Association studies and regression
Fall 2016

Sriram Sankararaman
Acknowledgments: Fei Sha, Ameet Talwalkar
• HW1 posted on course website: http://web.cs.ucla.edu/~sriram/courses/cm226.fall-2016/html/index.html
• Due in two weeks in class.
Linear regression

- Predicting a continuous output
- Linear regression is the linear combination of features
  \[ f : x \rightarrow y, \text{ with } f(x) = \beta_0 + \sum_j \beta_j x_j = \beta_0 + \beta^T x \]
- If we minimize residual sum of squares as our learning objective, we get a closed-form solution of parameters
- Probabilistic interpretation: maximum likelihood if assuming residual is Gaussian distributed
- MLE can be computed analytically
Outline

Logistic regression
  General setup
  Maximum likelihood estimation
  Gradient descent
  Newton’s method

Application to GWAS
Logistic regression

What about binary phenotypes, e.g., disease vs healthy?

<table>
<thead>
<tr>
<th>Individual</th>
<th>SNPs</th>
<th>Phenotype</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A C G A A</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>C C G G T</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>C C T A T G</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>A T G A A</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>A T G G T A T</td>
<td></td>
</tr>
</tbody>
</table>

Can we use linear regression?
Logistic regression

Setup for binary outputs

- **Input:** \( x \in \mathbb{R}^m \)
- **Output:** \( y \in \{0, 1\} \)
- **Training data:** \( D = \{(x_i, y_i), i = 1, 2, \ldots, n\} \)
- **Model:**

\[
p(y = 1|x; \beta_0, \beta) = \sigma[g(x)]
\]

where

\[
g(x) = \beta_0 + \sum_j \beta_j x_j = \beta_0 + \beta^T x
\]

and \( \sigma[\cdot] \) stands for the _sigmoid_ function

\[
\sigma(a) = \frac{1}{1 + e^{-a}}
\]
Why the sigmoid function?

What does it look like?

\[ \sigma(a) = \frac{1}{1 + e^{-a}} \]

where \( a = \beta_0 + \beta^T x \)

Properties

- Bounded between 0 and 1 \( \leftarrow \) thus, interpretable as probability
- Monotonically increasing thus, usable to derive classification rules
  - \( \sigma(a) > 0.5 \), positive (classify as '1')
  - \( \sigma(a) < 0.5 \), negative (classify as '0')
  - \( \sigma(a) = 0.5 \), undecidable
- Nice computational properties

Linear or nonlinear classifier?

Association studies and regression

Logistic regression
Why the sigmoid function?

What does it look like?

\[ \sigma(a) = \frac{1}{1 + e^{-a}} \]

where \( a = \beta_0 + \beta^T x \)

Properties

- Bounded between 0 and 1 ← thus, interpretable as probability
- Monotonically increasing thus, usable to derive classification rules
  - \( \sigma(a) > 0.5 \), positive (classify as '1')
  - \( \sigma(a) < 0.5 \), negative (classify as '0')
  - \( \sigma(a) = 0.5 \), undecidable
- Nice computational properties

Linear or nonlinear classifier?
Why the sigmoid function?

What does it look like?

\[ \sigma(a) = \frac{1}{1 + e^{-a}} \]

where \( a = \beta_0 + \beta^T x \)

Properties

- Bounded between 0 and 1 \( \leftarrow \) thus, interpretable as probability
- Monotonically increasing thus, usable to derive classification rules
  - \( \sigma(a) > 0.5 \), positive (classify as '1')
  - \( \sigma(a) < 0.5 \), negative (classify as '0')
  - \( \sigma(a) = 0.5 \), undecidable
- Nice computational properties

Linear or nonlinear classifier?
\( \sigma(a) \) is nonlinear, however, the decision boundary is determined by

\[
\sigma(a) = 0.5 \Rightarrow a = 0 \Rightarrow g(x) = \beta_0 + \beta^T x = 0
\]

which is a \textit{linear} function in \( x \)
Likelihood function

Probability of a single training sample \((x_i, y_i)\)

\[
p(y_i | x_i, (\beta_0, \beta)) = \begin{cases} 
\sigma(\beta_0 + \beta^T x_i) & \text{if } y_i = 1 \\
1 - \sigma(\beta_0 + \beta^T x_i) & \text{otherwise}
\end{cases}
\]

Compact expression, exploring that \(y_i\) is either 1 or 0

\[
p(y_i | x_i, (\beta_0, \beta)) = \sigma(\beta_0 + \beta^T x_i)^{y_i} [1 - \sigma(\beta_0 + \beta^T x_i)]^{1-y_i}
\]
Likelihood function

Probability of a single training sample \((x_i, y_i)\)

\[
p(y_i|x_i, (\beta_0, \beta)) = \begin{cases} 
\sigma(\beta_0 + \beta^T x_i) & \text{if } y_i = 1 \\
1 - \sigma(\beta_0 + \beta^T x_i) & \text{otherwise}
\end{cases}
\]

Compact expression, exploring that \(y_i\) is either 1 or 0

\[
p(y_i|x_i, (\beta_0, \beta)) = \sigma(\beta_0 + \beta^T x_i)^{y_i}[1 - \sigma(\beta_0 + \beta^T x_i)]^{1-y_i}
\]
Log-likelihood of the whole training data $\mathcal{D}$

$$
\mathcal{L}(\beta_0, \beta) = \log P(\mathcal{D}|(\beta_0, \beta)) = \sum_{i} \{y_i \log \sigma(b + \beta^T x_i) + (1 - y_i) \log [1 - \sigma(b + \beta^T x_i)]\}
$$

It is convenient to work with its negation

$$
\mathcal{NLL}(\beta_0, \beta) = -\sum_{i} \{y_i \log \sigma(\beta_0 + \beta^T x_i) + (1 - y_i) \log [1 - \sigma(\beta_0 + \beta^T x_i)]\}
$$
Log Likelihood

Log-likelihood of the whole training data $\mathcal{D}$

$$\mathcal{L}(\beta_0, \beta) = \log P(\mathcal{D}|(\beta_0, \beta))$$

$$= \sum_i \{ y_i \log \sigma(b + \beta^T x_i) + (1 - y_i) \log[1 - \sigma(b + \beta^T x_i)] \}$$

It is convenient to work with its negation

$$\mathcal{N}(\beta_0, \beta) = -\sum_i \{ y_i \log \sigma(\beta_0 + \beta^T x_i) + (1 - y_i) \log[1 - \sigma(\beta_0 + \beta^T x_i)] \}$$
Shorthand notation

This is for convenience

- Append 1 to $x$
  \[\tilde{x} \leftarrow [1 \ x_1 \ x_2 \ \cdots \ x_m]\]
- Append $b$ to $\beta$
  \[\tilde{\beta} \leftarrow [\beta_0 \ \beta_1 \ \beta_2 \ \cdots \ \beta_m]\]
- The negative log-likelihood is then
  \[NLL(\tilde{\beta}) = - \sum_i \{y_i \log \sigma(\tilde{\beta}^T \tilde{x}_i) + (1 - y_i) \log[1 - \sigma(\tilde{\beta}^T \tilde{x}_i)]\}\]
Find the optimal parameters for logistic regression

We will maximize the likelihood

\[ \tilde{\beta} = \arg\max_{\beta} \mathcal{L}(\tilde{\beta}) \]
\[ = \arg\min_{\beta} \mathcal{N}\mathcal{L}(\tilde{\beta}) \]

However, this function is complex and we cannot find the simple solution as we did in linear regression. So we need to use numerical methods.

- Numerical methods are messier, in contrast to cleaner analytic solutions.
- In practice, we often have to tune a few optimization parameters — patience is necessary.
An overview of numerical methods

- Gradient descent (our focus in lecture): simple, especially effective for large-scale problems
- Newton’s method: classical and powerful method

Gradient descent is often referred to as an *first-order* method as it requires computation of gradients (i.e., the first-order derivative) of the function.

In contrast, Newton’s method is often referred as to an *second-order* method (as it requires second derivatives).
Gradient Descent

Start at a random point

\[ f(w) \]

\[ w_0 \]

\[ w^* \]

\[ w_0 \]
Gradient Descent

Start at a random point
Determine a descent direction

\[ f(w) \]

\( w \)

\( w_0 \)

\( w^* \)
Gradient Descent

- Start at a random point
- Determine a descent direction
- Choose a step size

![Graph](image)
Gradient Descent

Start at a random point

Determine a descent direction
Choose a step size
Update
Gradient Descent

Start at a random point

Repeat
  - Determine a descent direction
  - Choose a step size
  - Update

Until stopping criterion is satisfied
Gradient Descent

Start at a random point

Repeat
  - Determine a descent direction
  - Choose a step size
  - Update

Until stopping criterion is satisfied
Gradient Descent

Start at a random point

**Repeat**
- Determine a descent direction
- Choose a step size
- Update

**Until** stopping criterion is satisfied
Gradient Descent

Start at a random point

Repeat

- Determine a descent direction
- Choose a step size
- Update

Until stopping criterion is satisfied
Gradient Descent

Start at a random point

Repeat
  - Determine a descent direction
  - Choose a step size
  - Update
Until stopping criterion is satisfied

Gradient Descent

$f(w)$

$w^*$

$w_1$

$w_2$

$w_0$

$w$
Gradient Descent

Start at a random point

**Repeat**
- Determine a descent direction
- Choose a step size
- Update

**Until** stopping criterion is satisfied
Gradient Descent

Where Will We Converge?

$f(w)$ Convex
Any local minimum is a global minimum

$g(w)$ Non-convex
Multiple local minima may exist

Least Squares, Ridge Regression and Logistic Regression are all convex!
Gradient Descent

Update Rule: 
\[ w_{i+1} = w_i - \alpha_i \frac{df}{dw}(w_i) \]

Choosing Descent Direction (1D)

We can only move in two directions
Negative slope is direction of descent!

Step Size
Negative Slope
Gradient Descent

Choosing Descent Direction

We can move anywhere in $\mathbb{R}^d$

Negative gradient is direction of steepest descent!

2D Example:
• Function values are in black/white and black represents higher values
• Arrows are gradients

Update Rule: $w_{i+1} = w_i - \alpha_i \nabla f(w_i)$
Convex function

A function $f : \mathbb{R} \rightarrow \mathbb{R}$ is convex if $f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$ for all $x, y$ and for all $0 \leq \lambda \leq 1$.

If $f$ is twice-differentiable, $f$ is convex iff $\frac{d^2 f(x)}{dx^2} \geq 0$. 

![Convex and Non-convex functions](image)
Remarks on convexity

- Numerical optimization uses local information (function value, gradient...) to search for optimum.
  - Leads to local optimum.
- For a convex function, local optimum is the global optimum.
  - Convex functions are “easy” to minimize.
Convexity for multivariate functions

A function \( f : \mathbb{R}^m \rightarrow \mathbb{R} \) is convex if
\[
f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)
\]
for all \( x, y \in \mathbb{R}^m \) and for all \( 0 \leq \lambda \leq 1 \).

A twice-differentiable function \( f \) is convex iff its Hessian \( \nabla^2 f(x) \) is positive semidefinite.

\[
\nabla^2 f(x) \succeq 0 \equiv \mathbf{v}^T \nabla^2 f(x) \mathbf{v} \geq 0, \text{ for all } \mathbf{v}
\]

\[
\nabla^2 f(x) = \begin{bmatrix}
\frac{\partial^2 f(x)}{\partial x_1^2} & \ldots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_m} \\
\frac{\partial^2 f(x)}{\partial x_2^2} & \ldots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_m} \\
\vdots & \ddots & \vdots \\
\frac{\partial^2 f(x)}{\partial x_1 \partial x_m} & \ldots & \frac{\partial^2 f(x)}{\partial x_m^2}
\end{bmatrix}
\]
Example: \( \min f(\theta) = 0.5(\theta_1^2 - \theta_2)^2 + 0.5(\theta_1 - 1)^2 \)

- We compute the gradients

\[
\frac{\partial f}{\partial \theta_1} = 2(\theta_1^2 - \theta_2)\theta_1 + \theta_1 - 1 \\
\frac{\partial f}{\partial \theta_2} = -(\theta_1^2 - \theta_2)
\]

- Use the following \textit{iterative} procedure for \textit{gradient descent}

1. Initialize \( \theta_1^{(0)} \) and \( \theta_2^{(0)} \), and \( t = 0 \)

2. do

\[
\theta_1^{(t+1)} \leftarrow \theta_1^{(t)} - \eta \left[ 2(\theta_1^{(t)})^2 - \theta_2^{(t)} \right] \theta_1^{(t)} + \theta_1^{(t)} - 1 \\
\theta_2^{(t+1)} \leftarrow \theta_2^{(t)} - \eta \left[ -(\theta_1^{(t)})^2 - \theta_2^{(t)} \right]
\]

\[ t \leftarrow t + 1 \]

3. until \( f(\theta^{(t)}) \) \textit{does not change much}
Gradient descent

\[ \theta^{(t+1)} \leftarrow \theta^{(t)} - \eta \nabla f(\theta^t) \]

Remarks

• \( \eta \) is often called *step size* – literally, how far our update will go along the direction of the negative gradient

• Note that this is for *minimizing* a function, hence the subtraction \((-\eta)\)

• With a *suitable* choice of \( \eta \), the iterative procedure converges to a stationary point where

\[ \nabla f(\theta) = 0 \]

• A stationary point is only necessary for being the minimum (though we’re happy when the function is convex)
Seeing in action

Choosing the right $\eta$ is important

small $\eta$ is too slow? 

large $\eta$ is too unstable?
Choosing the right $\eta$ is important

- small $\eta$ is too slow?
- large $\eta$ is too unstable?
Choosing the right $\eta$ is important

small $\eta$ is too slow?  

large $\eta$ is too unstable?
Gradient Descent Update for Logistic Regression

**Simple fact: derivatives of \( \sigma(a) \)**

\[
\frac{d \sigma(a)}{d a} = \frac{d}{d a} \left(1 + e^{-a}\right)^{-1} = \frac{-(1 + e^{-a})'}{(1 + e^{-a})^2}
\]

\[
= \frac{e^{-a}}{(1 + e^{-a})^2} = \frac{1}{1 + e^{-a}} \frac{e^{-a}}{1 + e^{-a}}
\]

\[
= \sigma(a)[1 - \sigma(a)]
\]
Gradients of the negative log-likelihood

**Negative log-likelihood**

\[ \mathcal{NLL}(\tilde{\beta}) = - \sum_i \{ y_i \log \sigma(\tilde{\beta}^T \tilde{x}_i) + (1 - y_i) \log[1 - \sigma(\tilde{\beta}^T \tilde{x}_i)] \} \]

**Gradients**

\[
\frac{\partial \mathcal{NLL}(\tilde{\beta})}{\partial \tilde{\beta}} = - \sum_i \left\{ y_i [1 - \sigma(\tilde{\beta}^T x_i)] x_i - (1 - y_i) \sigma(\tilde{\beta}^T x_i) x_i \right\} \\
= - \sum_i \left\{ y_i - \sigma(\tilde{\beta}^T x_i) \right\} x_i
\]

**Remark**

- \( e_i = \left\{ y_i - \sigma(\tilde{\beta}^T x_i) \right\} \) is called *error* for the \( i \)th training sample.
Gradients of the negative log-likelihood

**Negative log-likelihood**

\[
\mathcal{NLL} (\tilde{\beta}) = - \sum_i \{ y_i \log \sigma(\tilde{\beta}^T \tilde{x}_i) + (1 - y_i) \log[1 - \sigma(\tilde{\beta}^T \tilde{x}_i)] \}
\]

**Gradients**

\[
\frac{\partial \mathcal{NLL} (\tilde{\beta})}{\partial \tilde{\beta}} = - \sum_i \left\{ y_i [1 - \sigma(\tilde{\beta}^T x_i)] x_i - (1 - y_i) \sigma(\tilde{\beta}^T x_i) x_i \right\}
\]

\[
= - \sum_i \left\{ y_i - \sigma(\tilde{\beta}^T x_i) \right\} x_i
\]

**Remark**

- \( e_i = \left\{ y_i - \sigma(\tilde{\beta}^T x_i) \right\} \) is called error for the \( i \)th training sample.
Numerical optimization

**(Batch) Gradient descent**

- Choose a proper step size $\eta > 0$
- Iteratively update the parameters following the negative gradient to minimize the error function

\[
\tilde{\beta}^{(t+1)} \leftarrow \tilde{\beta}^{(t)} + \eta \sum_i \left\{ y_i - \sigma(\tilde{\beta}^T x_i) \right\} x_i
\]

**Remarks**

- The step size needs to be chosen carefully to ensure convergence.
- The step size can be adaptive (i.e. varying from iteration to iteration). For example, we can use techniques such as *line search*
- There is a variant called *stochastic* gradient descent, also popularly used.
Intuition for Newton’s method

Approximate the true function with an easy-to-solve optimization problem

In particular, we can approximate the \( \mathcal{NLL} \) around \( \tilde{\beta}^{(t)} \) by a quadratic function, and then minimize this quadratic function.
Intuition for Newton’s method

Approximate the true function with an easy-to-solve optimization problem

In particular, we can approximate the $\mathcal{NLL}$ around $\tilde{B}^{(t)}$ by a quadratic function, and then minimize this quadratic function.
Approximation

Second Order Taylor expansion around $x^{(t)}$

$$f(x) \approx f(x^{(t)}) + f'(x^{(t)})(x - x^{(t)}) + \frac{1}{2}f''(x^{(t)})(x - x^{(t)})^2$$

Taylor expansion of negative log-likelihood function around $\bar{\beta}^{(t)}$

$$\mathcal{NLL}(\bar{\beta}) \approx \mathcal{NLL}(\bar{\beta}^{(t)}) + (\bar{\beta} - \bar{\beta}^{(t)})^T g^{(t)} + \frac{1}{2}(\bar{\beta} - \bar{\beta}^{(t)})^T H^{(t)}(\bar{\beta} - \bar{\beta}^{(t)})$$

where

- $g^{(t)} \equiv \nabla \mathcal{NLL}(\bar{\beta}^{(t)})$ is the gradient evaluated at $\bar{\beta}^{(t)}$
- $H^{(t)} \equiv \nabla^2 \mathcal{NLL}(\bar{\beta}^{(t)})$ is the Hessian matrix evaluated at $\bar{\beta}^{(t)}$
Approximation

Second Order Taylor expansion around $x(t)$

$$f(x) \approx f(x(t)) + f'(x(t))(x - x(t)) + \frac{1}{2}f''(x(t))(x - x(t))^2$$

Taylor expansion of negative log-likelihood function around $\tilde{\beta}^{(t)}$

$$\mathcal{NLL}(\tilde{\beta}) \approx \mathcal{NLL}(\tilde{\beta}^{(t)}) + (\tilde{\beta} - \tilde{\beta}^{(t)})^T g^{(t)} + \frac{1}{2}(\tilde{\beta} - \tilde{\beta}^{(t)})^T H^{(t)}(\tilde{\beta} - \tilde{\beta}^{(t)})$$

where

- $g^{(t)} \equiv \nabla \mathcal{NLL}(\tilde{\beta}^{(t)})$ is the gradient evaluated at $\tilde{\beta}^{(t)}$
- $H^{(t)} \equiv \nabla^2 \mathcal{NLL}(\tilde{\beta}^{(t)})$ is the Hessian matrix evaluated at $\tilde{\beta}^{(t)}$
So what is the Hessian matrix?

The matrix of second-order derivatives

$$H = \frac{\partial^2 NLL(\tilde{\beta})}{\partial \tilde{\beta} \tilde{\beta}^T}$$

In other words,

$$H_{ij} = \frac{\partial}{\partial \tilde{\beta}_j} \left( \frac{\partial NLL(\tilde{\beta})}{\partial \tilde{\beta}_i} \right)$$

So the Hessian matrix is $\mathbb{R}^{(m+1) \times (m+1)}$, where $\tilde{\beta} \in \mathbb{R}^{(m+1)}$. 
Optimizing the approximation

Minimize the approximation

\[ \mathcal{NLL}(\tilde{\beta}) \approx \mathcal{NLL}(\tilde{\beta}^{(t)}) + (\tilde{\beta} - \tilde{\beta}^{(t)})^T g(t) + \frac{1}{2}(\tilde{\beta} - \tilde{\beta}^{(t)})^T H(t)(\tilde{\beta} - \tilde{\beta}^{(t)}) \]

and use the solution as the new estimate of the parameters

\[ \tilde{\beta}^{(t+1)} \leftarrow \min_{\tilde{\beta}} (\tilde{\beta} - \tilde{\beta}^{(t)})^T g(t) + \frac{1}{2}(\tilde{\beta} - \tilde{\beta}^{(t)})^T H(t)(\tilde{\beta} - \tilde{\beta}^{(t)}) \]

The quadratic function minimization has a \textit{closed} form, thus, we have

\[ \tilde{\beta}^{(t+1)} \leftarrow \tilde{\beta}^{(t)} - \left( H(t) \right)^{-1} g(t) \]

i.e., the Newton’s method.
Optimizing the approximation

Minimize the approximation

\[
\mathcal{NLL}(\tilde{\beta}) \approx \mathcal{NLL}(\tilde{\beta}^{(t)}) + (\tilde{\beta} - \tilde{\beta}^{(t)})^T g^{(t)} + \frac{1}{2} (\tilde{\beta} - \tilde{\beta}^{(t)})^T H^{(t)} (\tilde{\beta} - \tilde{\beta}^{(t)})
\]

and use the solution as the new estimate of the parameters

\[
\tilde{\beta}^{(t+1)} \leftarrow \min_{\tilde{\beta}} (\tilde{\beta} - \tilde{\beta}^{(t)})^T g^{(t)} + \frac{1}{2} (\tilde{\beta} - \tilde{\beta}^{(t)})^T H^{(t)} (\tilde{\beta} - \tilde{\beta}^{(t)})
\]

The quadratic function minimization has a \textit{closed} form, thus, we have

\[
\tilde{\beta}^{(t+1)} \leftarrow \tilde{\beta}^{(t)} - \left( H^{(t)} \right)^{-1} g^{(t)}
\]

\textit{i.e.,} the Newton’s method.
Contrast gradient descent and Newton’s method

Similar

• Both are iterative procedures.

Different

• Newton’s method requires second-order derivatives.
• Newton’s method does not have the magic $\eta$ to be set.
Other important things about Hessian

Our negative log-likelihood function is convex

\[
\frac{\partial NLL(\tilde{\beta})}{\partial \tilde{\beta}} = \sum_i \{\sigma(\tilde{\beta}^T x_i) - y_i\} x_i
\]

\[\Rightarrow H = \frac{\partial^2 NLL(\tilde{\beta})}{\partial \tilde{\beta} \tilde{\beta}^T}\]

For any vector \( v \),

\[v^T H v \geq 0\]

Thus, positive semi-definite. Thus, the objective function is convex, with only one global optimum.

Association studies and regression  Logistic regression  41 / 65
Our negative log-likelihood function is convex

\[
\frac{\partial NLL(\tilde{\beta})}{\partial \tilde{\beta}} = \sum_i \{\sigma(\tilde{\beta}^T x_i) - y_i\} x_i
\]

\[\Rightarrow H = \frac{\partial^2 NLL(\tilde{\beta})}{\partial \tilde{\beta} \tilde{\beta}^T}\]

For any vector \(v\),

\[v^T H v \geq 0\]

Thus, positive semi-definite. Thus, the objective function is convex, with only one global optimum.
Good about Newton’s method

**Fast (in terms of convergence)!**

Newton’s method finds the optimal point in a *single* iteration when the function we’re optimizing is quadratic.

In general, the better our Taylor approximation, the more quickly Newton’s method will converge.
Bad about Newton’s method

Not scalable!

Computing and inverting Hessian matrix can be very expensive for large-scale problems where the dimensionally $m$ is very large. There are fixes and alternatives, such as Quasi-Newton/Quasi-second order method.
• Logistic regression allows prediction of 0 – 1 outputs.
• Need to find the minimum of the negative log likelihood.
• Unlike linear regression, no analytical solution.
• We use numerical optimization.
  • $\mathcal{N LL}$ is convex
  • Gradient descent (first-order)
  • Newton’s method (second-order).
  • Quasi-Newton
Outline

Logistic regression
  General setup
  Maximum likelihood estimation
  Gradient descent
  Newton’s method

Application to GWAS
Why does GWAS approach work?

Before we look at associating genotype to phenotype, we first need to understand genetic variation.
Some definitions

Frequency of reference allele at a SNP \( j \) in a population

\[
p_j = \frac{\sum_i x_{i,j}}{2n}
\]

Frequency of alternate allele \( 1 - p_j \)

Locus 1: C is reference

Frequency of reference allele 0.25
Frequency of alternate allele 0.75
Linkage

- Recombination (more precisely, lack of recombination) leads to linkage, i.e. positions close on the chromosome will be inherited together
- Alleles at nearby loci will be correlated
• Correlation or non-independence between the alleles at a set of loci
• For a pair of SNPs, coefficient of LD

\[ D = \text{Cov}[X_1, X_2] = p_{1,1} - p_1p_2 \]
How does this relate to association studies?

We can narrow down the genetic region associated with disease if our SNP is in LD with the causal mutation.
Linkage Disequilibrium

- How does this relate to association studies?
- Instead of testing all possible SNPs, find a set of tag SNPs that are in strong LD with other SNPs
- Reduces the cost of genotyping per individual
- There is a loss in power because the tested SNP may not correlate exactly with the causal SNP.
- For this to work, need to understand how far LD extends in the human genome
Linkage Disequilibrium

LD is not uniform along the genome.

- Has block-like patterns
- Blocks range from few to 100kb
- Previous works assumed this was uniform and most LD < 3kb
- Active research area to understand why.
LD is not uniform along the genome.

- What is the practical implication?
- If LD blocks are big, picking one SNP to tag each block might be enough
- HapMap project
- Very large-scale GWAS based on genotyping arrays
- Genotype 500K SNPs chosen to tag HapMap SNPs
Back to GWAS
Single SNP association testing

G is reference allele. AA=0, AG=1, GG=2

\[ Y_i = \beta_0 + \beta_1 x_i + \epsilon_i, \quad \epsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2) \]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard error</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\hat{\beta}_0)</td>
<td>-0.04</td>
<td>0.13</td>
<td>0.76</td>
</tr>
<tr>
<td>(\hat{\beta}_1)</td>
<td>-0.10</td>
<td>0.16</td>
<td>0.53</td>
</tr>
</tbody>
</table>
Back to GWAS

Single SNP association testing

\[ Y_i = \beta_0 + \beta_1 x_i + \epsilon_i, \quad \epsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2) \]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard error</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \hat{\beta}_0 )</td>
<td>-0.04</td>
<td>0.13</td>
<td>0.76</td>
</tr>
<tr>
<td>( \hat{\beta}_1 )</td>
<td>-0.10</td>
<td>0.16</td>
<td>0.53</td>
</tr>
</tbody>
</table>

G is reference allele. AA=0, AG=1, GG=2

\( G \) is reference allele. AA=0, AG=1, GG=2

\[ Y_i = \beta_0 + \beta_1 x_i + \epsilon_i, \quad \epsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2) \]
Back to GWAS
Single SNP association testing

G is reference allele. AA=0, AG=1, GG=2

\[ Y_i = \beta_0 + \beta_1 x_i + \epsilon_i, \quad \epsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2) \]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard error</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \hat{\beta}_0 )</td>
<td>-0.39</td>
<td>0.11</td>
<td>0.001</td>
</tr>
<tr>
<td>( \hat{\beta}_1 )</td>
<td>0.67</td>
<td>0.12</td>
<td>3.51 \times 10^{-7}</td>
</tr>
</tbody>
</table>
Back to GWAS

Single SNP association testing

$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i, \quad \epsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard error</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\hat{\beta}_0$</td>
<td>-0.39</td>
<td>0.11</td>
<td>0.001</td>
</tr>
<tr>
<td>$\hat{\beta}_1$</td>
<td>0.67</td>
<td>0.12</td>
<td>$3.51 \times 10^{-7}$</td>
</tr>
</tbody>
</table>
GWAS
Other models

True model: \( \beta_0 = 1, \beta_1 = 0.5, \beta_2 = 0.25 \)

- Additivity might not hold
- Code each genotype as a distinct covariate.
- For \( k \) genotypes, we need \( k - 1 \) covariates.

\[
\mathbb{E} [Y | x] = \beta_0 + \beta_1 \mathbf{1}\{x = AG\} + \beta_2 \mathbf{1}\{x = GG\}
\]
True model: $\beta_0 = 1, \beta_1 = 0.5, \beta_2 = 0.25$

- Test the null $H_0 : \beta_1 = 0, \beta_2 = 0$
- Likelihood-ratio test of the full model against the null is asymptotically $\chi^2$-squared distributed with 2 degrees of freedom.
GWAS

Other models

True model: $\beta_0 = 1, \beta_1 = 0.5, \beta_2 = 0.25$

<table>
<thead>
<tr>
<th>Parameter $\hat{\beta}$</th>
<th>Estimate</th>
<th>Standard error</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\hat{\beta}_0$</td>
<td>0.98</td>
<td>0.05</td>
<td>$2 \times 10^{-16}$</td>
</tr>
<tr>
<td>$\hat{\beta}_1$</td>
<td>0.45</td>
<td>0.09</td>
<td>$2.1 \times 10^{-6}$</td>
</tr>
<tr>
<td>$\hat{\beta}_2$</td>
<td>0.29</td>
<td>0.18</td>
<td>0.12</td>
</tr>
</tbody>
</table>
Which model to apply?

GWAS
Preprocessing and quality control

- Missing genotype data
- Test of phenotype normality (for quantitative phenotypes)
Missing genotypes

- Ignore: loss in power
- Impute missing genotypes
  - Can use LD \textit{i.e.}, correlations among SNPs, to predict missing genotypes
  - LD can be inferred from reference panels of haplotypes, \textit{e.g.} HapMap, 1000 Genomes

Hoffman and Witte, TIG, 2010
Confounders

- Age, sex
- Batch or processing effects
- Population structure

A number of approaches that we will discuss later
• Under the null, P-values are uniformly distributed.

• Assumption: most SNPs in the genome have no effect on phenotype

• P-values across SNPs uniformly distributed
GWAS
Model checking

- Q-Q plot
- Plot $p(i)$ against $-\log \left( \frac{i}{m+1} \right)$. 
• Control the type-I error of each SNP-phenotype test at a significance level $\bar{\alpha}$.

• FWER: Probability of one or more type-I errors in a family of tests.

• Would like to bound $FWER \leq \alpha$. 
Bonferroni procedure

Set $\tilde{\alpha} = \frac{\alpha}{n}$

$$FWER = \Pr\{\bigcup_{i \in \{\text{set of true null hypotheses}\}} \text{Reject hypothesis } i \text{ at level } \tilde{\alpha}\}$$

$$\leq \sum_{i \in \{\text{set of true null hypotheses}\}} \Pr_0\{\text{Reject hypothesis } i \text{ at level } \tilde{\alpha}\}$$

$$= n \frac{\alpha}{n}$$

$$\leq \alpha$$
**Bonferroni procedure**

- What is $n$?
- $n$: number of SNPs tested
- Typical $n = 1$ million, $\Rightarrow \tilde{\alpha} = 5 \times 10^{-8}$.
  - Pros: computationally efficient. Data-independent.
  - Cons: Very conservative. SNPs are correlated so that the tests are not independent.
  - In an extreme case, assume perfect linkage among all SNPs. Then $FWER = \tilde{\alpha}$.
- Set $n = n_{eff}$, where $n$ is the effective number of tests. Determine $n_{eff}$ empirically.
Alternatives

- Permutation procedure
  - Pros: Provides accurate control of FWER.
  - Cons: Computationally expensive. Data-dependent

- False Discovery Rate
WTCCC
Manhattan plot

Association studies and regression
Application to GWAS
62 / 65
WTCCC
Q-Q plot

Association studies and regression
Beyond GWAS

If the GWAS shows a significant association:

- Replication
  - Needed to rule out artefacts in the discovery
  - Get accurate estimates of effect sizes

- Fine-mapping
  - LD useful for GWAS
  - LD hurts localization

- Understand biology
• To understand genotype-phenotype map, useful to study populations of individuals. The approach of GWAS.
• Enabled by LD across the genome.
• Single SNP regression the first step (Linear or logistic). Inference is easy (convex).
• Model checking critical. Confounders, deviations of the phenotype distribution, missing data, and multiple testing are challenges.