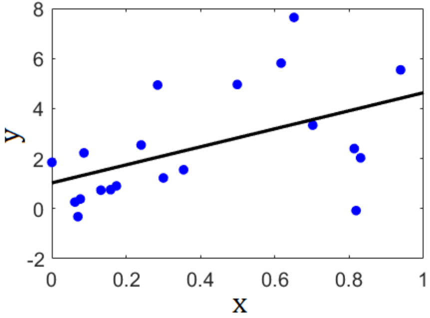
# 4. Linear regression

- Linear regression
- Bayesian linear regression
- Solving linear systems

# Linear regression: Application

# Fitting a line to data

Fit a line to observations $y_i$ given input $x_i$, $i = 1, ..., n$




$$\begin{array}{ll} \underset{\alpha,\beta}{\text{minimize}} & \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ \text{subject to} & \alpha x_i + \beta = y_i \end{array}$$

# Fitting a line to data

$$\underset{\alpha,\beta}{\text{minimize}} \quad \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 \qquad \text{subject to} \quad \alpha x_i + \beta = y_i$$

Write in matrix form

$$\underset{\theta}{\text{minimize}} \quad \|X\theta - y\|_2^2 = \sum_{i=1}^{m} (x_i^T \theta - y_i)^2$$

where

$$X = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_m & 1 \end{bmatrix}, \quad y = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_m \end{bmatrix}$$

Solution

$$\theta = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

(Go to demo)

# Buying a house

How much does this house cost?

- Quality of construction?
- Is the neighborhood nice?
- Who won the world series last year?

Which factors matter? Market analysis required...

# Linear regression over reals

- $x[1] =$ Quality of construction?
- $x[2] =$ Is the neighborhood nice?
- $x[3] =$ Who won the world series last year?

Market analysis:

1. For each house $i$, collect $x_i = [x_i[1], x_i[2], ..., x_i[n]]^T$, and construct

$$X = \begin{bmatrix} x_1 & x_2 & \cdots x_m \end{bmatrix}^T \in \mathbb{R}^{m \times n}$$

2. Collect past observations

$$y = \begin{bmatrix} y_1 & y_2 & \cdots y_m \end{bmatrix}^T, \quad y_i \in \mathbb{R} = \text{cost of house } i$$

3. Find $\theta = \underset{\theta}{\operatorname{argmin}} \|X\theta - y\|_2^2$

4. My predictions: given $x_{\text{new}}$, predict cost $y_{\text{new}} = \theta^T x_{\text{new}}$.

# Selling a house

Will they buy the house?

- Quality of construction?
- Is the neighborhood nice?
- Who won the world series last year?

Which factors matter? Market analysis required...

## Linear regression for classification?

- $x[1] =$ Quality of construction?
- $x[2] =$ Is the neighborhood nice?
- $x[3] =$ Who won the world series last year?

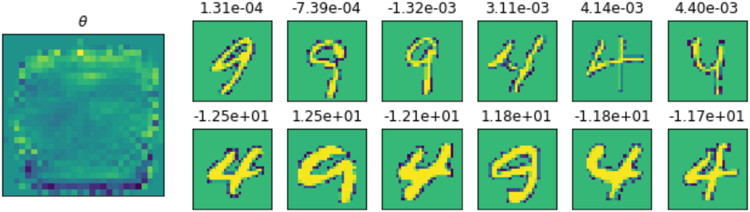
Market analysis:

1. For each house $i$, collect $x_i = [x_i[1], x_i[2], ..., x_i[n]]^T$, and construct
$$X = \begin{bmatrix} x_1 & x_2 & \cdots x_m \end{bmatrix}^T \in \mathbb{R}^{m \times n}$$
2. Collect past observations
$$y = \begin{bmatrix} y_1 & y_2 & \cdots y_m \end{bmatrix}^T, \quad y_i = \begin{cases} 1 & \text{bought it} \\ -1 & \text{didn't buy it} \end{cases}$$
3. Find $\theta = \underset{\theta}{\operatorname{argmin}} \|X\theta - y\|_2^2$
4. Realtor's predictions: given $x_{\text{new}}$, predict $y_{\text{new}} = \mathbf{sign}(\theta^T x_{\text{new}})$.

# Use in classification

Go to demo

$x^T\theta$




Linear regression: training error: 10.98%, test error: 12.36%

# Generalized linear regression

- Most equations of motions are polynomials

## Generalized linear regression

- Most equations of motions are polynomials
- A reasonable model:

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + ... \text{ nonlinear?}$$

## Generalized linear regression

- Most equations of motions are polynomials
- A reasonable model:

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \ldots \text{ nonlinear?}$$

- Nope, it's generalized linear!

$$\bar{x} = \begin{bmatrix} 1 & x & x^2 \cdots x^{n-1} & & \\ 1 & x_2 & x_2^2 \cdots x_2^{n-1} & & \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 \cdots x_m^{n-1} & & \end{bmatrix}$$

# Generalized linear regression

- Most equations of motions are polynomials
- A reasonable model:

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \ldots \text{ nonlinear?}$$

- Nope, it's generalized linear!

$$x = \begin{bmatrix} 1 & x & x^2 \cdots x^{n-1} & & \\ 1 & x_2 & x_2^2 \cdots x_2^{n-1} & & \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 \cdots x_m^{n-1} & & \end{bmatrix}$$

- Fit $\theta$ as solution to

$$\underset{\theta}{\text{minimize}} \quad \|\bar{X}\theta - y\|_2^2$$

## Generalized linear regression

- Most equations of motions are polynomials
- A reasonable model:

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \text{... nonlinear?}$$

- Nope, it's generalized linear!

$$\bar{x} = \begin{bmatrix} 1 & x & x^2 \cdots x^{n-1} & & \\ 1 & x_2 & x_2^2 \cdots x_2^{n-1} & & \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 \cdots x_m^{n-1} & & \end{bmatrix}$$

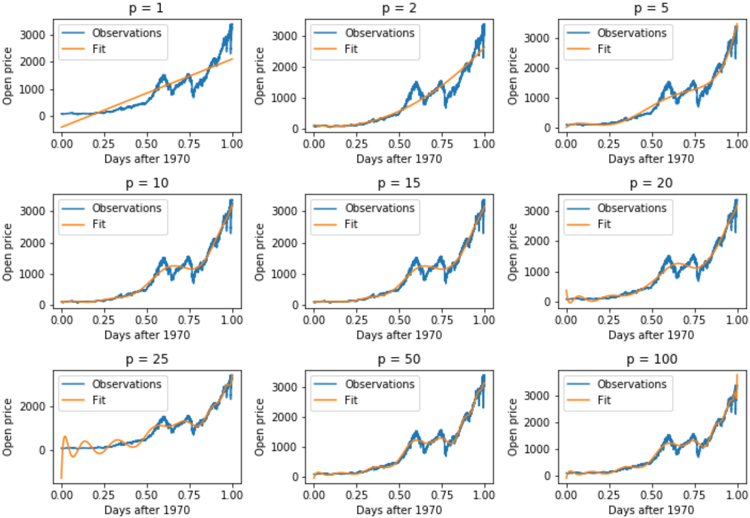
- Fit $\theta$ as solution to

$$\underset{\theta}{\text{minimize}} \quad \|\bar{X}\theta - y\|_2^2$$

- From now on, assume $X_{ij}$ can contain nonlinearities, and ignore it

# Polynomial fit demo

Go to demo: S+P 500 historical open prices



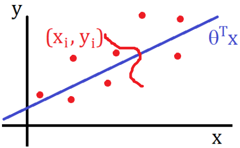
# Bayesian linear regression

# MLE under linear model, Gaussian noise

Suppose that

$$y_i \sim \mathcal{N}(\underbrace{\theta^T x_i}, 1)$$

(Gaussian with mean $\cancel{\theta}$, variance 1).

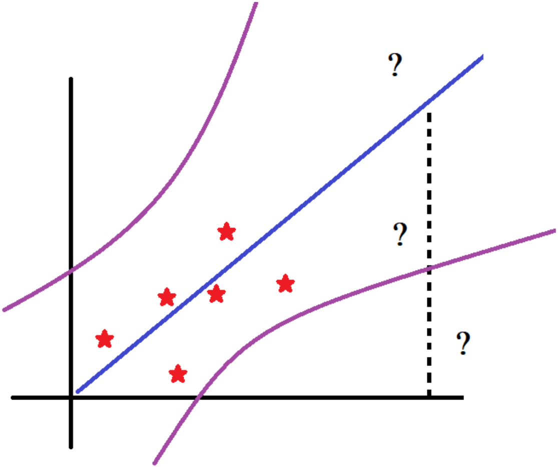



Then

$$
\begin{aligned}
\theta_{\mathbf{MLE}} &= \operatorname*{argmax}_{\theta} \Pr(\mathcal{Y}|\mathcal{X}, \theta) \\
&= \operatorname*{argmax}_{\theta} \frac{1}{\sqrt{2\pi}} \prod_i \exp\left(-\frac{(y_i - \theta^T x_i)^2}{2}\right) \\
&\overset{\log}{=} \operatorname*{argmin}_{\theta} \sum_i (y_i - \theta^T x_i)^2
\end{aligned}
$$

$\theta_{\mathbf{MLE}}$ is the solution to <u>linear regression</u>!

# What if $\theta$ is uncertain?




Not enough “good” observations

# Maximum a posteriori (MAP) estimator

The **maximum a posteriori estimator** is the best guess of $\theta$ given $\bar{D}$

$$\theta_{\mathbf{MAP}} = \operatorname*{argmax}_{\theta} \Pr(\theta|\bar{D}, \mu)$$

To compute, use **Bayes' rule**

D = "my throat hurts"
theta = "sick" vs "faking it"

$$\underbrace{\Pr(\theta|\bar{D}, \mu)}_{\text{posterior}} = \frac{\overbrace{\Pr(\bar{D}|\theta, \mu)}^{\text{likelihood}} \; \overbrace{\Pr(\theta|\mu)}^{\text{prior}}}{\underbrace{\Pr(\bar{D}|\mu)}_{\text{doesn't depend on } \theta}}$$

$\mu$ is a modeling hyperparameter

# MAP under linear model, Gaussian noise + uncertainty

$$\underbrace{\Pr(\theta|\bar{D},\mu)}_{\text{posterior}} = \frac{\overbrace{\Pr(\bar{D}|\theta,\mu)}^{\text{likelihood}} \ \overbrace{\Pr(\theta|\mu)}^{\text{prior}}}{\underbrace{\Pr(\bar{D}|\mu)}_{\text{doesn't depend on } \theta}}$$

Model: $y \sim \mathcal{N}(x^T\theta, 1)$, $\theta \sim \mathcal{N}(\mu, I)$

N = normal distribution

$$\begin{aligned}
\theta_{\mathbf{MAP}} &= \operatorname*{argmax}_{\theta} \frac{\overbrace{\prod_i \exp(-(\theta^T x_i - y_i)^2)}^{\text{likelihood}} \cdot \overbrace{\exp(-(\theta-\mu)^2)}^{\text{prior}}}{\text{constant}} \\
&\overset{\log}{=} \operatorname*{argmax}_{\theta} - \sum_{i=1}^{m} (x_i^T\theta - y_i)^2 + -\sum_{k=1}^{d} (\theta[k] - \mu[k])^2
\end{aligned}$$

$\Rightarrow$ (2-norm) regularized least squares!

# MAP under linear model, Gaussian noise + uncertainty

Model: $y \sim \mathcal{N}(x^T\theta, 1)$, $\theta \sim \mathcal{N}(\mu, I)$

$$\sum_{i=1}^{m}(x_i^T\theta - y_i)^2 + \sum_{k=1}^{d}(\theta[k] - \mu[k])^2 = \frac{1}{2}\|X\theta - y\|_2^2 + \frac{\alpha}{2}\|\theta - \mu\|_2^2$$

$$\overset{\text{Assume } \mu = 0}{=} \frac{1}{2}\|X\theta - y\|_2^2 + \frac{\alpha}{2}\|\theta\|_2^2$$

ridge regression

*It's not that $\mu = 0$ somehow makes life easier, but that without knowing anything else, it's as good a choice as any

Solving linear systems

# Matrix form

$$f(\theta) = \frac{1}{2}\sum_{i=1}^{m}(x_i^T\theta - y_i)^2 + \frac{\alpha}{2}\sum_{k=1}^{d}\theta[k]^2 = \frac{1}{2}\|X\theta - y\|_2^2 + \frac{\alpha}{2}\|\theta\|_2^2$$

- Pack data $x_i$ as ~~columns~~ rows of $X \in \mathbb{R}^{m\times d}$, labels $y_i$ as values of $y \in \mathbb{R}^m$
- Euclidean norm (2-norm) of a vector

$$\|\theta\|_2 := \sqrt{\sum_{i=1}^{d}\theta[k]^2}$$

- Generalized regularization parameter $\alpha > 0$, we take $\mu = 0$ w.l.o.g.

# Normal equations

$$f(\theta) = \frac{1}{2}\|X\theta - y\|_2^2 + \frac{\alpha}{2}\|\theta\|_2^2$$

- Convex, minimized when gradient $= 0$

$$\nabla f(\theta) = X^T(X\theta - y) + \alpha\theta = 0 \qquad (\star)$$

where the gradient of $f : \mathbb{R}^d \to \mathbb{R}$ is defined as

$$\nabla f(\theta) := \begin{bmatrix} \frac{\partial f(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial f(\theta)}{\partial \theta_d} \end{bmatrix}$$

- The linear system $(\star)$ is called the <u>normal equations</u>

# Solving normal equations

$$\underbrace{(X^T X + \alpha I)}_{=:A}\,\theta = \underbrace{X^T y}_{=:b}$$

- Is $A$ invertible?
- If $A$ is invertible, then $\theta = A^{-1}b$. But is that a good idea in practice?

## Solving normal equations

$$\underbrace{(X^T X + \alpha I)}_{=:A}\theta = \underbrace{X^T y}_{=:b}$$

- Is $A$ invertible?

  Ans: yes, if $\alpha > 0$ or $X$ is <u>full rank</u>

- If $A$ is invertible, then $\theta = A^{-1}b$. But is that a good idea in practice?

## Solving normal equations

$$\underbrace{(X^T X + \alpha I)}_{=:A}\,\theta = \underbrace{X^T y}_{=:b}$$

- Is $A$ invertible?

  Ans: yes, if $\alpha > 0$ or $X$ is full rank

- If $A$ is invertible, then $\theta = A^{-1}b$. But is that a good idea in practice?

  Ans: No, badly conditioned matrix can cause terrible numerical issues

## Eigenvalue decomposition of a PSD matrix

Def: A is positive semidefinite (PSD) if $x^T A x \geq 0$ for all x

- Eigenvalue decomposition of symmetric ~~PSD~~ matrix $A$

$$A = \sum_{i=1}^{d} \lambda_i u_i u_i^T = U \Lambda U^T$$

where the eigenvectors are orthonormal

$$U = [u_1, ..., u_d], \qquad U^T U = U U^T = I$$

and w.l.o.g. the eigenvalues

$$\Lambda = \mathbf{diag}(\lambda_1, ..., \lambda_d) \,/\, \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$$

- If $A = X^T X + \alpha I$ then $\lambda_d \geq \alpha$. Why?

# Eigenvalue decomposition of a PSD matrix

- Eigenvalue decomposition of symmetric PSD matrix $A$

$$A = \sum_{i=1}^{d} \lambda_i u_i u_i^T = U \Lambda U^T$$

where the eigenvectors are orthonormal

$$U = [u_1, ..., u_d], \qquad U^T U = U U^T = I$$

and w.l.o.g. the eigenvalues

$$\Lambda = \mathbf{diag}(\lambda_1, ..., \lambda_d) \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$$

- If $A = X^T X + \alpha I$ then $\lambda_d \geq \alpha$. Why?

$$\lambda_d = u_d^T A u_d = \underbrace{\tfrac{1}{2} \|X u_d\|_2^2}_{\geq 0} + \tfrac{\alpha}{2} \underbrace{\|u_d\|_2^2}_{=1}$$

## Inverse of a PSD matrix

$$A = \sum_{i=1}^{d} \lambda_i u_i u_i^T = U \Lambda U^T, \qquad A^{-1} = \sum_{i=1}^{d} \frac{1}{\lambda_i} u_i u_i^T = U \Lambda^{-1} U^T$$

If $A = X^T X + \alpha I$ then $\lambda_d \geq \alpha$.

- But what if $\lambda_d = \alpha = 0$?
- If $\lambda_d \geq \alpha > 0$ is really small?

## Inverse of a PSD matrix

$$A = \sum_{i=1}^{d} \lambda_i u_i u_i^T = U \Lambda U^T, \qquad A^{-1} = \sum_{i=1}^{d} \frac{1}{\lambda_i} u_i u_i^T = U \Lambda^{-1} U^T$$

If $A = X^T X + \alpha I$ then $\lambda_d \geq \alpha$.

- But what if $\lambda_d = \alpha = 0$? Ans: Then inverse does not exist.
- If $\lambda_d \geq \alpha > 0$ is really small?

## Inverse of a PSD matrix

$$A = \sum_{i=1}^{d} \lambda_i u_i u_i^T = U \Lambda U^T, \qquad A^{-1} = \sum_{i=1}^{d} \frac{1}{\lambda_i} u_i u_i^T = U \Lambda^{-1} U^T$$

If $A = X^T X + \alpha I$ then $\lambda_d \geq \alpha$.

- But what if $\lambda_d = \alpha = 0$? Ans: Then inverse does not exist.
- If $\lambda_d \geq \alpha > 0$ is really small? Ans: Inverse exists, but $\frac{1}{\lambda_d}$ is really big
  We call this scenario <u>badly conditioned</u> (numerically unstable)

# Inverse of a PSD matrix

$$A = \sum_{i=1}^{d} \lambda_i u_i u_i^T = U \Lambda U^T, \qquad A^{-1} = \sum_{i=1}^{d} \frac{1}{\lambda_i} u_i u_i^T = U \Lambda^{-1} U^T$$

If $A = X^T X + \alpha I$ then $\lambda_d \geq \alpha$.

- But what if $\lambda_d = \alpha = 0$? Ans: Then inverse does not exist.
- If $\lambda_d \geq \alpha > 0$ is really small? Ans: Inverse exists, but $\frac{1}{\lambda_d}$ is really big

  We call this scenario badly conditioned (numerically unstable)

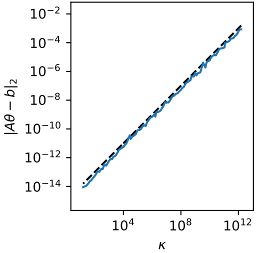
PSD

The condition number of $A$ is

$$1 \leq \kappa(A) := \frac{\max_i \lambda_i}{\min_i \lambda_i}$$

$> 0$

We desire $\kappa(A)$ as close to 1 as possible.



## What does this mean for you the implementer?

$$(X^T X + \alpha I)\theta = X^T y \iff A\theta = b$$

- In practice, we won't really expect $\alpha$ really big to give good results
- So, don't use matrix inverses!
- Alternatives:
  - Cholesky factorization + backsolve $O(d^3)$ complexity (in MATLAB, `A \ b`)
  - Conjugate gradient (only matrix-vector products, even better if $A$ is sparse)
  - Gradient / stochastic gradient method (cheaper if coarse precision is ok)

# Extension to harder problems

Why do we care so much about quadratic problems?

- Most smooth models can be modeled locally as a quadratic
- (undampened) Newton's method iteratively solves the linear system

  $$\nabla^2 f(\theta)^T(\theta_{\text{next}} - \theta) = -\nabla f(\theta)$$

  Newton + variants (Quasi-Newton, trust region,...) used in practice
- Landscape analysis: Condition number of $\nabla^2 f(\theta)$ describes "flatness" of $\theta$

## Summary

Linear regression

- Use in data fitting, in classification
- Can be used for nonlinear models, if aggregated linearly

Bayesian linear regression

- Fits under uncertainty (MLE vs MAP)

Solving linear systems

- Normal equations
- Positive semidefinite matrices, eigenvalue decompositions
- Problem conditioning
- Gradient descent