Spring 2025

CS 2840

Pranav Mahableshwarkar

Instructor: Sorin Istrail

LECTURE 1: 1/23/2025

Introduction

- Human genome contains ~ 3G basepairs → 46 chromosomes.
- 2 individuals are 99% the same → difference = 10M basepairs
- GWAS Task: Finding, computationally, specific regions of shifts.

Definition: A SNP – single nucleotide polymorphism – a difference in one basepair occurs once every 600 bp. Most SNPs are common → in 2–3% of the population.

Genomic Foundations

We will consider GWAS, protein folding, Linkage Disequilibrium, and more in this course.

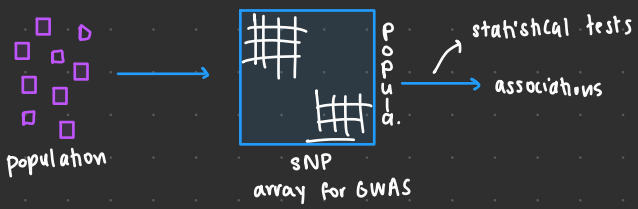
ALGORITHMS

- MLE and E-M Algorithms, covering topics of spectral graph theory, to examine the substructures of populations.
- ex: there are 2600+ subpopulations within in India
- protein folding / misfolding, AlphaFold, → applied to mad cow disease, for example

AN INTRO TO Genome Wide Association Studies (GWAS)

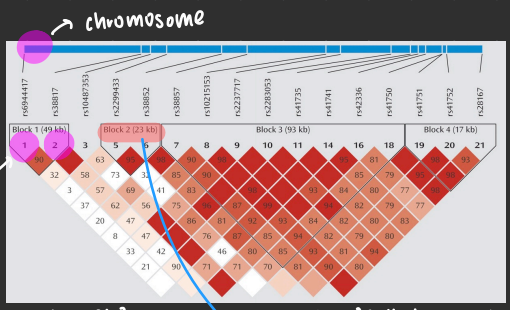
Before GWAS, disease context was random and non-rigorous.

- we now can have 100,000s of data points → low cost of sequencing
- can be applied to unrelated people over many years, allowing for more subtle detections



Defining GWAS

- method for interrogating all 10 million variable points
- Infinite Allele Statement: There can only ever be 2 letters at a position, every SNP is binary this is because the genome is so large, multiple mutations @ 1 pt. uncommon



our SNPs are arranged in blocks (haplotype blocks) that allow us to group them

↓

further studied in linkage disequilibrium.

- After GWAS we can use SNPs to inform the risk for disease.
- We can also identify gene pathways from significant GWAS SNP hits.

What is Disease?

The naive thought is a broken protein causing a cascade of issues (monogenic). This is NOT the case for most complex diseases.

SNPs in the Genome

Definition: A position in the genome at which two or more diff bases in the genome occur, each with a frequency ~ 1%

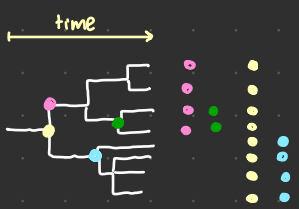
The most abundant type of polymorphism.

Haplotype: G [C/G] T C G [A/T] C A A C A [G/C] — the haplotype is the sequence of the SNPs: CAG

From this, haplotypes are our alleles that we can binarize due to the infinite sites assumption. → our matrix is entirely binary

Recombination: When new sequences are generated from 2 sequences: p_1p_2, q_1q_2 → p_1q_2, q_1p_2

LINKAGE DISEQUILIBRIUM



- variations in chromosome populations over time
- easily visualized through phylogenetic trees

- **definition:** a way of statistically defining disease associations.

LECTURE 2: GWAS computational Pipeline

Goal: discover patterns of SNPs associated with the disease and not the control.

Note that there will be patterns of SNPs on ALL chromosomes!!

Chapter 1: Haplotype Phasing Problem

Every study participant → genotyped → SNP Array

Humans are diploid: mother chrom 1, 2, ..., 22 sex chromosome: female XX 23
father chrom 1, 2, ..., 22 male XY 24

SNP = single nucleotide polymorphism

↳ SNP has 2 alleles: the allele from the mother and the allele from the father

Definition: A haplotype is a single DNA sequence representing the SNPs that occur every ~500-600 bp or so

SNP 1

Say from the mother we have: A A T T
father we have: G A G A

in practice it is very challenging to construct the haplotypes because we only have the pairs, we don't know which parent the individual nucleotides originated from.

Haplotype Phasing comes down to inferring the mother father haplotypes.

★ WITH ONE INDIVIDUAL THIS IS IMPOSSIBLE ★

COMBINATORICS of HAPLOTYPES:

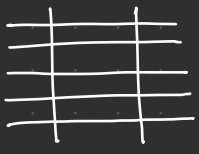
major minor alleles

Definitions: a genotype is a string over $\{0,1,2\}$, a haplotype: is a string over $\{0,1\}$

- the genotype is made up of two haplotypes!

two haplotypes = $\begin{Bmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{Bmatrix} = 0\ 1\ 2\ 0$

ambiguous, different.



this is a SNP if: there is high % of people that have it. Recall that due to the infinite sites model, only two nucleotides are possible at a given position. i.e AA, GA, AG, or GG

How does phasing get complicated?

0 2 2 0 → 0 1 0 0 / 0 0 1 0 / 0 1 1 0 / 0 0 0 0 } there are 2 explanations!, if there are 3 2s in the genotype we will have 2^2 possibilities.

0 2 2 1 2 → 1: 0 0 0 1 0 / 0 1 1 1 1 ; 2: 0 0 0 1 1 / 0 1 1 1 0 → 3 ... → this generalizes to 2^{n-1} explanations for n ambiguities

The Haplotype Phasing Problem

- Input: $g_1, g_2, \cdots, g_n$
- Output: $h_{11}, h_{12}, \cdots, h_{n1}, h_{n2}$ → all the haplotypes for all of the given genomes.

A A T G C C G T T → A A [T/C] G [A/C] C [G/A] T T → geno: 0 0 2 0 2 1 2 0 0

0 0 1 0 1 1 1 0 0

0 0 0 0 0 1 0 0 0, the Clark Rule haplotype

now we have two haplotypes

Where do we start haplotyping?

When you have unambiguous genome snippets or single ambiguity in the snippet, you have good start pts

1. g_1 = ATTCT < ATTCT / ATTCT
2. g_2 = AT [G/A] CT < ATGCT / ATACT

3 ISSUES IN THE CLARK REGIME

1. No full homozygote or single SNP
2. We arrive at a point where there are no new updates we can make, and there are no remaining candidates. The remaining genomes are called orphans.
3. We find incorrect haplotypes. This rule is heuristic and does not guarantee exceptions. Thes errors are called anamolous haplotypes.

g_1 = 0 1 2 2 2

g_2 = 0 0 2 1 2

g_3 = 0 0 0 0 0 < h_1 : 0 0 0 0 0 / h_1 : 0 0 0 0 0

g_4 = 0 2 1 1 1 ↘ h_2 : 0 0 1 1 1 / h_3 : 0 1 1 1 1 } these are automatically phased.

Applying $h_3 \to g_1$ → 0 1 1 1 1 → h_4: 0 1 0 0 0 / 0 1 2 2 2

Applying $h_2 \to g_2$ → 0 0 1 1 1 → h_5: 0 0 0 1 0 / 0 0 2 1 2

} now we have explained all of the genotypes using Clark's Rule.

LECTURE 3: The Haplotype Phasing Problem

Chapter 1 Outline
- 1.1 Clark Algorithm (greedy)
- 1.2 The E-M Algorithm (ML, maximum likelihood)
- 1.3 The Parsimony Algorithm
- 1.4 The ML-Phasing and Parsimony Phasing

FORMALIZING the CLARK PROCEDURE

(one '2' in genotype.)

1. Identify all homozygotes and single hetero-zygotes (Aa, aA) haplotypes. Phase them and call these **resolved genotypes.** [See Lecture 2 Notes for examples]
2. Determine whether any of the unsolved haplotypes can be used to solve any of the remaining ambiguous genotypes using the **Clark Rule.**
3. Iteratively continue until we have resolved all genotypes.

3 ISSUES: See Lecture 2 Notes for these enumerated issues

E-M Algorithm (1.2)

Goal: Finding the haplotype frequencies in a population and the maximum likelihood haplotype phasing.

Define: The **E-M Algorithm** is an **iterative algorithm** to compute successive sets of haplotype frequencies.

1. $p_1, p_2, \ldots p_T$ starting on initial $p_1^0, p_2^0, \ldots, p_T^0$ (2^n haplotypes, there are a lot!)
2. These initial values are used as if they are unknown true frequencies to estimate the **explanation frequencies.** $\Rightarrow P(h_k h_e)^0$, expectation step
   (this a genome explanation for a genotype.)
3. The expected explanation frequencies are used in turn to estimate haplotype frequencies at the next step – maximization step $\rightarrow p_1', p_2', \ldots, p_T'$

   continue until convergence → when consecutive steps yield minimal changes.

CONSTRUCTION + ALGORITHM PROCESS

definition: a genotype is a multi-locus genotype whose multi-locus haplotype phase is unknown
an explanation is a particular combination of 2 haplotypes, explaining a genotype

In E-M, we ultimately are trying to define a model that **maximizes the probability** or **likelihood** of observing the given data (genotypes)

Likelihood Function

Ultimately, we can/will only estimate frequencies over known/seen phenotypes due to computational limitations.

ASSUMPTION: The distribution of the given sample is multinomial w.r.t latent and unknown frequencies $P_1, P_2, \dots$

$$\text{Thus} \quad \mathbb{P}(\text{sample} \mid P_1, \dots, P_m) = \frac{n!}{n_1! n_2! \cdots n_m!} \times \prod_{i=1}^{m} P_i^{n_i}$$

(num individuals)

Here, n_i : represent counts for the m unique genotypes.

The number of explanations (c_j) for s_j heterozygous loci can be defined as follows: (note that this UNIQUE to 1 genotype.)

$$c_j = 2^{s_j - 1}$$

→ this is clearly seen above in our Clark's Rule examples (see above)

Putting this together, P_j : the probability of the j^{th} genotype is the following. Let us also add the following notation: $H_j = \{h_k, h_\ell\}$ where $|H_j| = c_j$ is the set of all explan. for a given phenotype.

$$P_j = \sum_{i=1}^{c_j} \mathbb{P}(\text{explanation } i) = \sum_{h_k, h_\ell \in H_j} \mathbb{P}(h_k h_\ell)$$

→ recall that 2 haplotypes make up every explanation.

Here: $\mathbb{P}(h_k h_\ell) = p_k^2$ if $k = \ell$, $2 p_k p_\ell$ if $k \neq \ell$.

Putting everything together:

$$\text{Likelihood} = L(\underbrace{p_1, p_2, \dots, p_h}_{\text{haplotype freqs sum to 1.}}) = a_1 \prod_{j=1}^{m} \left(\sum_{h_k, h_\ell \in H_j} \mathbb{P}(h_k h_\ell) \right)^{n_j}$$

(n_j → number of occ. of genotype j; a_1 → multinomial constant)

How do we optimize this?

Normally, with Maximum Likelihood Estimation, we take partials w.r.t our model, set it to 0 and find our optima. Here, this would result in h-1 equations:

$$U_t = \text{score of the } t^{th} \text{ haplotype} = \frac{\partial \log L}{\partial p_t} = \sum_{j=1}^{m} \frac{n_j}{P_j} \frac{\partial P_j}{\partial p_t} = \sum_{j=1}^{m} \frac{n_j}{P_j} \sum_{h_\ell \in H_j} \left(2 - \mathbb{1}_{k=\ell}\right) p_\ell$$

(log for numerical stability)

Setting these all to 0 and solving would be very tedious. Thus, enter the

E-M Algorithm!

E-M Algorithm

The goal, as aforementioned, is to compute successive sets of haplotype frequencies $p_1 \cdots p_m$

- These frequencies are used to estimate explanation frequencies [E-STEP]
- Then, these are used to update haplotype frequencies for the next iteration, M-step

First, we have to initialize the haplotype frequencies: $p_1^{(0)}, p_2^{(0)}, \ldots, p_m^{(0)}$

- This could be uniformly distributed: $IP(\text{expl. } h_k h_e \text{ for } g_j) = P_j(h_k h_e)^{(0)} = \frac{1}{c_j}$
- other initial conditions include $p_i = p_j$ for all i, j (equal IP over haplotypes) which represents complete linkage equilibrium
- haplotype frequencies chosen at random.

For these notes, we take the first approach.

Expectation Step

At the t^{th} iteration, we use the previous iterations' h frequencies. to determine the IP of resolving each genotype into possible explanations.

$$IP_j(h_k h_e)^{(t)} = \frac{n_j}{n} \frac{IP(h_k h_e)^{(t)}}{IP_j(\quad)}$$

interpretation: weighted cond. probability

function of $p_k, p_e \ldots$

We can understand this as calculating the expected frequency of every explanation for each genotype.

Maximization Step

We then use the gene counting method for optimization.

$$p_i^{(t+1)} = \frac{1}{2} \sum_{j=1}^{m} \sum_{\underbrace{h_k h_e}_{z \text{ iterable}} \in H_j} P_j(h_k h_e)^{(t)} \delta_{zi}$$

Here, $\delta_{zi} \in \{0, 1, 2\}$ represents the no. of times haplotype i is present in explan z

☆ derivation of this will appear in HW1 ☆

This derivation will later appear in the notes.

DERIVATION OF M-STEP USING LAGRANGE MULT.

Recall: Classical Expectation Maximization

We have a statistical model that generates data X, a set of unobserved data Z, and a set of unknown or missing parameters θ.

These yield the likelihood function: $L(\theta; X, Z) = p(X, Z \mid \theta)$. The maximum likelihood estimate is determined by maximizing the probability of observing the given data.

$$\max L(\theta; X) = p(X \mid \theta) = \int p(X, Z \mid \theta)\, dz \rightarrow$$ or by taking partial derivatives and using analytical optimization methods

Expectation Step:

Define $Q(\theta \mid \theta^{(t)}) = \mathbb{E}[\log \text{likelihood function}]$ w.r.t Z given $X, \theta^{(t)}$

- $Q(\theta \mid \theta^{(t)}) = \mathbb{E}_{Z \sim p(\cdot \mid X, \theta^{(t)})}[\log p(X, Z \mid \theta)]$

think of this as your expected likelihood over all possible latent variables Z

Maximization Step:

$$\theta^{t+1} = \underset{\theta}{\operatorname{argmax}}\, Q(\theta \mid \theta^{t}) \rightarrow$$

NOTE: this step is often performed by analytically computing the maximum value via lagrange multipliers.

APPLICATION to HAPLOTYPE PHASING

Here, $X \rightarrow$ our observed genotypes $Z \rightarrow$ the explanations for our genotypes (unobserved)

$\theta \rightarrow p_0, \ldots, p_T$ for T haplotypes

Recall our likelihood function:

$$L(p_1, \ldots, p_T) = a_i \prod_{j=1}^{m} \left(\sum_{h_k, h_e \in H_j} P(h_k h_e) \right)^{n_j}$$

But what does this really represent?

LECTURE 4:

02/04/2025

For the complete worked through EM example, please see the slideshows course website.

Example:

Say we only have genotypes: 22, 02, 20, 00, 11

↳ notice that there are missing genotypes → this is okay!!! You rarely observe all possible g.

This means the only possible haplotypes are: 00, 01, 10, 11

↳ let us denote these as $\theta_{00}, \theta_{01}, \theta_{10}, \theta_{11}$ → which will be iteratively determined in E-M.

↳ there are n_A genotypes of 22, n_B for 02, etc.

Now, for group A, for example:

$$\mathbb{P}(Y_A)^{(t)} = P_A^{(t)} = \sum_{h_k, h_\ell \in H_A} \mathbb{P}(h_k h_\ell) = \sum_{h_k, h_\ell \in H_A} (2 - \delta_{k\ell}) P_k P_\ell$$

I used $\mathbb{1}$ in Lecture 3.

$\mathbb{1}_{k=\ell}$ = 00/11 or 01/10 (for 22)

$$= 2\theta_{00}^{(t)}\theta_{11}^{(t)} + 2\theta_{01}^{(t)}\theta_{10}^{(t)}$$

Now in the E-Step, we want to calculate the Expected Number of Each Haplotypes

- $n_{00}^{(t+1)} = n_A P\left(\frac{00}{11} \mid Y_A\right) + n_B + n_C + 2n_D + 0n_E$

  $= 2\theta_{00}^{(t)}\theta_{11}^{(t)} / P_A$

- $\theta_{00}^{(t+1)} = n_{00}^{(t+1)} / 2n$ where $2n$ = total number of haplotypes

1.3 Maximum Likelihood Phasing

Define Haplotypes: Sequences over $\{0, 1\}$ Genotypes: $\{0, 1, 2\}$

Let us consider the Likelihood Function:

$$L(P_1, P_2, \ldots, P_T) = (P_1 P_2 + P_3 P_4)(P_1 P_5 + P_4 P_6)(P_1 P_1)(P_7 P_8)$$

T = total number of haplotypes, example over 4 genotypes.

We want to find the $P_{1:8}^{opt}$ that maximizes the Likelihood function. This is a computationally infeasible problem.

Parsimony: the smallest number of haplotypes involved. Note that in practice finding these values is done by simplifying the polynomial optimization.

LECTURE 5

02/06/2025

TWO OPTIMIZATIONS (that we can exactly solve):

1. We have $X_1, X_2, \dots, X_r \in \mathbb{R}, > 0$, $\sum_{i=1:r} x_i$ = some constant, if x_i are probabilities ↳ sum to 1.

   Problem: the solution maximizing the product $P = \Pi x_i$

   Solution: optimal → $x_1 = x_2 = \cdots = x_n$

   This used in the idea of equal likelihood → if we wanted to maximize.

2. Find the solution to: $\max(p' = x_1^{n_1} x_2^{n_2} \cdots x_n^{n_n})$ → optimal when $\frac{x_1}{n_1} = \frac{x_2}{n_2} = \cdots = \frac{x_n}{n_n}$

   This is used in renormalization within some E-M applications... see more in CS1820!

NP-COMPLETENESS PROOF for HP/PARSIMONY

Theorem: parsimony haplotype phasing is NP-hard and maximum-likelihood hap phasing is NP-hard

SNPs are biallelic → $\{0, 1\}$: Hubbel denotes genomes differently → h_1: 0 1 0, h_2: 0 1 1 $\xrightarrow{+}$ g: 0 2 1

where 1 is now the ambiguous symbol.

PARSIMONY PHASING PROBLEM:

Given $G = \{g_1, g_2, \dots, g_n\}$ of observed genotypes, find the minimal size of inferred haplotypes $H = \{h_1, h_2, \dots, h_k\}$ such that every genotype can be represented as a sum of two haplotypes.

We show that every instance of a general graph R may be connected into a set of genotypes so that a minimum set of inferred haplotypes corresponds to a minimum clique partition of R. ← this is an NP-hard problem!

What is a Clique? R_1 • R_2 — R_3 △ R_4 

A graph where every node is connected to every node.

Clique Partition? A partition of a graph into cliques that cover the entire graph. Minimizing the number of cliques is NP-hard. If we can reduce our problem to this, then our parsimony phasing must also be NP-hard

REDUCTION

First we must generate the graph $\mathcal{G} = (V, E) \Leftrightarrow G = \{g_1, g_2, \dots, g_n\}$ the genotypes set

Algorithm: Given genotypes of length 2n.

| **Notation:** $g_{ij} = g_i[j]$ the j^{th} coordinate of the i^{th} genotype.

- We will have $V_{1:n}$ vertices,
- $i = 1:n$: Then, $g_{ij} = 2$ if $i = j$, $g_{ij} = 1$ if V_i is connected by an edge to V_j
- $i = n+1 : 2n$: Create an identity matrix where
  - $g_{ij} = 1$ if $j = n+i$, 0 otherwise, $1 \le i \le n$, $n+1 \le j \le 2n$

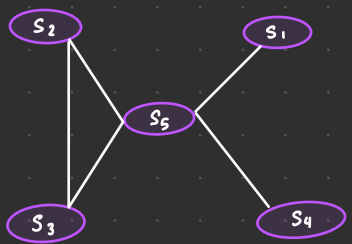
Example:

★ GRAPH CAME FIRST ★

Vertex	Genotype	identity matrix ↗
S_1	$g(S_1) = 20001$	1 0 0 0 0
S_2	$g(S_2) = 02101$	0 1 0 0 0
S_3	$g(S_3) = 01201$	0 0 1 0 0
S_4	$g(S_4) = 00021$	0 0 0 1 0
S_5	$g(S_5) = 11112$	0 0 0 0 1

these are the original genotypes and define edges

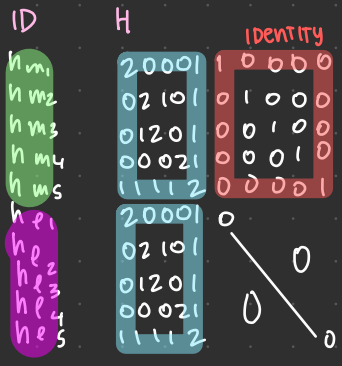
↳ these ensure uniqueness for each $g(S_i)$



Now let us define the haplotype set $H = h_1, h_2, \dots, h_T$ where we can write that $g_{i,n+i} = h_{e,n+i} + h_{m,n+i}$ → recall that $n+i \to 2n$ is our identity matrix.

- Then, $h_{e,n+i}$ or $h_{m,n+i} = 1$ ★ BUT NOT BOTH ★

WLOG (without loss of generality) we assume that $h_{m,n+i} = 1$ → thus h_{mi} for $i = 1:n$ must ALL BE UNIQUE!!!



the 2 blocks are the same

Let h_{e_i} be the haplotype consistent with Q genotypes $\{g_{st}, \dots, g_{su}\}$. We want to show that $\{S_t, \dots, S_u\}$ is a clique in $\mathbb{Z}$.

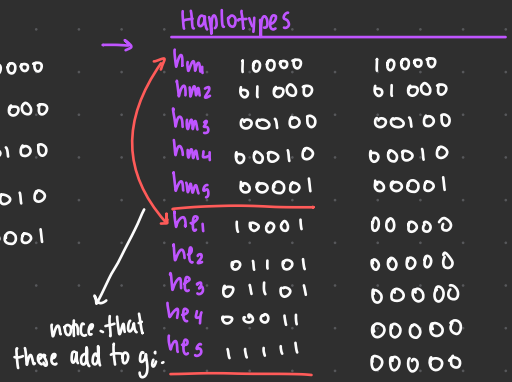
Recall that we can explain g_i with $g_i = h_{ei} + h_{mi}$ (unique)

For example, let $D = \{S_2, S_3, S_5\}$ for some h_{e_i}.

For $g_{i,i} = 2$ for $i = 1 \le i \le n$, which implies that $h_{ei} = h_{mi} = 1$. Thus, we must have $h_{e,k} = 1$ for $k = t, \dots, u$ to satisfy the connections (see example below).

$g_{i,j} = 1$ occurs on all diagonals for $1 < j < n$ since we need 2 on the diagonal.
Otherwise $g_{i,j} = 1$ means the nodes S_i, S_j are connected, 0 if not.

Vertex	Genotype
s_1	$g(s_1) = 20001\ 10000$
s_2	$g(s_2) = 02101\ 01000$
s_3	$g(s_3) = 01201\ 00100$
s_4	$g(s_4) = 00021\ 00010$
s_5	$g(s_5) = 11112\ 00001$



WRAPPING UP THE PROOF

Therefore the set of genotypes such that $g_{i,j} = 1$ for all $i \neq j$ corresponds to a set of vertices that are a CLIQUE!

s_2, s_3, s_5: 02101, 01201, 11112 → clique!

Thus, the cliques generated in this graph each have their own common haplotype! Thus the number of haplotypes is a factor of the number of edges.

★ So to minimize the number of haplotypes, we must minimize the number of cliques! ★

This completes the reduction! PPHP is NP-hard.

LECTURE 6

2/11/2025

★ SEE COURSE WEBSITE ★ FOR SLIDES ★

LECTURE 7

2/13/2025

We need to explore and understand population genetics and linkage disequilibrium.

- Mutations / Recombinations } cause genetic variation in a population.
- Random mating
- genotype → phenotype

LINKAGE DISEQUILIBRIUM / EQUILIBRIUM

Random Mating allows us to consider → recall that humans are diploid → ~2 alleles per gene

HARDY-WEINBERG EQUILBRIUM

- With random mating, the alleles of $\forall$ gene are combined at random according to HW-props
- Given genotypes: A_1A_1, A_1A_2, A_2A_2

If $\mathbb{P}(A_1) = q_1$, $\mathbb{P}(A_2) = q_2$, $1 = p_1 + A_1$ and then HWE: A_1A_1 is q_1^2, A_1A_2 is $2q_1q_2$, A_2A_2 is q_2^2.

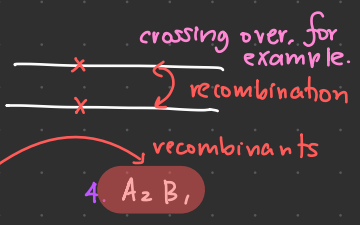
"Random Association" = the frequency of a gamete carrying any particular combination of alleles equals the products of the frequency of the alleles.

Definitions:

- