

# CS 189/289

Today's lecture:

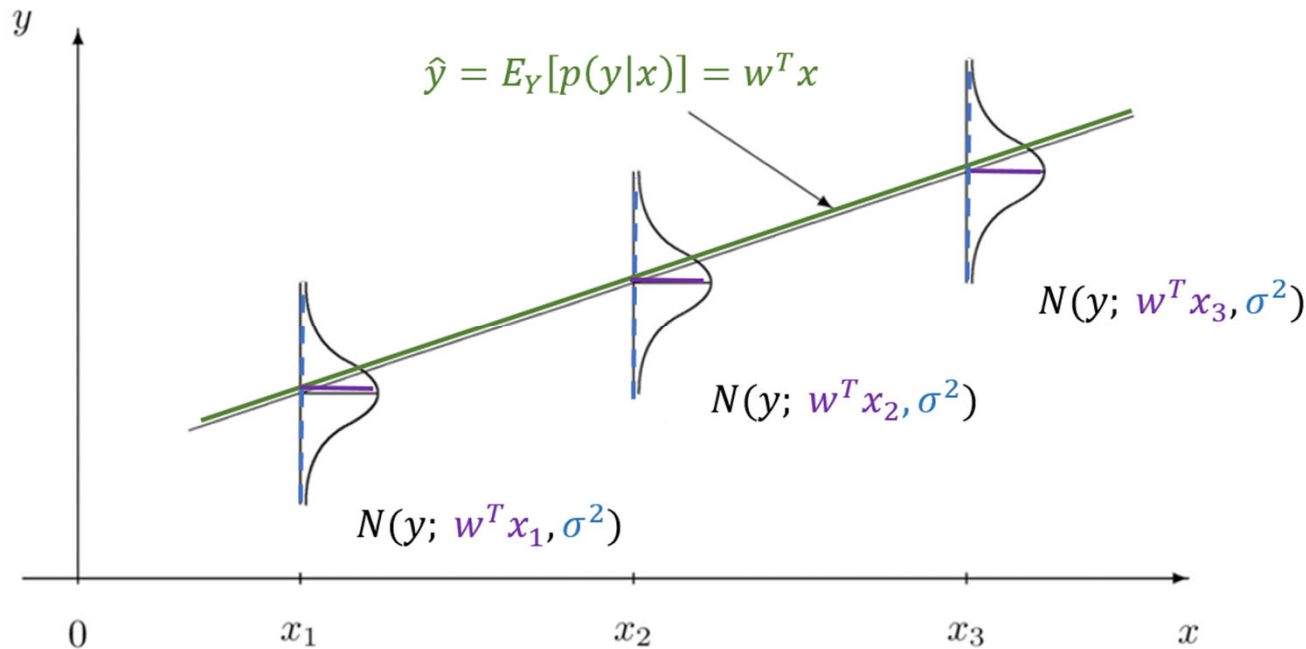
Linear regression part II (regularization)

Assigned reading:

- 1.25, 1.26 (regularization, model selection)
- 2.1.3 (Bayes Theorem)
- 2.6 (Bayesian modeling)
- 4.1.6 (Regularization for linear regression)
- 9.2.2 (Lasso regularization)

# Reloading last lecture

Gaussian linear regression,  $p(y|x) = N(y|w^T x, \sigma^2)$ .



- $\theta_{MLE} = (w_{MLE}, \sigma_{MLE}^2) = \arg \max_{(w, \sigma^2)} \log p(D|\theta)$
- $\mathcal{L}_w = (y - Aw)^T (y - Aw)$ , set  $\frac{\partial \mathcal{L}}{\partial w} = 0$
- $A^T y = A^T A w$  ( $A \in \mathbb{R}^{N \times d}$ )
- $w_{MLE} = (A^T A)^{-1} A^T y$ ,  $\sigma_{MLE}^2 = \frac{1}{N} \sum_i (y_i - w^T x)^2$

careful!

- Not invertible if columns (features) in  $A$  are linearly dependent.
- Automatically happens when  $d > N$ , in which case,  $y = Aw$  exactly.

- When not invertible, there are  $\infty$  many equally good solutions for  $w_{MLE}$ .
- Called *underdetermined* linear regression.

$$\begin{pmatrix} x_1^T \\ \vdots \\ x_n^T \end{pmatrix} = A \in \mathbb{R}^{n \times d}$$

# Fixing underdetermined models

Two main categories of fixes:

1. Remove features until the problem is well-behaved ("feature selection").
2. Leave the features as they are, but *add constraints to the system* to "tighten it up" (aka "regularization").

# Intuition of why $\infty$ # of $\mathbf{w}_{MLE}$ solutions

- Suppose we have 2 *linearly dependent features* in the training data such that  $\alpha \mathbf{x}_1 = \mathbf{x}_2$ .
- Suppose we found one MLE solution,  $\hat{\mathbf{w}}$ .
- Then for any training data point,  $\hat{\mathbf{y}} = \mathbf{x}^T \hat{\mathbf{w}}$ .

$$\begin{aligned} &= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} \hat{w}_1 \\ \hat{w}_2 \end{bmatrix} \\ &= \hat{w}_1 x_1 + \hat{w}_2 x_2 = \hat{w}_1 x_1 + \hat{w}_2 \alpha x_1 \\ &= (\hat{w}_1 + \hat{w}_2 \alpha) x_1 = (\hat{w}_1 + \hat{w}_2 \alpha + \beta - \beta) x_1 \text{ for any } \beta. \\ &= ((\hat{w}_1 + \beta) + (\hat{w}_2 \alpha - \beta)) x_1 \text{ for any } \beta. \\ &= (\hat{w}_1 + \beta) x_1 + (\hat{w}_2 \alpha - \beta) \frac{x_2}{\alpha} = (\hat{w}_1 + \beta) x_1 + (\hat{w}_2 \alpha - \beta) \frac{1}{\alpha} x_2 \\ &= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} \hat{w}_1 + \beta \\ \frac{\hat{w}_2 \alpha - \beta}{\alpha} \end{bmatrix} = \mathbf{x}^T \tilde{\mathbf{w}} = \hat{\mathbf{y}} \text{ for } \tilde{\mathbf{w}} = \begin{bmatrix} \hat{w}_1 + \beta \\ (\hat{w}_2 \alpha - \beta) / \alpha \end{bmatrix} \end{aligned}$$

Of all the  $\{\mathbf{w}_{MLE}\}$  with zero error, is there one that intuitively might be generally be better?

