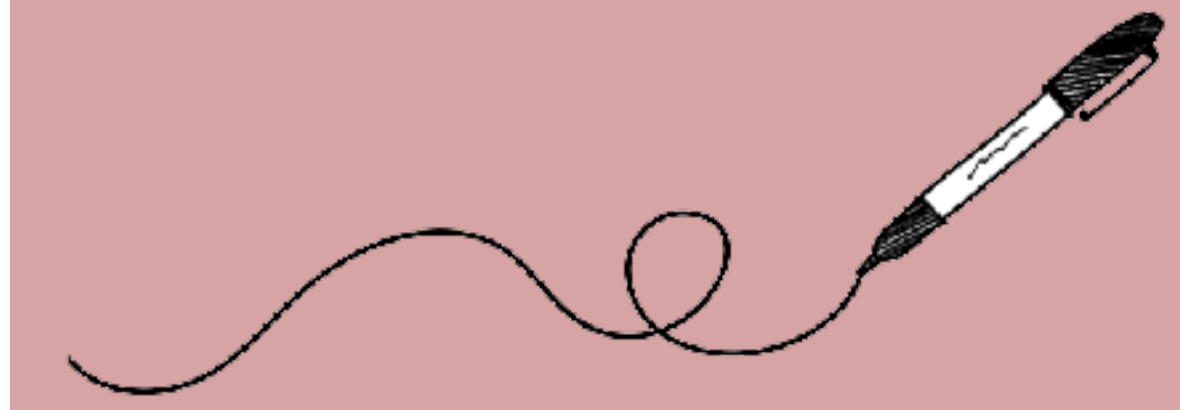
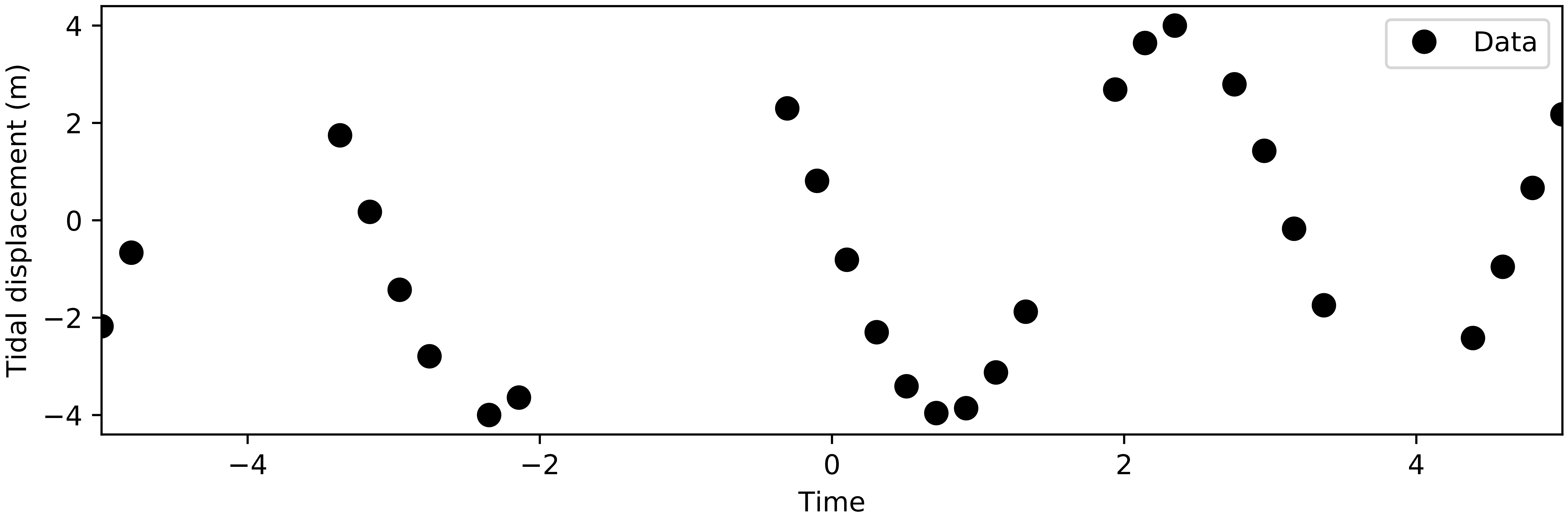
Underdetermined OLS: Interpolation example

How to best interpolate these data?





Underdetermined OLS: Interpolation example

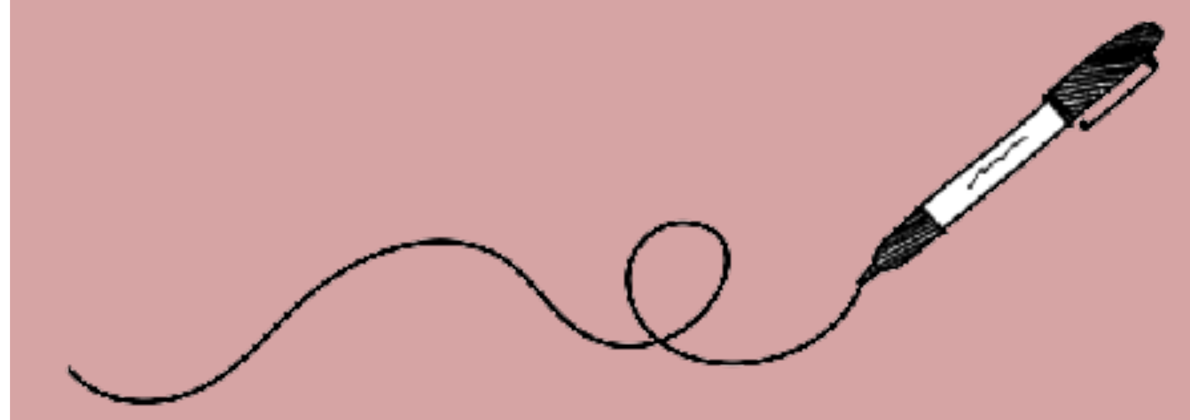
Take some time how this problem can be transferred to a

Gm = d

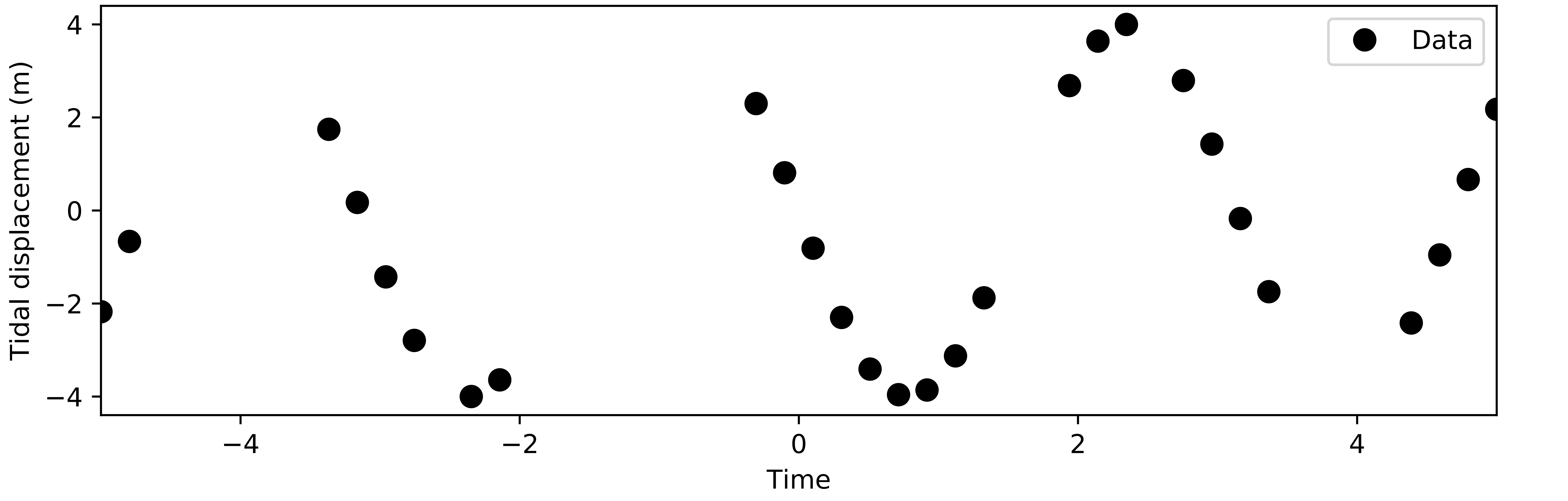
form.

What would be the model parameters ?

What are the dimensions involved?



Thanks Wikipedia.



Underdetermined OLS: Interpolation example

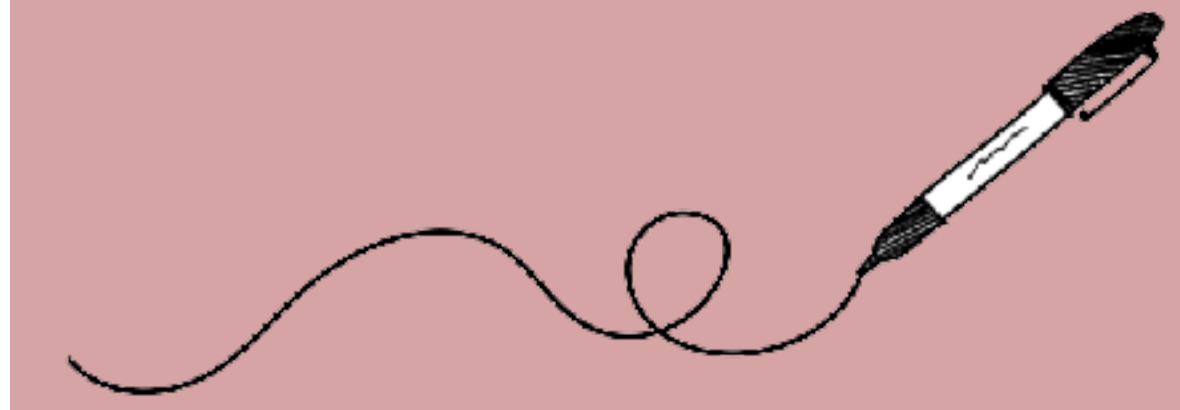
Take some time how this problem can be transferred to a

Gm = d

form.

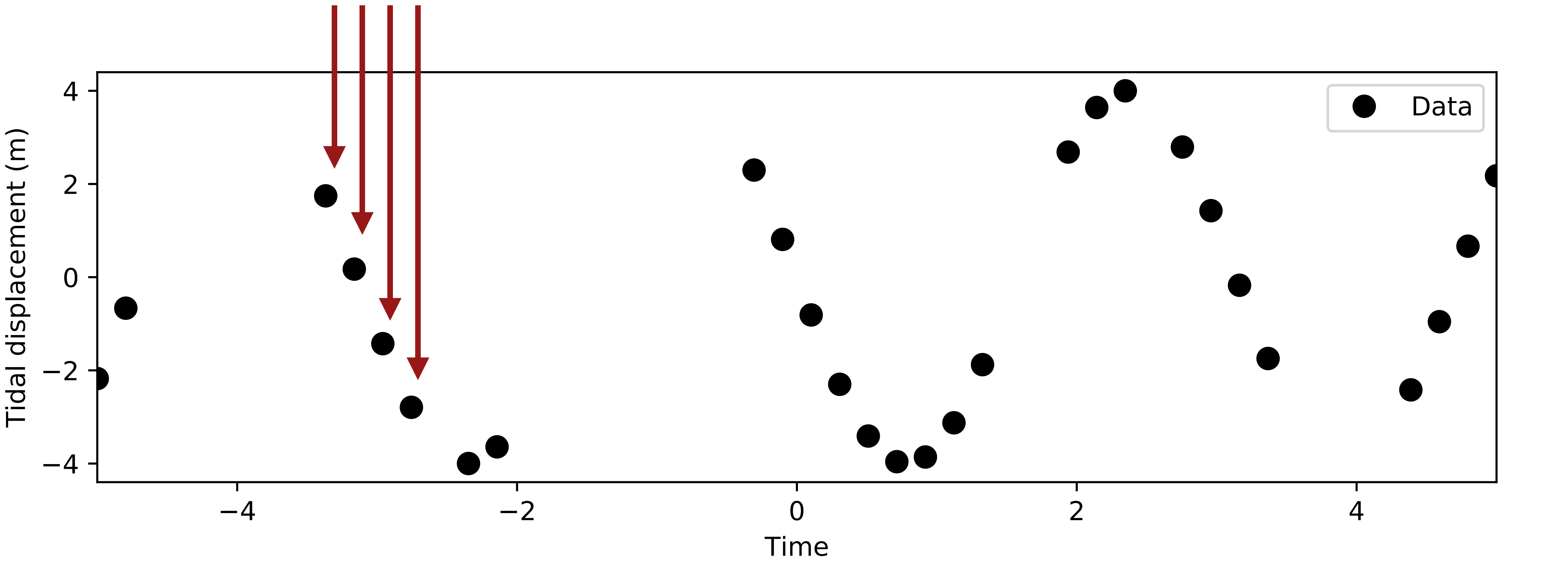
What would be the model parameters ?

What are the dimensions involved?



Thanks Wikipedia.

Data can be collected in data vector **d** with dimensions N_{obs} x 1.



Underdetermined OLS: Interpolation example

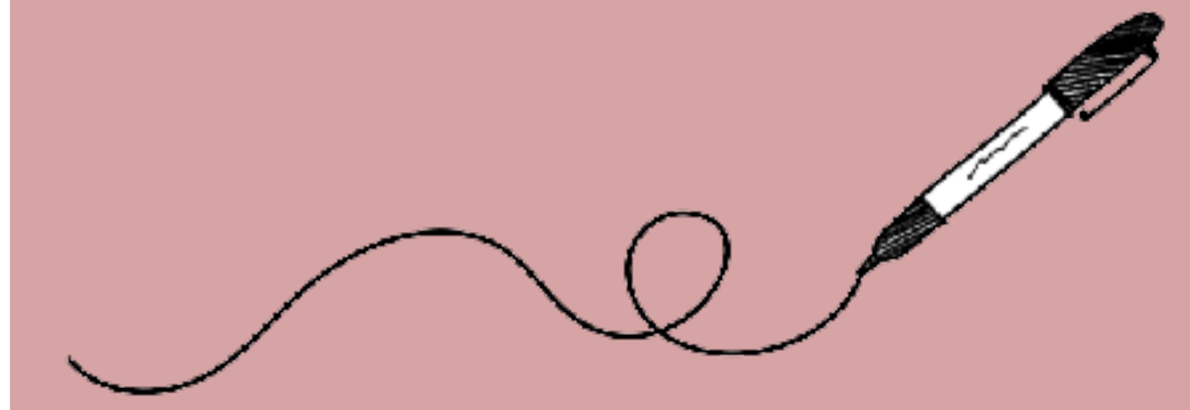
Take some time how this problem can be transferred to a

Gm = d

form.

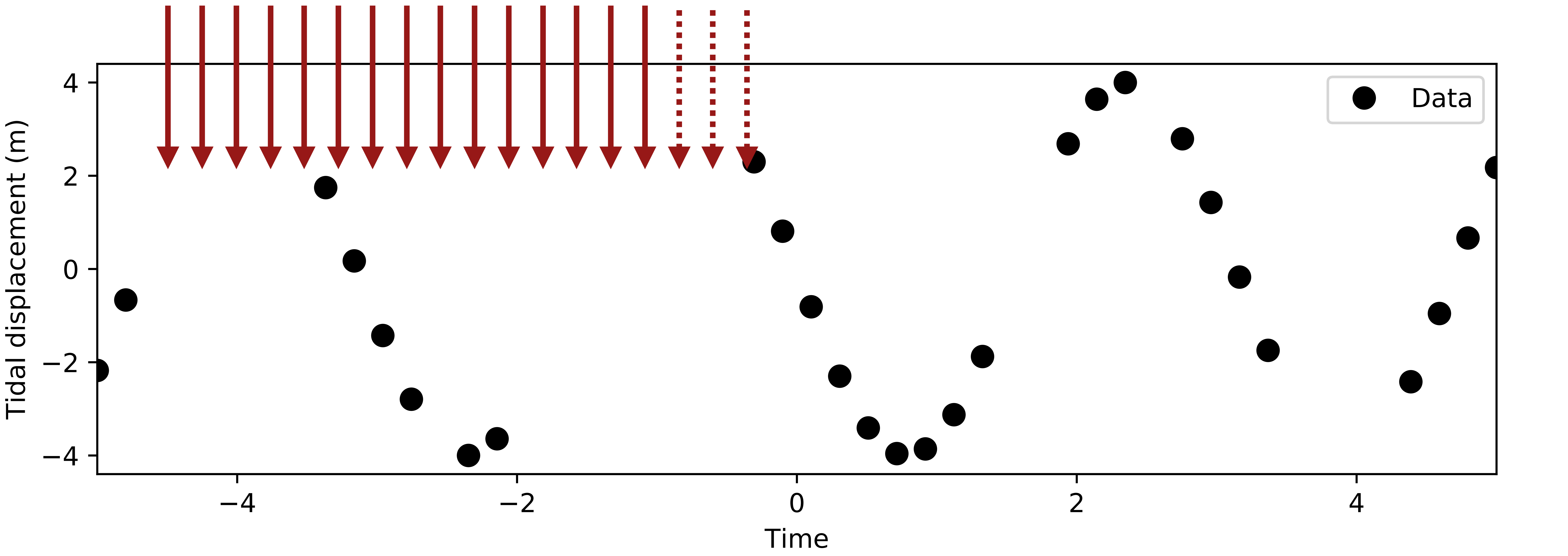
What would be the model parameters ?

What are the dimensions involved?



Thanks Wikipedia.

Model parameter vector **m** (N_p x 1) predicts tidal displacement at time x_k.



Underdetermined OLS: Interpolation example

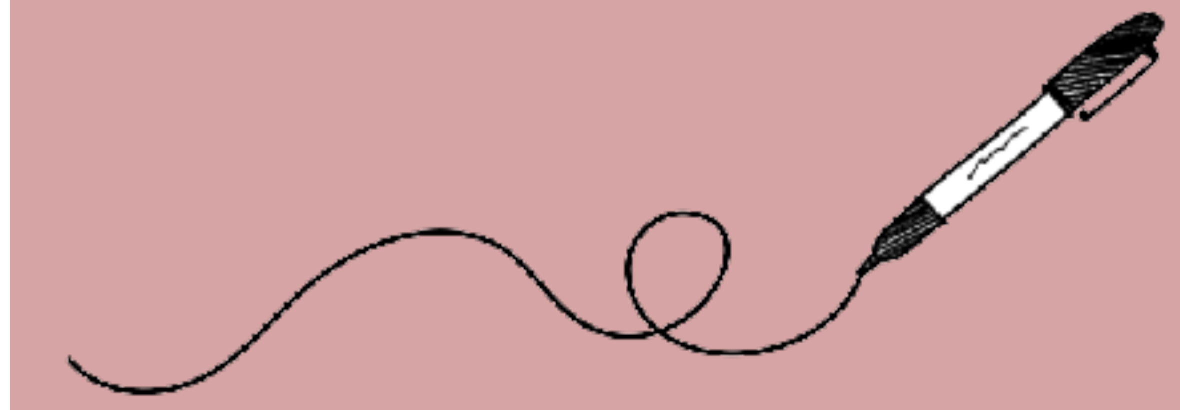
Take some time how this problem can be transferred to a

Gm = d

form.

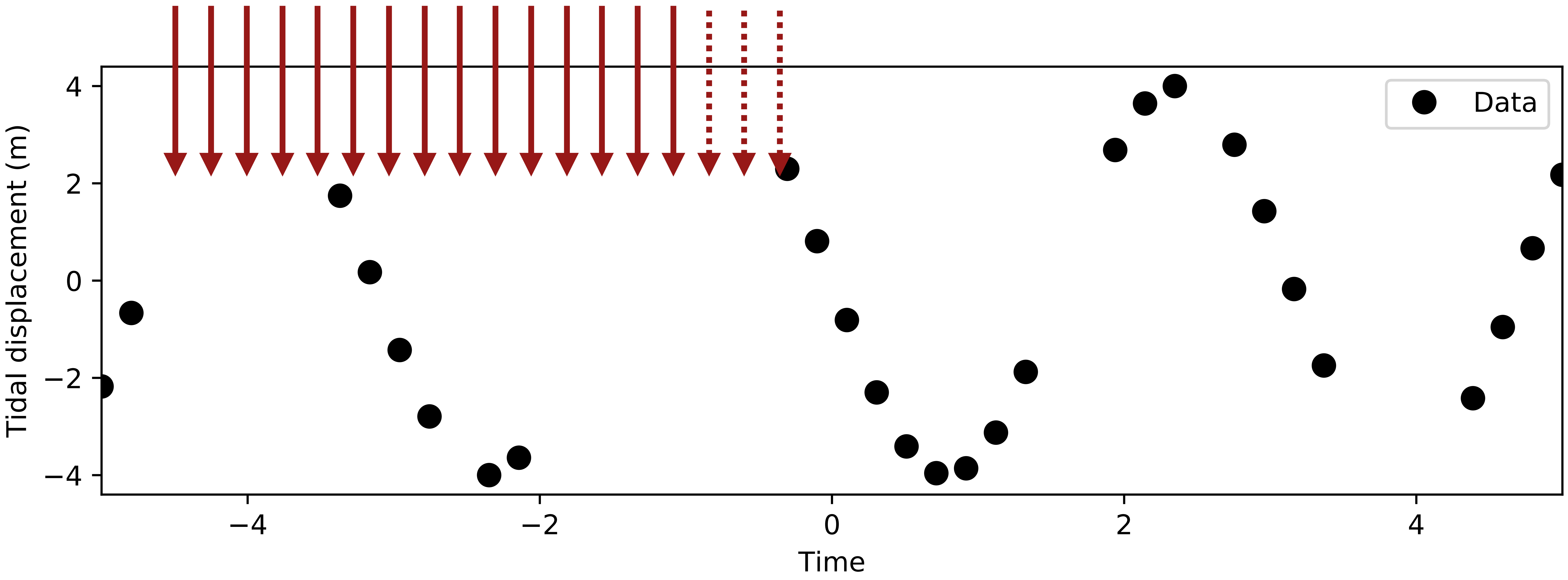
What would be the model parameters ?

What are the dimensions involved?



Thanks Wikipedia.

Model parameter vector **m** (N_p x 1) predicts tidal displacement at time x_k.



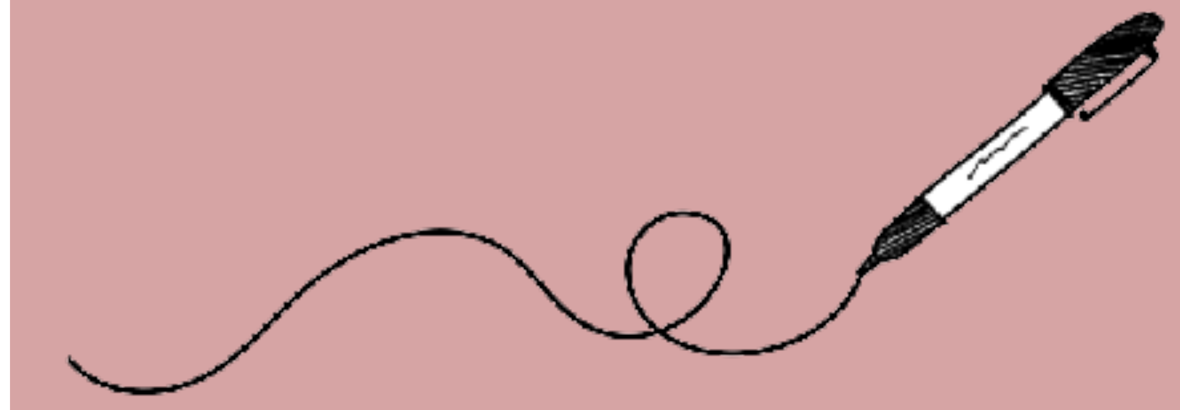
If **m** (N_p x 1) and **d** (N_{obs} x 1), then **G** (N_o x N_p). Also in this case N_{obs} > N_{obs}.

What does G look like ? Think Think Think.

Underdetermined OLS: Interpolation example

This is the interpolation problem formulated as an inverse problem

Gm=d



For didactic reasons I chose to show index as time, not position in vector.

$$\underbrace{\begin{pmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ \dots \\ \dots \\ \dots \\ \dots \\ \dots \\ m_{N_p} \end{pmatrix}}_{N_p \times 1} = \underbrace{\begin{pmatrix} d_1 \\ d_2 \\ d_5 \\ d_7 \\ \dots \\ \dots \\ \dots \\ d_{N_o} \end{pmatrix}}_{N_o \times 1}$$

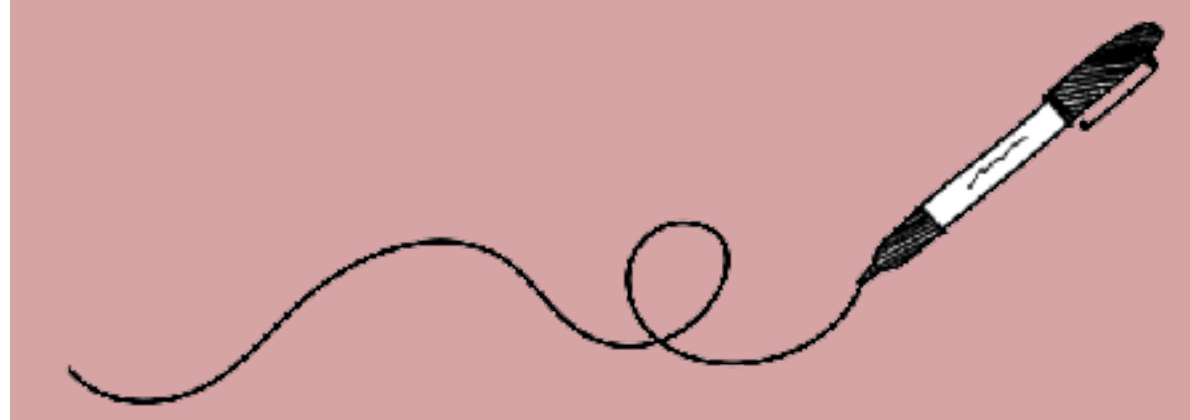
Underdetermined OLS: Interpolation example

This is the interpolation problem formulated as an inverse problem.

What do you expect for

$(G^TG)^{-1}$

?



For didactic reasons I chose to show index as time, not position in vector.

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix}}_{N_o \times N_p} \underbrace{\begin{pmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ \dots \\ \dots \\ \dots \\ \dots \\ \dots \\ m_{N_p} \end{pmatrix}}_{N_p \times 1} = \underbrace{\begin{pmatrix} d_1 \\ d_2 \\ d_5 \\ d_7 \\ \dots \\ \dots \\ \dots \\ d_{N_o} \end{pmatrix}}_{N_o \times 1}$$

Underdetermined OLS: Interpolation example

This is the interpolation problem formulated as an inverse problem.

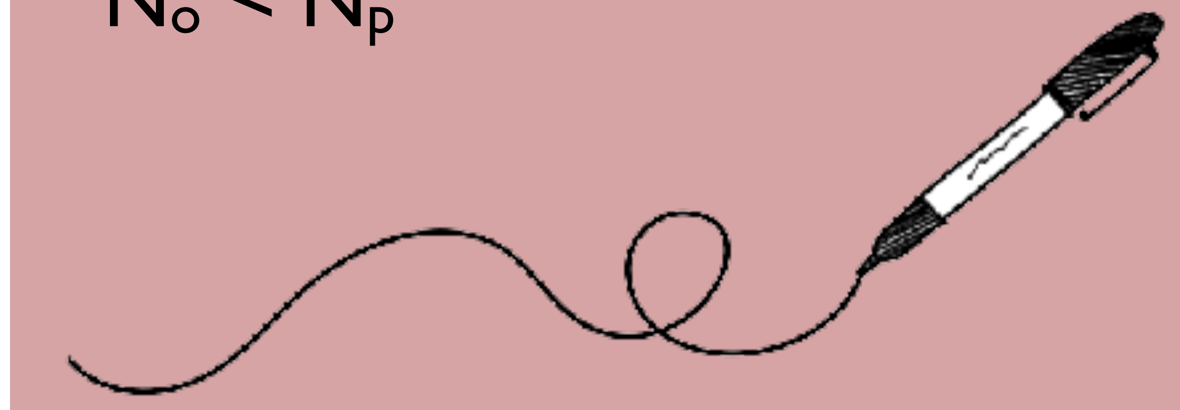
What do you expect for

$(G^TG)^{-1}$

?

It will not exist. This is our first underdetermined problem where

$N_o < N_p$

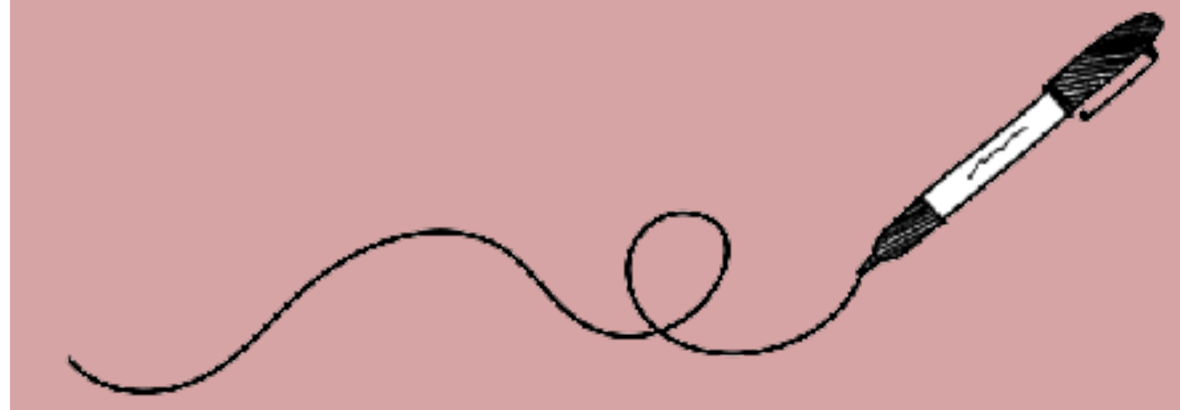


For didactic reasons I chose to show index as time, not position in vector.

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix}}_{N_o \times N_p} \underbrace{\begin{pmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ \dots \\ \dots \\ \dots \\ \dots \\ \dots \\ m_{N_p} \end{pmatrix}}_{N_p \times 1} = \underbrace{\begin{pmatrix} d_1 \\ d_2 \\ d_5 \\ d_7 \\ \dots \\ \dots \\ \dots \\ d_{N_o} \end{pmatrix}}_{N_o \times 1}$$

Underdetermined OLS: Interpolation example

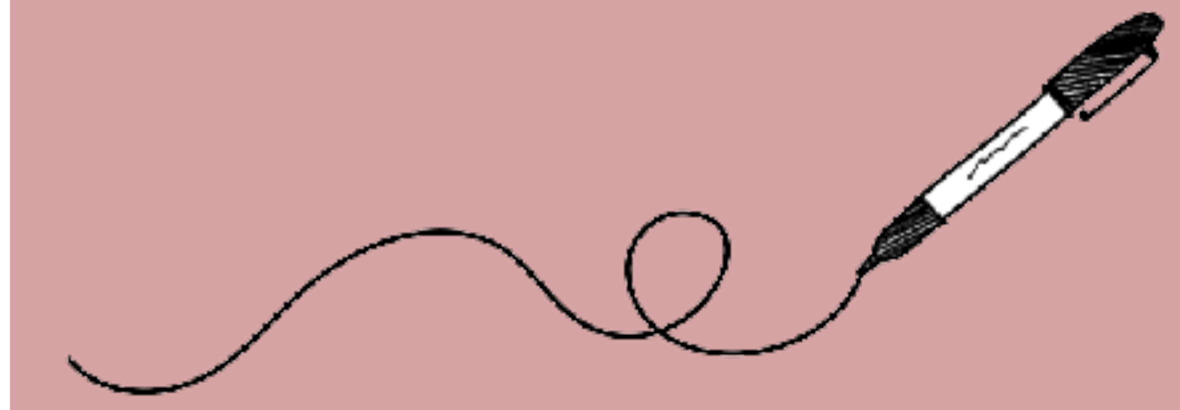
Adding a priori knowledge is an important thing. Not only for interpolation and underdetermined problems. It occurs often. Pay attention.

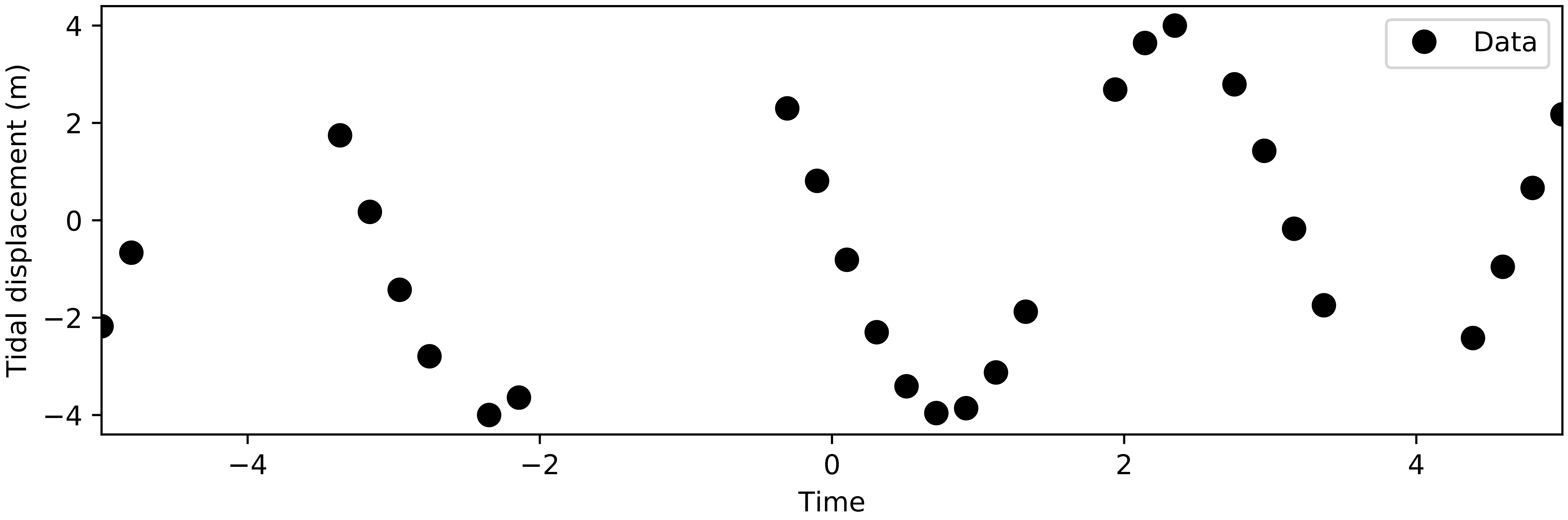


We need to find some way to add additional information to the problem. Otherwise we fail. Which constraints can we add ? Discuss.

Underdetermined OLS: Interpolation example

Adding a priori knowledge is an important thing. Not only for interpolation and underdetermined problems. It occurs often. Pay attention.



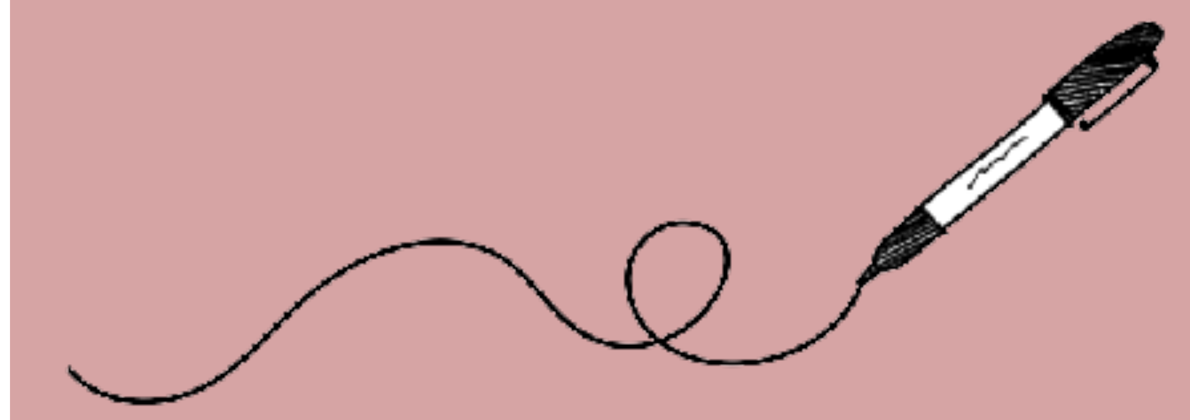


Towards a smoothness constraint.

We prescribe that the interpolated function should be smooth.

Smooth means that the *second* derivative should be small.

You should understand why this is the case.



We know this representation of the derivative (naive forward differencing).
This allows us to find a representation of the second derivative.

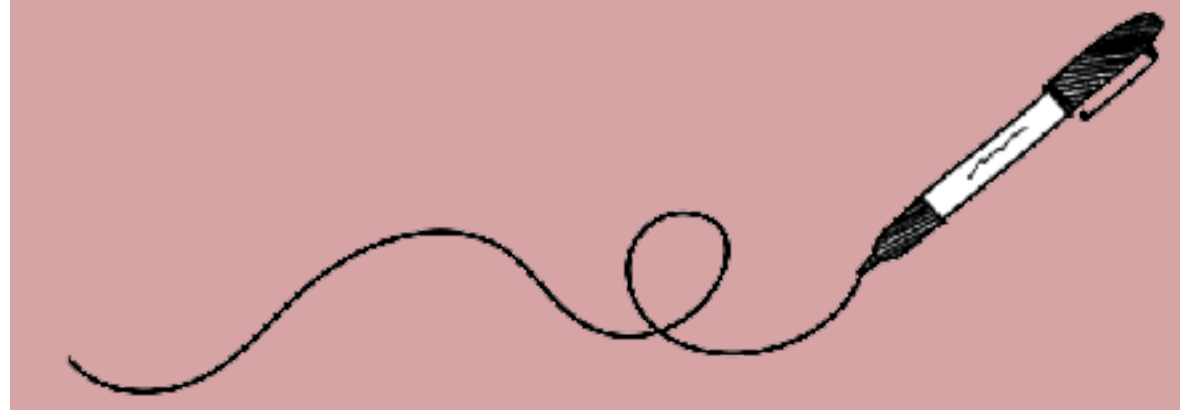
$$\mathbf{D} = \frac{1}{\Delta x}\begin{bmatrix} -1 & 1 & 0 & 0 & \dots \\ 0 & -1 & 1 & 0 & \dots \\ 0 & 0 & -1 & 1 & \dots \\ .. & .. & .. & .. & \dots \end{bmatrix}$$

$$\mathbf{DD} = \mathbf{A} = \frac{1}{\Delta x^2}\begin{bmatrix} 1 & -2 & 1 & 0 & 0 & \dots \\ 0 & 1 & -2 & 1 & 0 & \dots \\ 0 & 0 & 1 & -2 & 1 & \dots \\ .. & .. & .. & .. & \dots & \end{bmatrix}$$

Towards a smoothness constraint.

As with the forward differencing, there will be problems at the boundaries where the second derivative cannot be adequately determined.

Here we use flatness, meaning, we try to make the first derivative at the boundaries small.



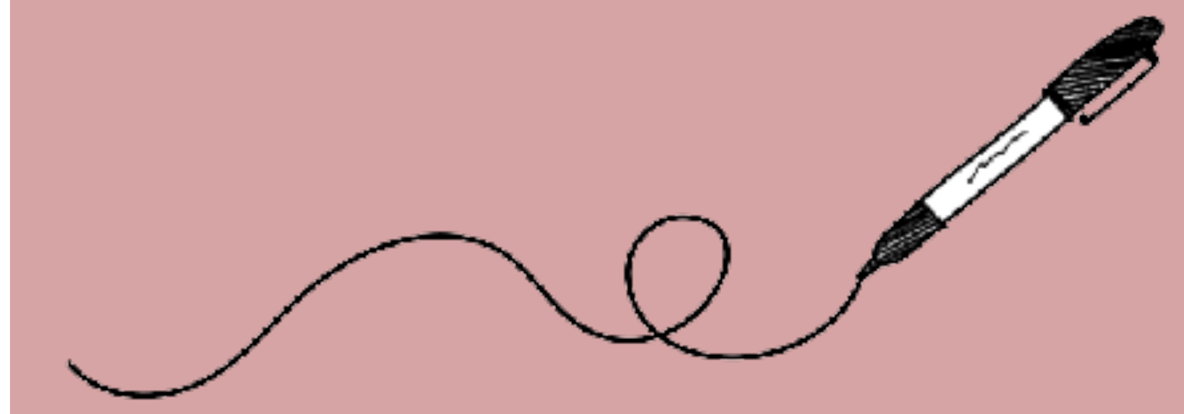
Flatness.

$$\mathbf{H} = \frac{1}{(\Delta x)^2}\begin{bmatrix} -\Delta x & \Delta x & 0 & 0 & 0 & \dots & 0 \\ 1 & -2 & 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & -2 & 1 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & 1 & -2 & 1 & 0 \\ 0 & \dots & 0 & 0 & 1 & -2 & 1 \\ 0 & \dots & 0 & 0 & 0 & -\Delta x & \Delta x \end{bmatrix}$$

Flatness.

Towards a smoothness constraint.

Now we have two conditions that should be fulfilled. First, model-data misfit should be small, and model should be smooth. Nice.

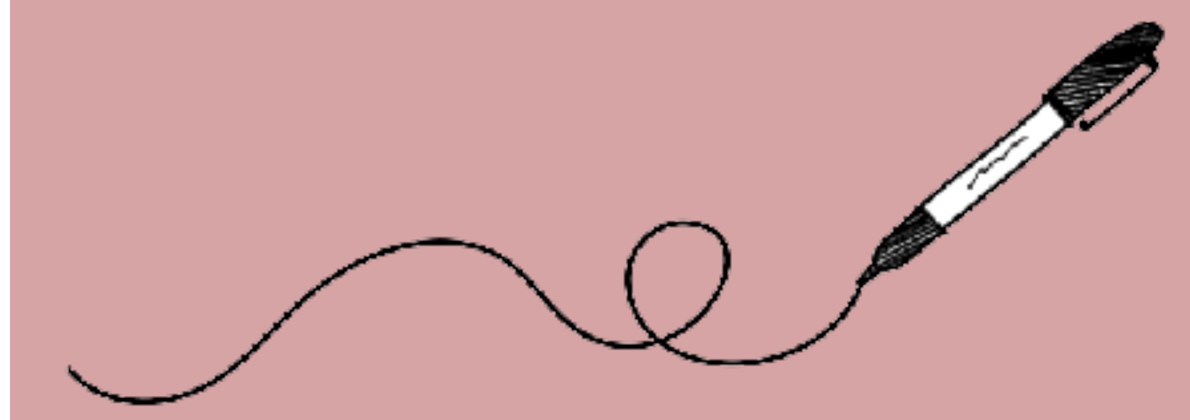


$$||\mathbf{d}^{obs} - \mathbf{G}\mathbf{m}||^2 + \lambda||\mathbf{A}\mathbf{m}||^2$$

Towards a smoothness constraint.

Now we have two conditions that should be fulfilled. First, model-data misfit should be small, and model should be smooth.

Minimization of this term is analog to minimising our previous cost function. Skipped here.



$$||\mathbf{d}^{obs} - \mathbf{G}\mathbf{m}||^2 + \lambda||\mathbf{A}\mathbf{m}||^2$$

$$\frac{\partial}{\partial m_i}||\mathbf{d}^{obs} - \mathbf{G}\mathbf{m}||_2^2 + \lambda||\mathbf{A}\mathbf{m}||_2^2 = 0$$

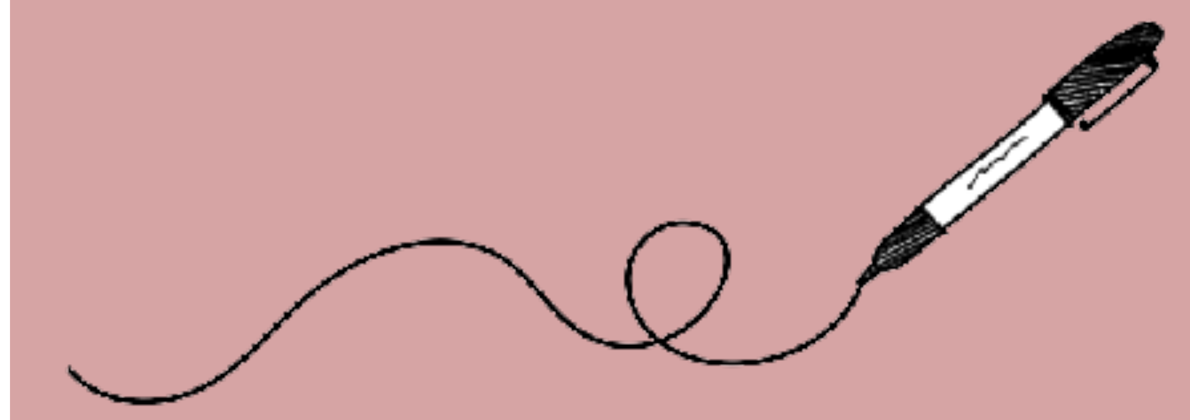
$$\mathbf{m} = (\mathbf{G}^T\mathbf{G} + \lambda\mathbf{A}^T\mathbf{A})^{-1}\mathbf{G}^T\mathbf{d}^{obs}$$

Yes. This we know how to handle.

More general: Regularization

Modifying the cost-function with additional constraints is known as *“Regularization”*. This is quite powerful and can help in many other ways (e.g., *the numerical problems of fitting higher order polynomials in Ex2*).

The lambda is called regularisation parameter, and allows (user-defined) tuning of regularisation vs. Model-Data misfit.



$$||\mathbf{d}^{obs} - \mathbf{Gm}||^2 + \lambda||\mathbf{Am}||^2$$

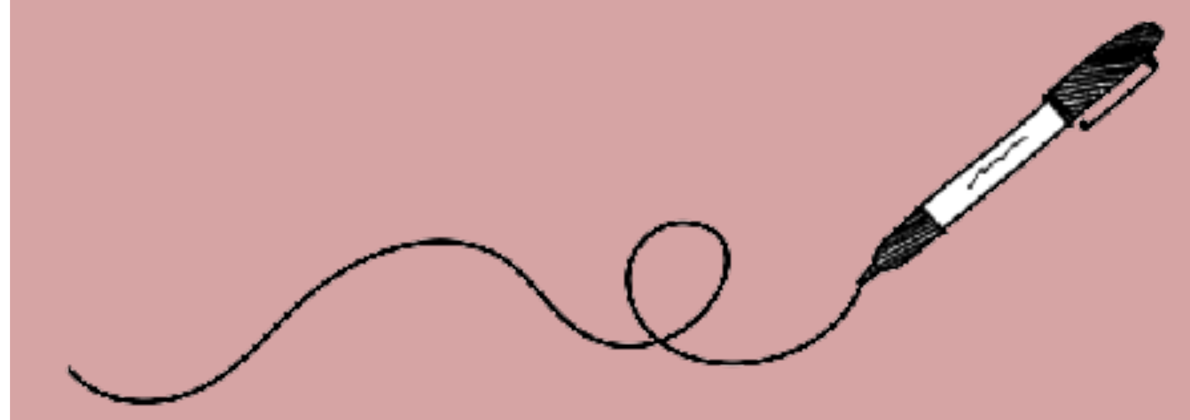
$$\frac{\partial}{\partial m_i}||\mathbf{d}^{obs} - \mathbf{Gm}||_2^2 + \lambda||\mathbf{Am}||_2^2 = 0$$

$$\mathbf{m} = (\mathbf{G}^T\mathbf{G} + \lambda\mathbf{A}^T\mathbf{A})^{-1}\mathbf{G}^T\mathbf{d}^{obs}$$

Yes. This we know how to handle.

More general: Regularization staying close to initial guess

This regularisation is popular if you have *some idea* of what the solution can be. This idea can be very vague, but even things like *density is positive* can be very useful.



$$||\mathbf{Gm} - \mathbf{d}||^2 + \lambda||\mathbf{m} - \mathbf{m_0}||^2 \qquad \lambda > 0 \in \mathbb{R}$$

Some guess of the solution. Almost anything in the right direction will help.

$$\mathbf{m}^{\mathbf{est}} = \mathbf{m_0} + \left(\mathbf{G}^{\mathbf{T}}\mathbf{G} + \lambda \ \mathbb{1} \ \right)^{-1} \mathbf{G}^{\mathbf{T}} \left[\mathbf{d} - \mathbf{G}\mathbf{m_0}\right]$$

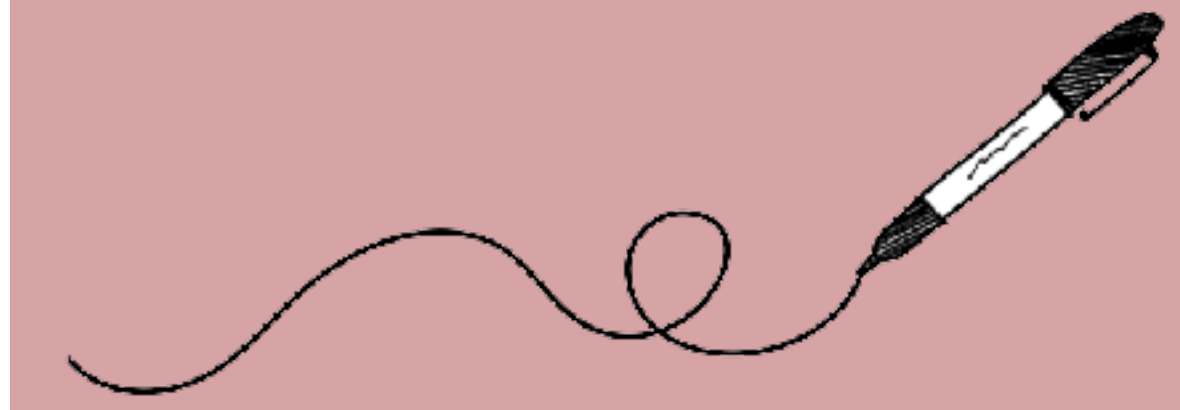
Yes. This we know how to handle.

More general: Regularization staying minimum length

This special case of an initial guess:

$$\mathbf{m_0} = \mathbf{0}$$

Is called a *minimum length regularisation*, or *damped least-squares*. Strictly speaking it works if model parameters are small. It is used a lot for neural networks.



$$||\mathbf{Gm} - \mathbf{d}||^2 + \lambda||\mathbf{m} - \mathbf{m_0}||^2 \qquad \lambda > 0 \in \mathbb{R}$$

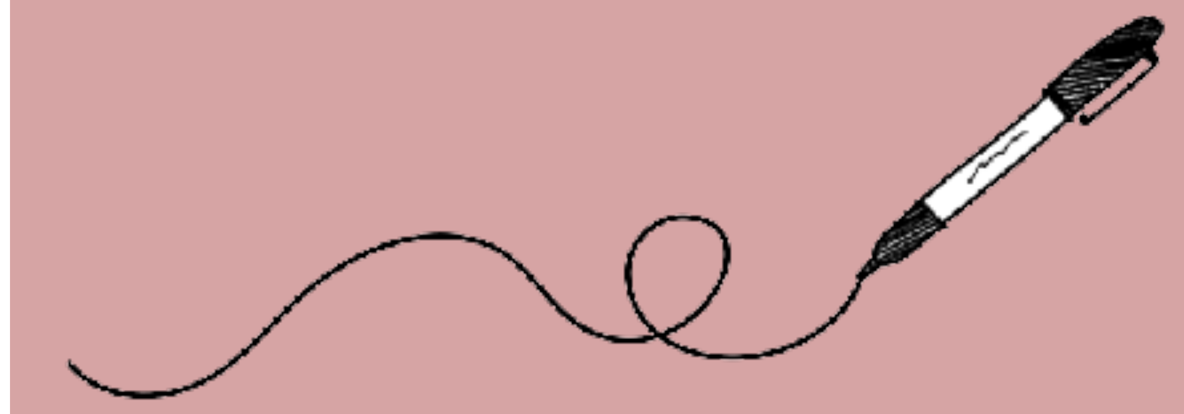
Special case $\mathbf{m_0} = \mathbf{0}$

$$\mathbf{m}^{\mathbf{est}} = \mathbf{m_0} + \left(\mathbf{G}^{\mathbf{T}}\mathbf{G} + \lambda\ \mathbb{1}\ \right)^{-1}\mathbf{G}^{\mathbf{T}}\ \mathbf{d}$$

This is called diagonal loading. The minimum length regularisation is used extensively in neural networks. Pythonians, try this one to improve fitting in Ex 3.

Summary Regularisation

Memorize this.



- Regularisation is a rigorous way to include a-priori knowledge *which virtually always exists.*
- Regularisation is required for underdetermined problems, but also helps elsewhere.
- Regularisation results in a tuning parameter which you need to choose. It balances the weight between the model-data misfit and the regularisation term.
- Regularisation is computationally comparatively easy b/c it requires the same methods as OLS.
- Regularisation is used to avoid overfitting dealt with later.