data.

Problem of multiple linear regression

Often basis functions are not orthonormal and for correlated basis functions, the whole system of equations needs to be taken into account

$\Rightarrow$ Matrix inversion is necessary

("partialling out" variables)

"Linearly independent": vectors are not perfectly correlated

"Orthogonal": correlation of vectors is exactly zero

```
% A matrix with linearly independent vectors
x= [2 0]'; y = [-1 1]';
M = [x,y]
det(M)
% with linearly dependent vectors:
y = x*-2;
M = [x,y];
det(M)
```

```
% try to invert the matrix with dependent vectors
inv(M)
% "fix" the collinearity
y = x*-2+ 1e-12;
M = [x,y];
inv(M) % the inverse matrix has huge values
       % - this may amplify errors in the data!
```

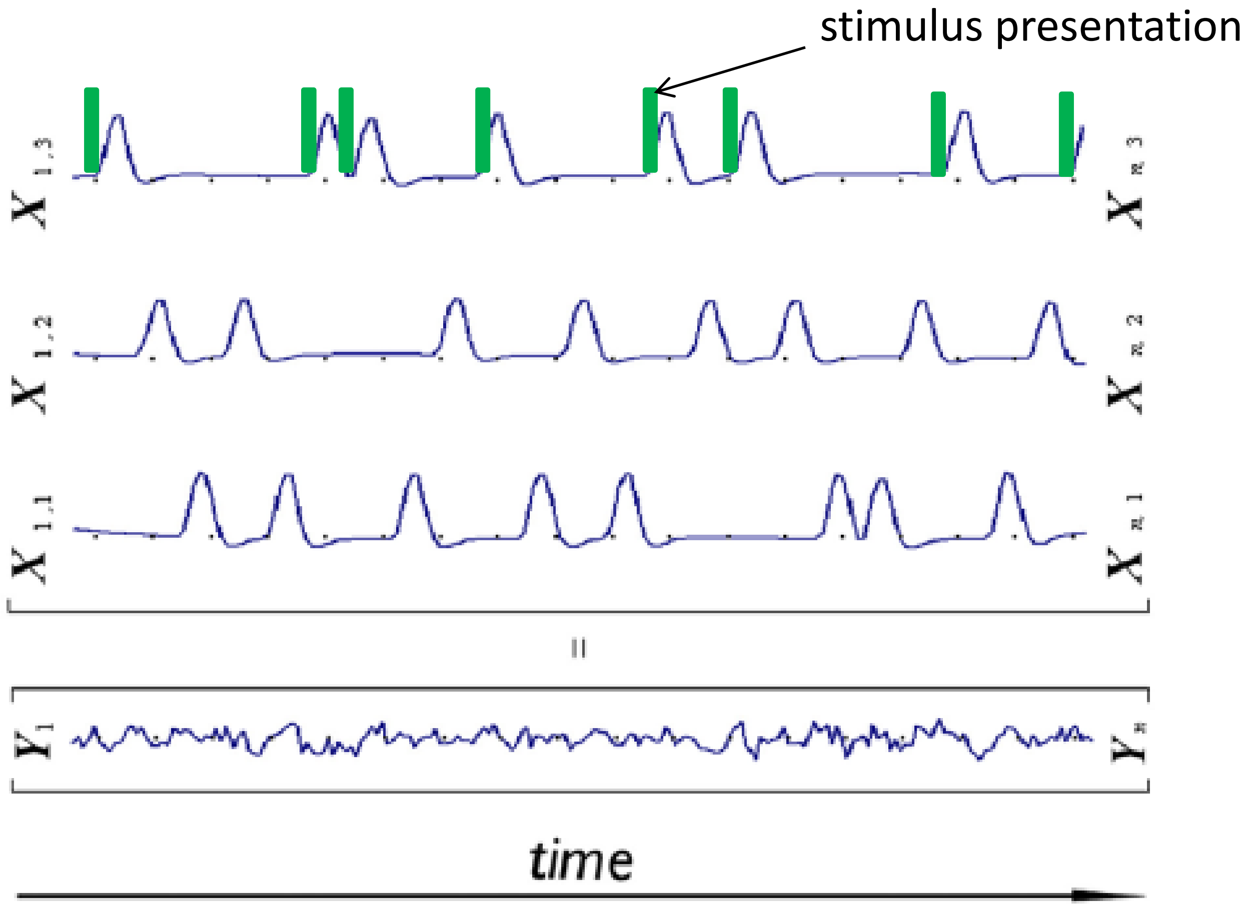
Examples in Neuroscience: fMRI

Predicted time course for event type 1

Predicted time course for event type 2

Predicted time course for event type 3

BOLD time course in one voxel



Measured time series

regression coefficients

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

errors

Design matrix of predictor variables

The choice of the right basis functions depends on the problem and what you know about it

– it's not about the math

"Overdetermined Problem" (e.g. Regression)

$$
\begin{aligned}
1*x_1+1*x_2&=1\\
2*x_1+1*x_2&=-1\\
2*x_1+2*x_2&=2\\
3*x_1+1*x_2&=0\\
3*x_1+2*x_2&=1.5\\
3*x_1+3*x_2&=2.5
\end{aligned}
\qquad
\begin{aligned}
x_1&=?\\
x_2&=?
\end{aligned}
$$

$$
\begin{pmatrix}1\\2\\2\\3\\3\\3\end{pmatrix}*x_1+\begin{pmatrix}1\\1\\2\\1\\2\\3\end{pmatrix}*x_2=\underbrace{\begin{pmatrix}1&1\\2&1\\2&2\\3&1\\3&2\\3&3\end{pmatrix}}_{\mathbf{M}}\begin{pmatrix}x_1\\x_2\end{pmatrix}=\mathbf{Mx}=\underbrace{\begin{pmatrix}1\\-1\\2\\0\\1.5\\2.5\end{pmatrix}}_{\mathbf{d}}=\mathbf{d}
$$

M is not invertible, there is no unique solution for **x**.

We can find the **x** that minimises the least-squares error: $\left\|\mathbf{Mx}-\mathbf{d}\right\|^2=\min$

The matrix that provides this least-squares solution is the "pseudoinverse" of **M: M^{-}**
(in Matlab: "pinv")

$$
\mathbf{M}^{-}=\begin{matrix}-0.0526 & 0.1579 & -0.1053 & 0.3684 & 0.1053 & -0.1579\\ 0.1158 & -0.1474 & 0.2316 & -0.4105 & -0.0316 & 0.3474\end{matrix}
$$

Generalisation of inverse matrix: **M*M^{-}*M = M and M^{-}*M*M^{-} = M^{-}**

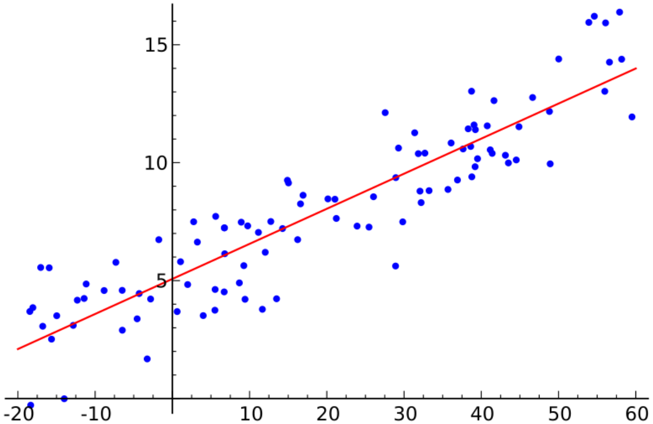
“Overdetermined Problem” (e.g. Regression)

If **Mx = d** is an overdetermined problem (more data than unknowns), then

$$\mathbf{M}^{-} = (\mathbf{M}^{T}\mathbf{M})^{-1}\mathbf{M}^{T}$$

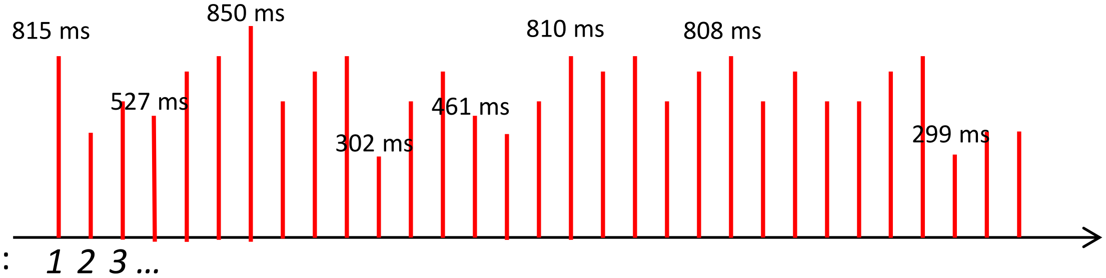
Is the unique (minimum norm) least squares solution

```
M = [1 1;2 3;-3 1];
size(M)% more rows than columns
% compute the inverse of the design matrix with pinv
inv_pinv = pinv(M);
% compute the pseudo inverse yourself
inv_ls = inv(M'*M)*M';
% compute the solution
sol_pinv = pinv(M)*[1 -1 3]';
sol_ls = inv_ls*[1 -1 3]';
```

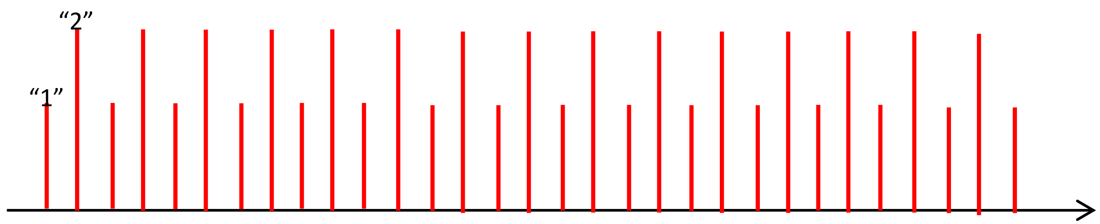


Behavioral RT experiment

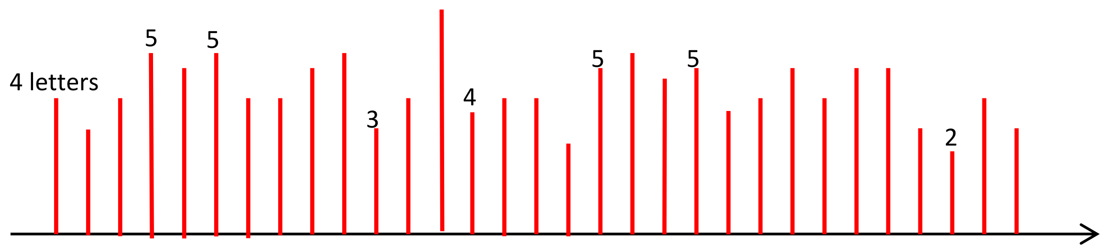
Which variables can explain these data?



Repetition



Word length?



RTs

regression coefficients

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

errors

Design matrix of predictor variables

Example

```
% Generate data for a behavioural experiment (forward model)
w_length = [3 2 5 4 4 4 5 2 5 3 4 6]';
rep = [1 2 1 2 1 2 1 2 1 2 1 2]';
offset = ones(size(rep));
RTs = offset*480 + w_length*10 +  rep*30 + randn(size(rep))*8;
% plot your data
figure; plot(RTs,'*')
% Create the design matrix
M = [offset, w_length, rep];
% get the solution
b = pinv(M)*RTs
% Check how solution predicts the data
figure; RTs_pred = b(1)*offset + b(2)*w_length + b(3)*rep;
plot(RTs_pred,'*')
% compare with “ground truth”
hold on; RTs_real = offset*480 + w_length*10 +  rep*30;
plot(RTs_real,'r*')
% plot difference between measured and predicted data
figure; plot(RTs - RTs_pred)
```

Underdetermined Problem (e.g. EEG/MEG inverse problem)

$$1*x_1 + 1*x_2 + 1*x_3 = 1$$
$$1*x_1 + 2*x_2 + 3*x_3 = -1$$

$$x_1 = ?$$
$$x_2 = ?$$
$$x_3 = ?$$

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix} * x_1 + \begin{pmatrix} 1 \\ 2 \end{pmatrix} * x_2 + \begin{pmatrix} 1 \\ 3 \end{pmatrix} * x_3 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \mathbf{Mx} = \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \mathbf{d}$$

M is not invertible, there is no unique solution for **x**.

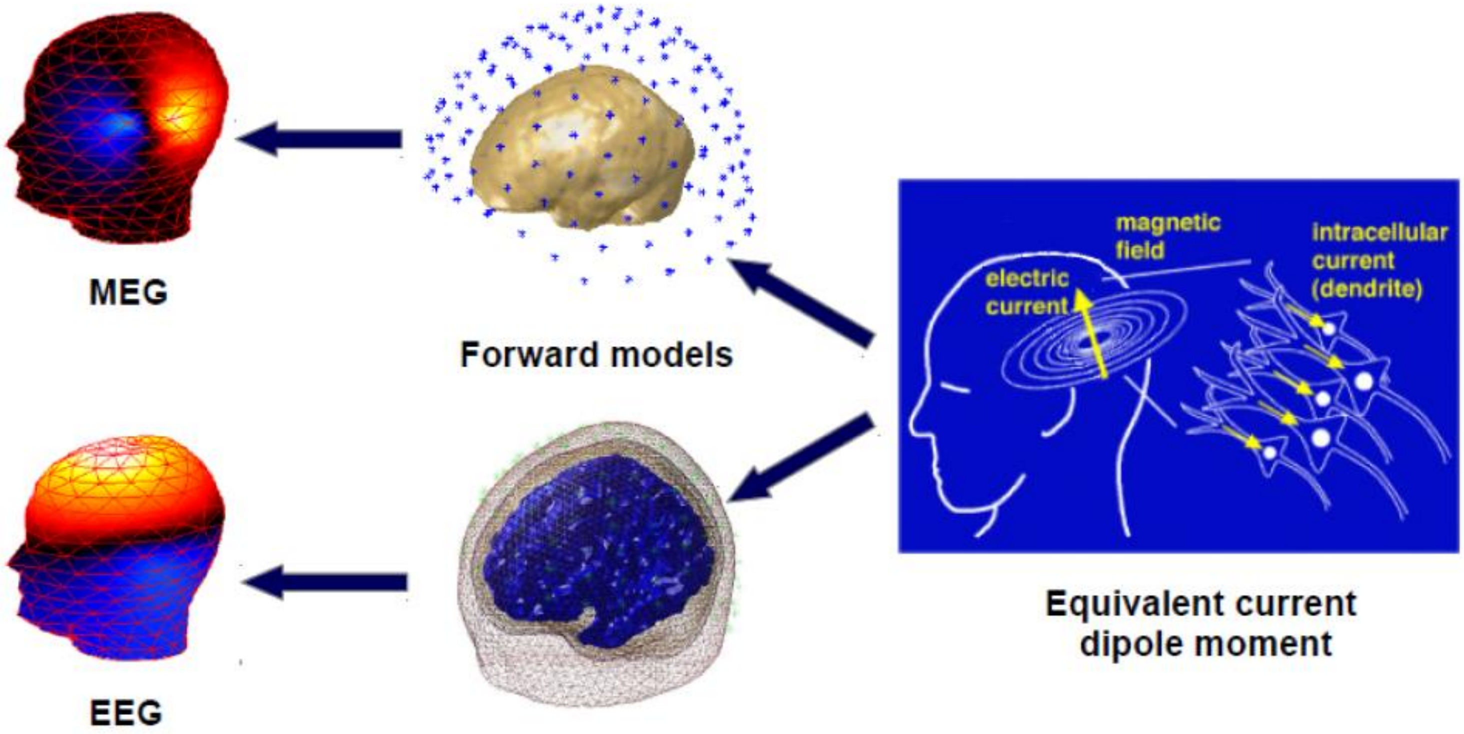
$$\mathbf{Mx} = \mathbf{d}$$

We can find the **x** that minimises the “norm” of the solution:

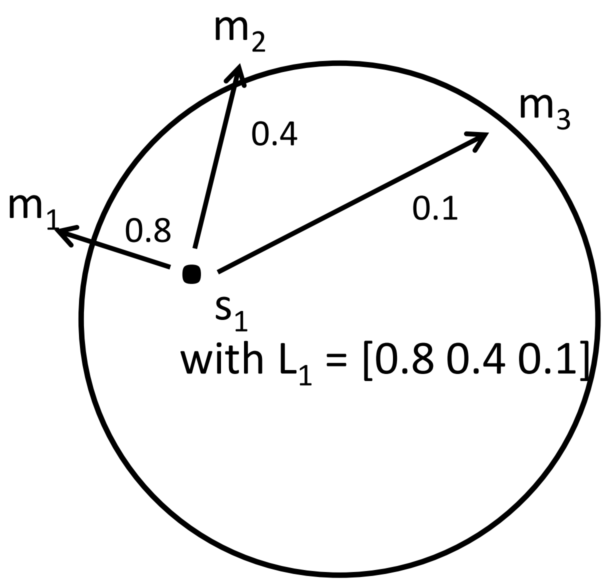
$$\|\mathbf{x}\|^2 = x_1^2 + x_2^2 + x_3^2$$

Again, the solution is given by the “pseudoinverse” of **M: M**$^{-}$ (in Matlab: “pinv”)

Example: M/EEG source reconstruction

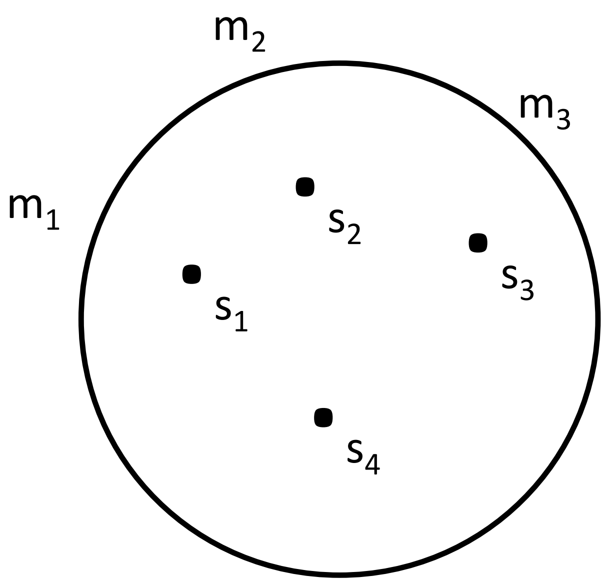


Example: M/EEG source reconstruction



- y: measured topography at a particular latency
- basis functions: EEG/MEG topographies for point sources of unit strength (dipoles), also known as leadfields
- a/b/c: source strengths for those point sources

Example: M/EEG source reconstruction



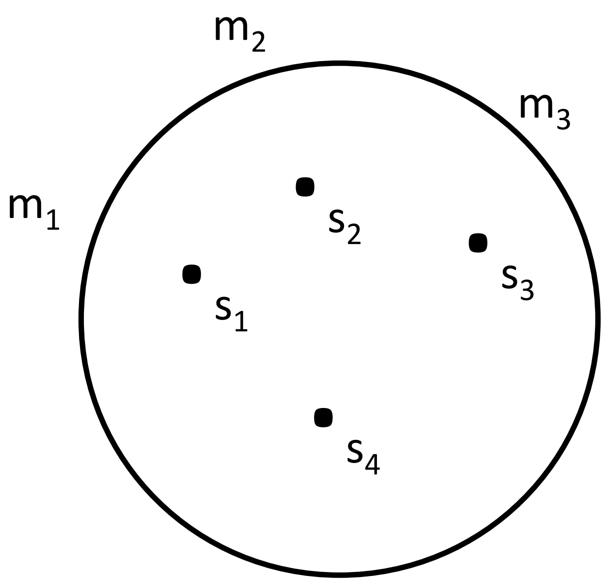
Let's assume we have four cortical sources s_1, s_2, s_3 and s_4 with L_1 = [0.8 0.4 0.1], L_2 = [0.3 0.9 0.2], L_3 = [0.1 0.3 0.7] and L_4 = [0.2 0.1 0.1]

We want to solve the m = Ls

(1) construct the leadfield matrix L
(2) use pinv to solve the GLM for m = [1 2 3]
(3) use the backslash operator to solve the GLM
(4) calculate the norm for both solutions

You just computed your very own Minimum Norm inverse solution!

Example: M/EEG source reconstruction



Let's assume we have four cortical sources s_1, s_2, s_3 and s_4 with L_1 = [0.8 0.4 0.1], L_2 = [0.3 0.9 0.2], L_3 = [0.1 0.3 0.7] and L_4 = [0.2 0.1 0.1]

```
L_1 = [0.8 0.4 0.1];
L_2 = [0.3 0.9 0.2];
L_3 = [0.1 0.3 0.7];
L_4 = [0.2 0.1 0.1];

% Construct design matrix
M = [L_1',L_2',L_3',L_4'];
size(M)
m = [1 2 3];
s = pinv(M)*m'
s_ml = M\m'
```

Thank you!