# Intuition for choosing one specific $\hat{\mathbf{w}}$ .

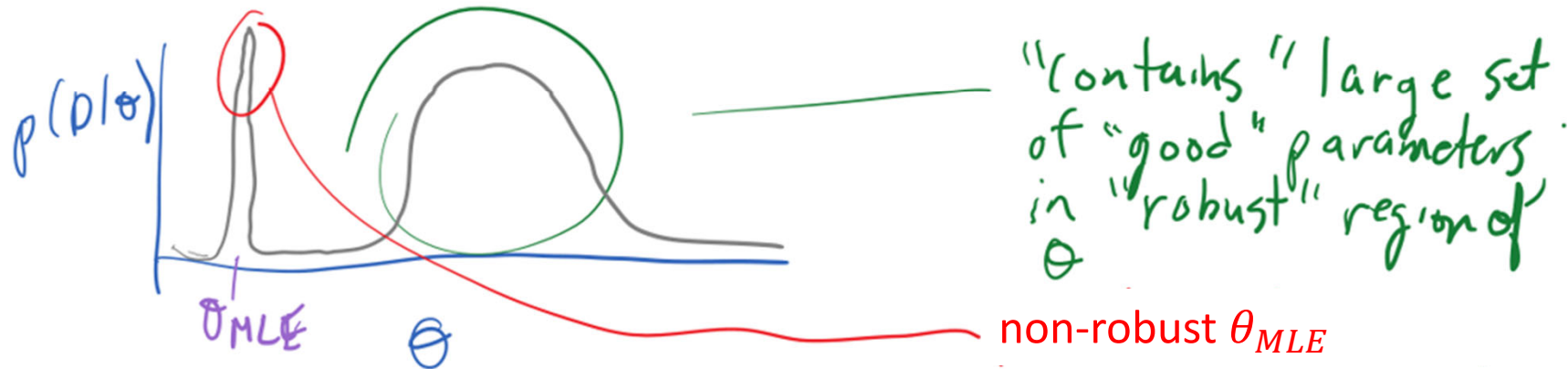
Of the  $\infty$  solutions for  $\mathbf{w}_{MLE}$ , choose the one with the least norm,  $\|\mathbf{w}_{MLE}\|_2$ .  
Why might this be a good idea?  $\hat{\mathbf{y}} = \mathbf{x}^T \hat{\mathbf{w}}$

- We would like each feature to have as little effect on the outcome as possible while still predicting well, so that model behaves “gracefully”.
- Consider prediction,  $\hat{\mathbf{y}} = \mathbf{w}^T \mathbf{x}$ . How much does the prediction change when we perturb,  $\mathbf{x}' = \mathbf{x} + \delta$ , for different norm  $\mathbf{w}$ ?
- With smaller coefficients the model is less sensitive to noisy data.
- What about for non-degenerate linear regression ( $\mathbf{A}^T \mathbf{A}$  is invertible)?
- Yes! For many problems (and models), small param norm is a good idea.
- This is one e.g. of *regularization*: in effect, reduce # free parameters, while keeping the same set of parameters!

# Recall how MLE can go wrong?

MLE yields a "point estimate" of our parameter

- When we perform MLE, we get just one single estimate of the parameter,  $\theta$ , rather than a distribution over it which captures uncertainty.
- In Bayesian statistics, we obtain a (posterior) distribution over  $\theta$ . We will touch more on this in a few lectures.



# L2 regularized linear regression

To *shrink*  $\mathbf{w}$  to have smaller norm than the MLE solution, we add a “penalty” term to the loss function:

$$\mathcal{L} = (\mathbf{y} - A\mathbf{w})^T (\mathbf{y} - A\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2$$
$$\mathbf{w}_{L_2} = \operatorname{argmin}_{\mathbf{w}} (\mathbf{y} - A\mathbf{w})^T (\mathbf{y} - A\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2$$

Also called “Ridge” regression, or L2 linear regression.  
Related to *Bayesian* modeling (next).

# The Bayesian modelling approach



- Bayesians put a *prior distribution* on the parameters,  $p(\theta)$ .
- Then they seek to compute the *posterior distribution*,  $p(\theta|D)$ .
- Then, predictive distribution is given by

$$p(y|x) = \int_{\theta} p(y|x, \theta) p(\theta|D) d\theta = E_{\theta}[p(y|x, \theta)]$$

- Procedurally, this is done using Bayes' rule:

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}.$$

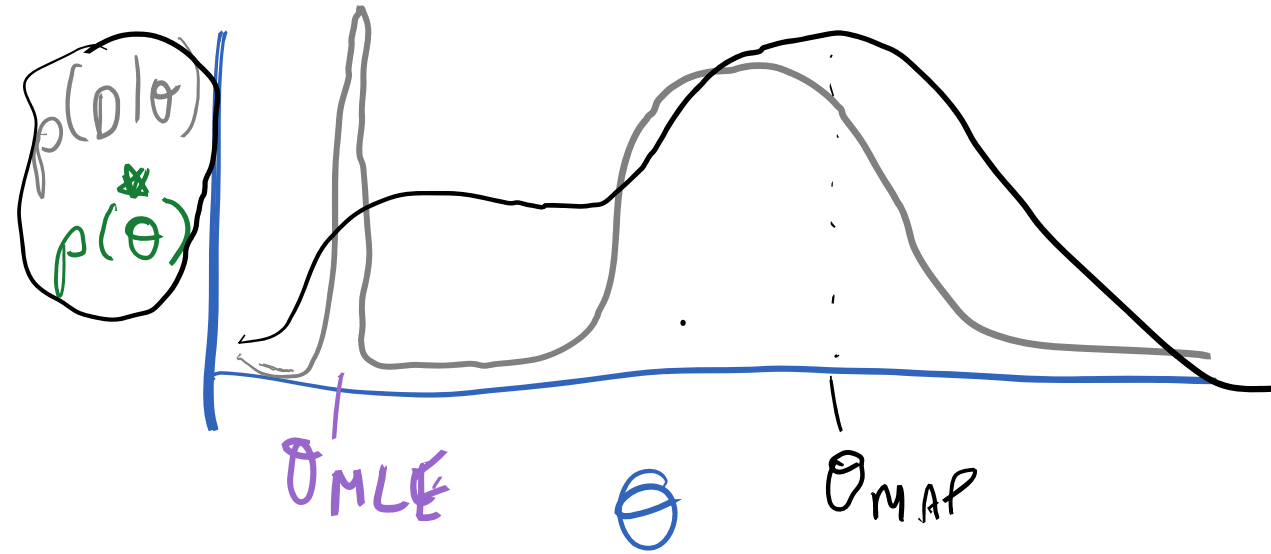
- Difficult in practice!  $p(D) = \int_{\theta} p(D, \theta) d\theta = \int_{\theta} p(D|\theta)p(\theta) d\theta$
- We will be lazy, instead being pseudo Bayesians, yielding L2 regression:
- $\theta_{lazy} = \operatorname{argmax}_{\theta} p(\theta|D)$  Maximum A Posteriori (MAP) estimation.



# MAP: the lazy Bayesian (*Maximum A Posteriori*)

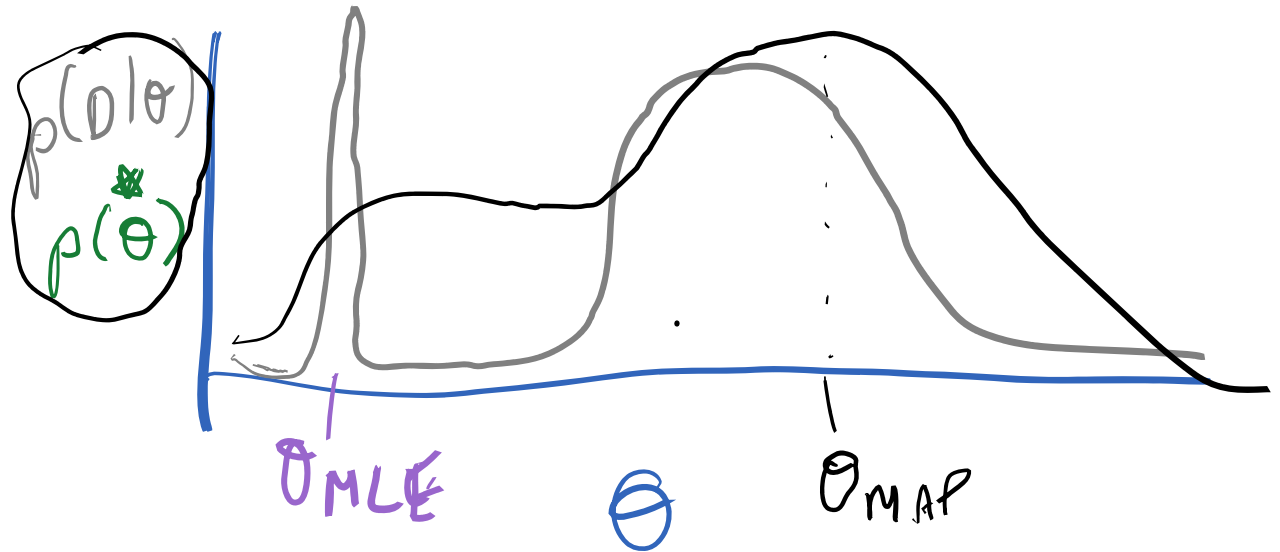
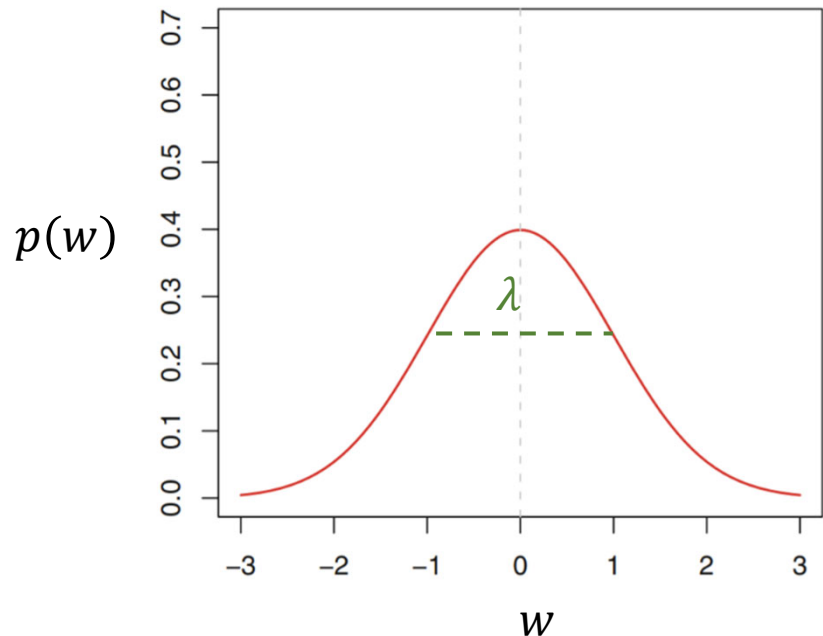
- Still use a prior over parameters,  $p(\theta)$ .
- Finds point estimate of the parameter that maximizes the posterior.
- $\theta_{MAP} = \operatorname{argmax}_{\theta} p(\theta|D)$

$$\begin{aligned} &= \operatorname{argmax}_{\theta} \frac{p(D|\theta)p(\theta)}{p(D)} \\ &= \operatorname{argmax}_{\theta} p(D|\theta)p(\theta) \end{aligned}$$



# A prior for small weights yields L2 regression!

- Zero-mean prior,  $p(w) = N(w; 0, \lambda I)$ .
- Bayesian posterior,  $p(w|D) = \frac{p(D|w)p(w)}{p(D)}$  is then “nice” in that everything is Gaussian (can work it out using MVGs).



# MAP for linear regression w Gaussian prior

$$\begin{aligned}w_{MAP} &= \operatorname{argmax}_w \log p(D|w) p(w) = \operatorname{argmax}_w \log p(D|w) + \log N(w; 0, \lambda I) \\&= \operatorname{argmax}_w \sum_{i=1}^N \log N(y_i | w^T x_i, \sigma^2) + \log N(w | 0, \lambda I) \\&= \operatorname{argmin}_w \frac{1}{2\sigma^2} (y - Aw)^T (y - Aw) - \sum_{i=1}^d \log \left[ \frac{1}{\sqrt{2\pi\lambda}} \exp \left( -\frac{(w_i - 0)^2}{2\lambda} \right) \right] \\&= \operatorname{argmin}_w \frac{1}{2\sigma^2} (y - Aw)^T (y - Aw) + \sum_{i=1}^d \left[ -\log \frac{1}{\sqrt{2\pi\lambda}} + \frac{w_i^2}{2\lambda} \right] \\&= \operatorname{argmin}_w \frac{1}{2\sigma^2} (y - Aw)^T (y - Aw) + \sum_d \frac{1}{2\lambda} w_i^2 \\&= \operatorname{argmin}_w \frac{1}{2\sigma^2} (y - Aw)^T (y - Aw) + \frac{1}{2\lambda} \|w\|_2^2 \\&= \operatorname{argmin}_w (y - Aw)^T (y - Aw) + 2\sigma^2 \frac{1}{2\lambda} \|w\|_2^2 \\&= \operatorname{argmin}_w (y - Aw)^T (y - Aw) + \lambda' \|w\|_2^2 \quad \text{for } \lambda' = \frac{\sigma^2}{\lambda}.\end{aligned}$$

Equivalence between  
MAP w Gaussian prior  
and L2 regression!

# Obtaining the MAP/L2 solution

$$\begin{aligned}w_{L_2} &= \operatorname{argmin}_w (y - Aw)^T (y - Aw) + \lambda \|w\|_2^2 \\ &= \operatorname{argmin}_w (y - Aw)^T (y - Aw) + \lambda w^T w\end{aligned}$$

Take partial derivative and set to zero:

$$\nabla_w \mathcal{L}_{MAP} = -2A^T y + 2A^T A w + 2\lambda I w$$

$$\rightarrow 0 = -A^T y + A^T A w + \lambda I w$$

$$\rightarrow A^T y = (A^T A + \lambda I) w$$

$$\rightarrow (A^T A + \lambda I)^{-1} A^T y = w$$

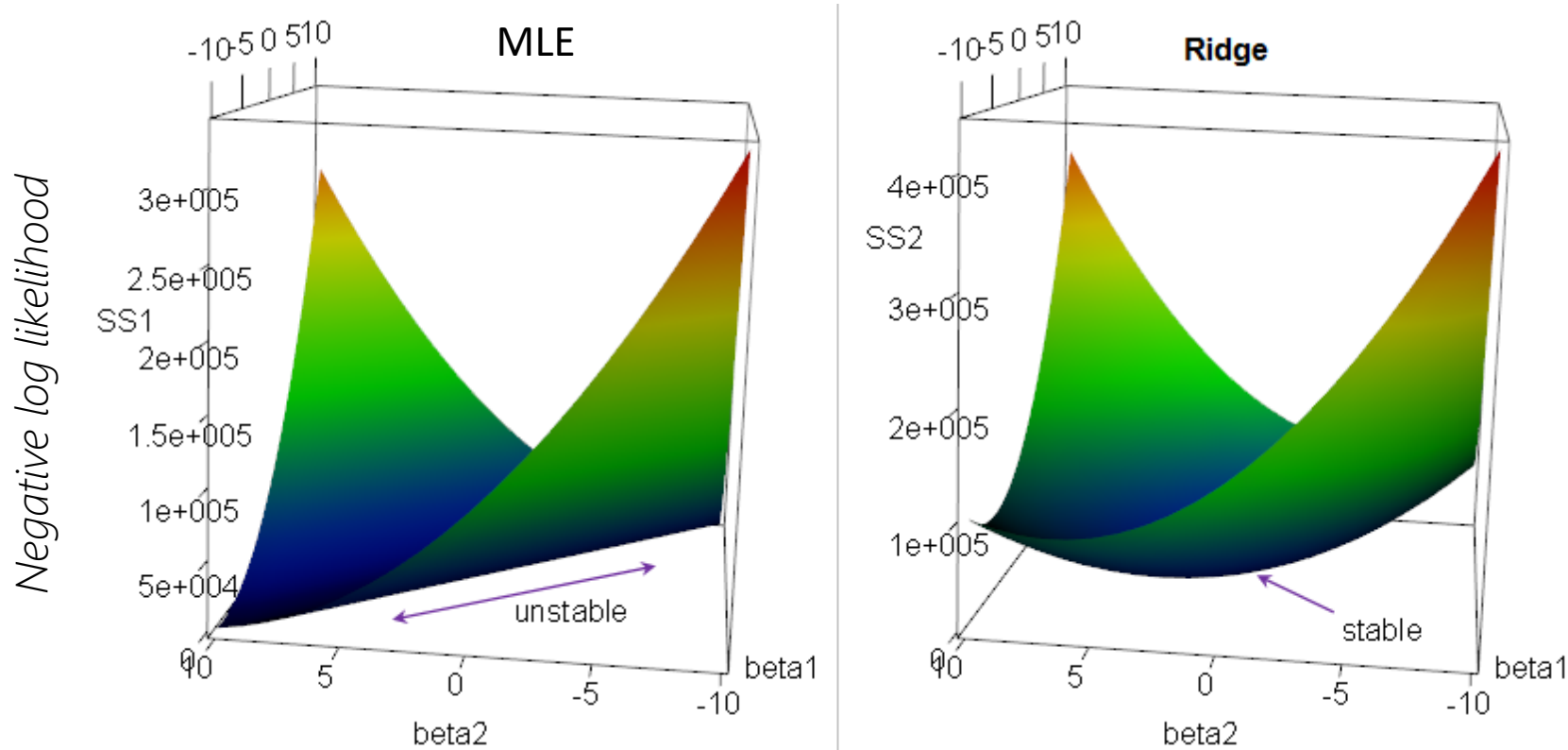
$$\text{So } w_{L_2} = (A^T A + \lambda I)^{-1} A^T y.$$

If  $\lambda > 0$ , we can invert  $(A^T A + \lambda I)$ .

$$\begin{aligned}A &= \Phi D \Phi^T \\ A^{-1} &= \Phi D^{-1} \Phi^T \\ \text{where } D^{-1} &= \begin{bmatrix} \lambda_1^{-1} & & \\ & \lambda_2^{-1} & \\ & & \ddots \\ & & & \lambda_n^{-1} \end{bmatrix}\end{aligned}$$

# Aside, why is this called “Ridge” regression?

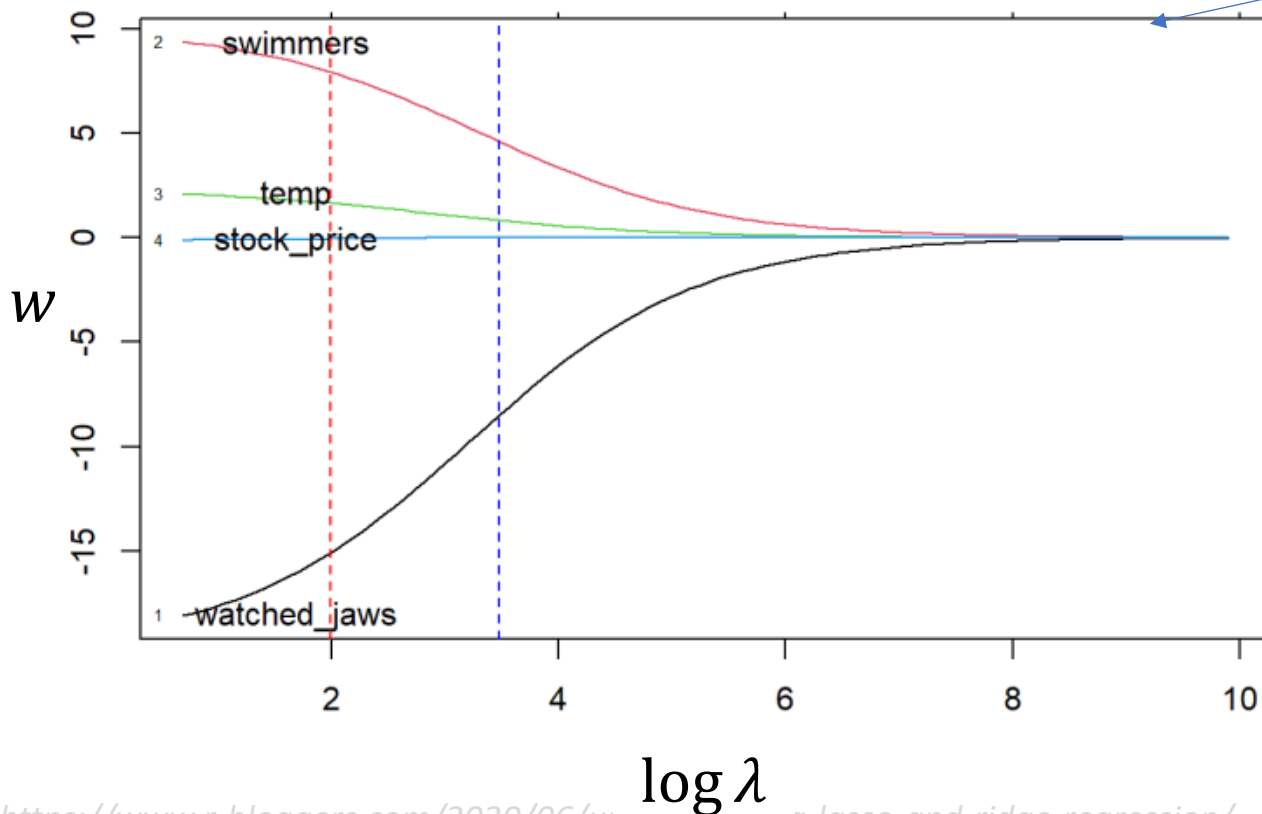
When some features are linearly dependent (can't invert  $\mathbf{A}^T \mathbf{A}$ ), we have  $\infty$  many equally good solutions that form a ridge.



# Effect of value of $\lambda$

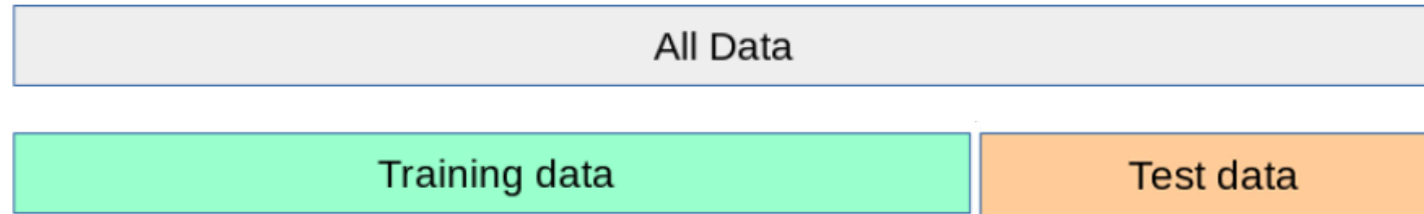
$$\mathcal{L}_{MAP} = (y - Aw)^T (y - Aw) + \lambda \|w\|_2^2$$

# of non-zero features

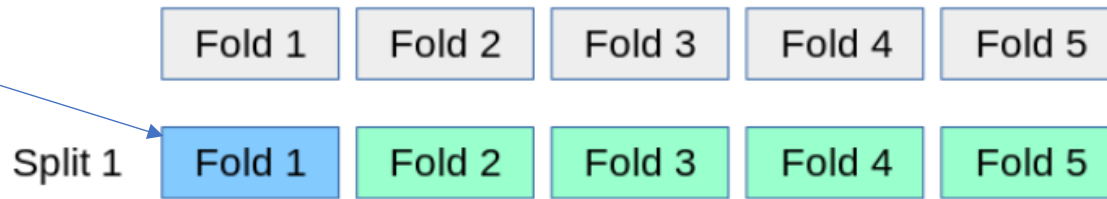


- Practically, how should we set  $\lambda$ ?
- Can we treat it as a parameter in the loss, and minimize wrt it?
- No: cannot use MLE!
- Need independent data, a *validation set* on which to evaluate the loss.

# Train/validation/test split



Validation set from  
the training data



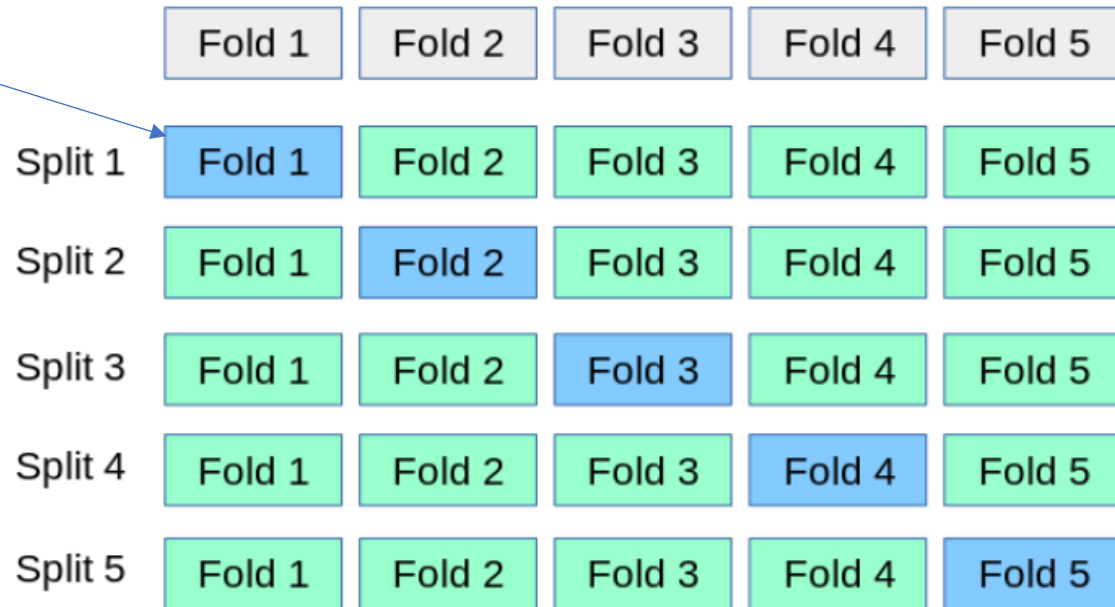
1. Find value of hyperparameter that is "best" on the **validation set**.
2. Quantify performance on the **test data**.

"Best"/quantify require computing the log likelihood on the held out data, and this is our metric.

# K-fold cross-validation



Validation set from  
the training data



1. Find value of hyperparameter that is "best" on the **validation set**.
2. Quantify performance on the **test data**.

"Best"/quantify require computing the log likelihood on the held out data, and this is our metric.

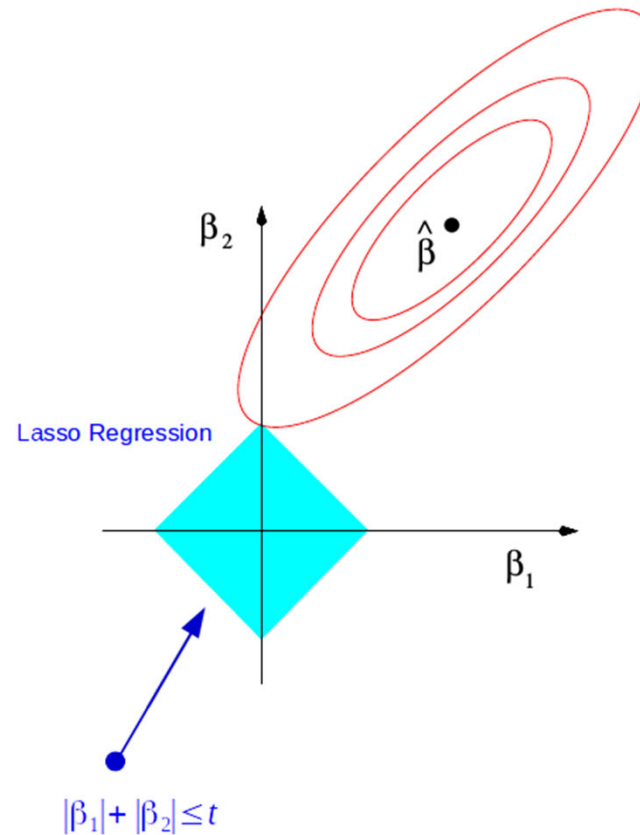


# Other priors for MAP in linear regression?

- What if wanted to have the linear function only depend on a few features (*i.e.*, the coefficients of  $\mathbf{w}$  are sparse)?
- Add a penalty that counts the # of non-zero weights—an  $L_0$  penalty,  $\lambda \|\mathbf{w}\|_0$ .
- Not differentiable! (becomes combinatorial optimization—nasty).
- But...the  $L_1$  norm penalty,  $\lambda \|\mathbf{w}\|_1 = \lambda \sum_d |\mathbf{w}_d|$ , tends to induce sparse  $\mathbf{w}$ —differentiable everywhere except at  $\mathbf{w}_i = \mathbf{0}$  which can be worked around.
- This is called *Lasso* regression, or  $L_1$ -penalized regression.

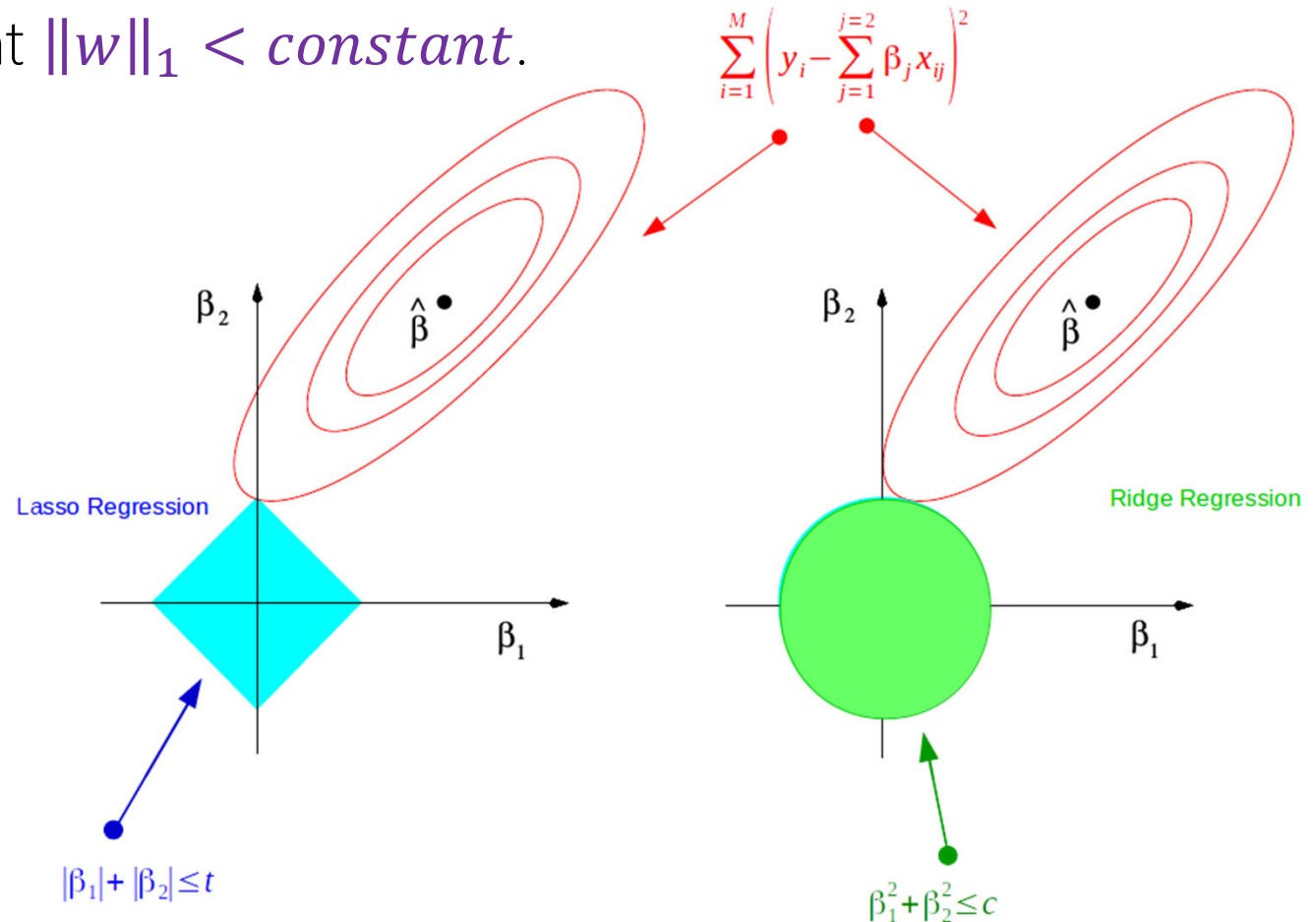
# $L_1$ -penalized linear regression, aka *Lasso*

- $w_{L_1} = \operatorname{argmin}_w (y - Aw)^T (y - Aw) + \lambda \|w\|_1$
- Why does the  $L_1$  norm penalty tends to induce sparse  $w$ ?
- Equivalent to MLE with constraint  $\|w\|_1 < \text{constant}$ .
- “Pointy” constraint surface is jutting out along the axes.
- In many cases, the  $L_1$  norm constraint will cause the unconstrained solution to intersect the constraint at a corner.
- The corners are where some coefficients are 0, which is a sparse solution..

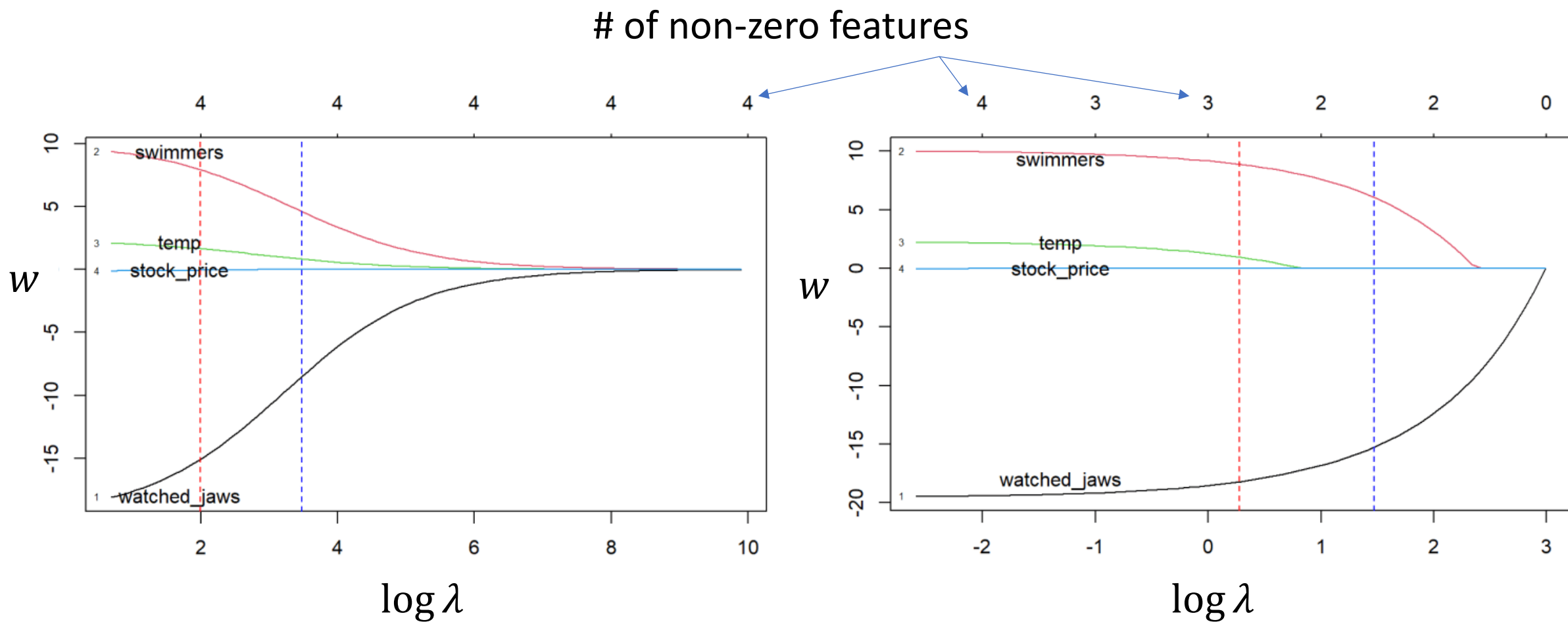


# $L_1$ -penalized linear regression, aka *Lasso*

- $w_{L_1} = \underset{w}{\operatorname{argmin}} (y - Aw)^T (y - Aw) + \lambda \|w\|_1$
- Why does the  $L_1$  norm penalty tends to induce sparse  $w$ ?
- Equivalent to MLE with constraint  $\|w\|_1 < \text{constant}$ .
- “Pointy” constraint surface is jutting out along the axes.
- In many cases, the  $L_1$  norm constraint will cause the unconstrained solution to intersect the constraint at a corner.
- The corners are where some coefficients are 0, which is a sparse solution..



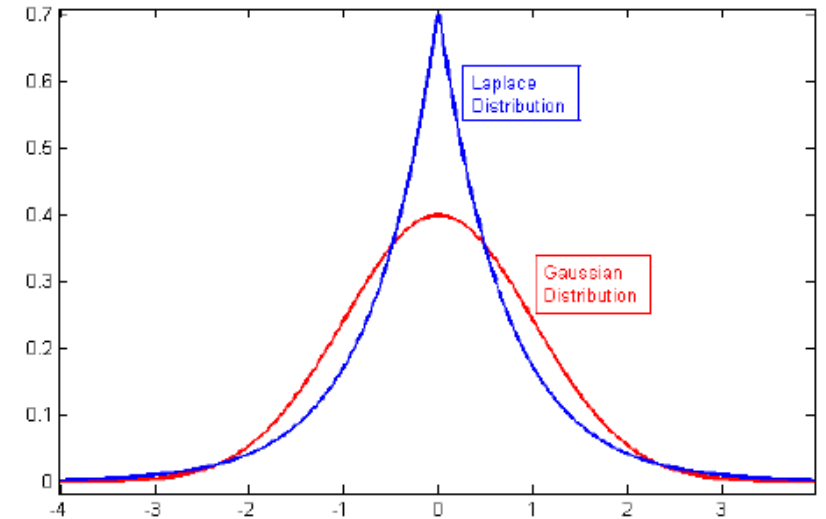
# Ridge vs Lasso: shrinkage vs sparsity



# MAP interpretation for Lasso/ $L_1$ -penalized linear regression?

- $L_2$  regression arose from a  $N(0, \lambda I)$  prior.
- Is there a prior corresponding to  $L_1$ ?
- Yes, the Laplace prior!

$$p(w) = \frac{1}{2\lambda'} \exp\left(-\frac{\|w\|_1}{\lambda'}\right).$$

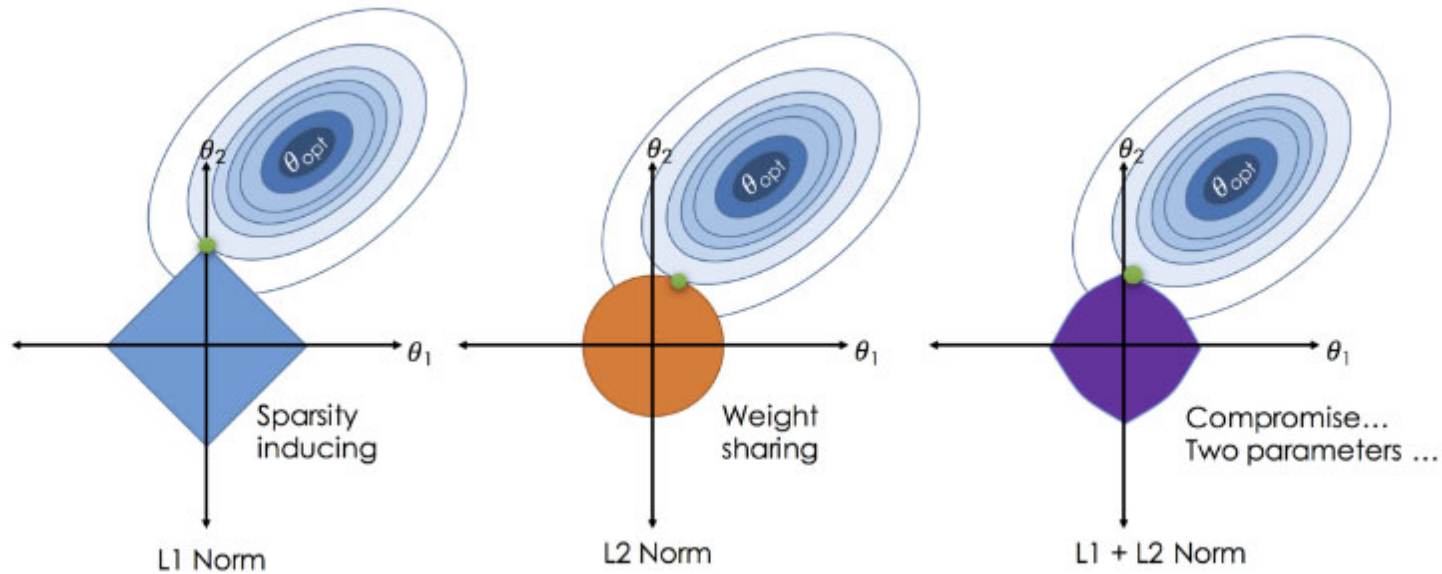


Issues with LASSO:

- If highly correlated features, tends to ignore all but one.

# Combine $L_1$ and $L_2$ penalties?

Yes, "elastic net regression".



Extra slides (not responsible for)

# Real application: Genome-Wide Association Studies (GWAS)

Input:

- A set of people with/without a disease
- Measure a large set of genetic markers for each person (e.g., SNPs: *single nucleotide polymorphism*).

Desired output:

- A list of genetic markers underlying the disease.



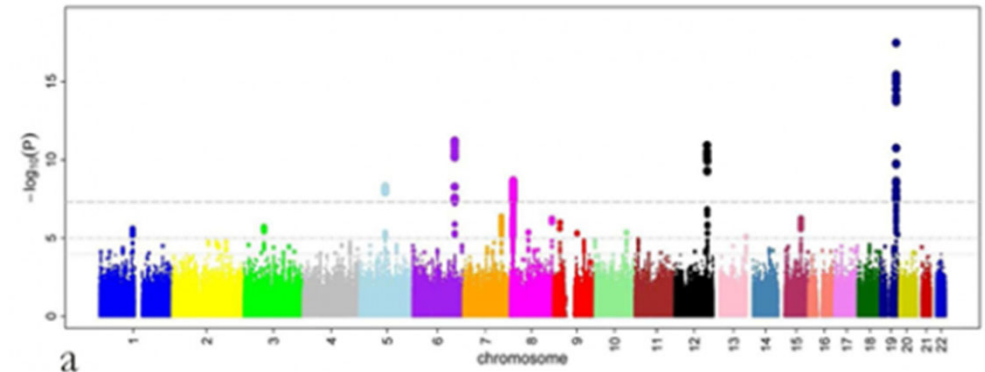
DNA from set of individuals

+

```
ATGTTGAATCTG
AAAGTGAAATGT
TATTATACGAAG
AAGTATTTGCTA
GACCTCAAAACC
CTTCATCATAAC
```



list of genetic markers







# Mixed Model Approach (GPR)

Generative model:

- sample a “latent ancestry vector”
- add as covariate

$$p(\vec{u}) = N(\vec{0}, \text{covariance matrix})$$

$$p(\vec{y}|\vec{u}) = N(X\beta + \eta\vec{u}, I\sigma^2)$$

“amount of  
confounding  
genetic signal”

➡ LIKELIHOOD

$$p(\vec{y}) = \int p(\vec{y}|\vec{u})p(\vec{u})d\vec{u}$$
$$p(\vec{y}) = N(X\beta, \eta^2 \text{covariance matrix} + I\sigma^2)$$

## 2. Exploit rank when few SNPs are used

If  $\mathbf{K}$  is computed in a particular way, and  $n > s_c$  then computations are linear in  $n$ .

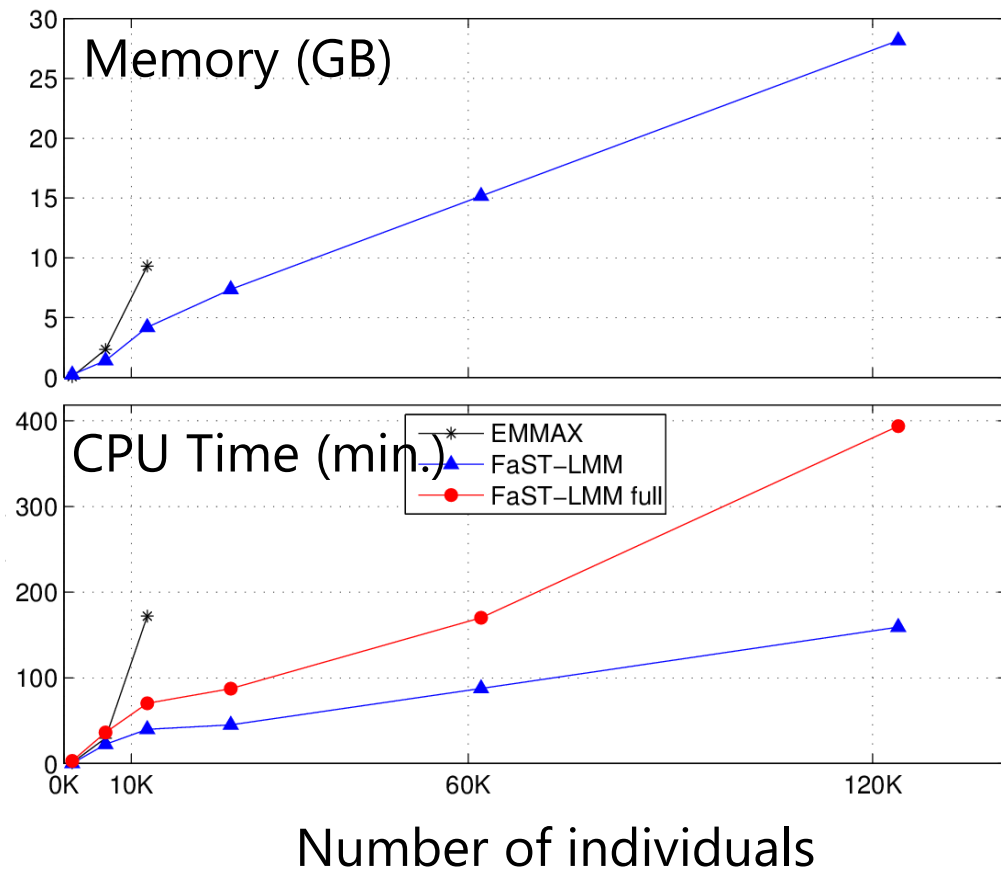
$$\begin{matrix} & \overbrace{\hspace{1cm}}^n \\ \underbrace{\hspace{1cm}}_n & \mathbf{K} \end{matrix} = \frac{1}{s_c} \begin{matrix} & \overbrace{\hspace{1cm}}^{s_c} \\ \underbrace{\hspace{1cm}}_n & \mathbf{W} \end{matrix} \times \mathbf{W}^T$$

*e.g.*, the realized relationship matrix (RRM) [Visscher et al]

$n$  is # people

$s_c$  is # of SNPs used to compute  $\mathbf{K}$

# Experimental running time and memory



## Related papers:

### Improved linear mixed models for genome-wide association studies

Jennifer Listgarten, C. Lippert, C. Kadie, R. Davidson, E. Eskin and D. Heckerman  
(equal contributions and corresponding)

*Nature Methods* 2012, [doi:10.1038/nmeth.2037](https://doi.org/10.1038/nmeth.2037)

### FaST Linear Mixed Models for Genome-Wide Association Studies

C. Lippert, Jennifer Listgarten, Y. Liu, C. Kadie, R. Davidson and D. Heckerman  
(equal contributions and corresponding)

*Nature Methods* Aug. 2011, [doi:10.1038/nmeth.1681](https://doi.org/10.1038/nmeth.1681)

### Epigenome-wide association studies without the need for cell-type composition

James Zou, C. Lippert, D. Heckerman, Martin Aryee, Jennifer Listgarten, *Nature Methods* 2014 ([journal link](#))

### FaST-LMM-Select for addressing confounding from spatial structure and rare variants

Jennifer Listgarten, Christoph Lippert, David

Heckerman (equal contributions and corresponding)  
*Nature Genetics* 2013 ([journal link](#))

- **EMMAX** out of memory after 13K individuals
- **FaST-LMM** fixed variance components
- **FaST-LMM full** re-estimate variance components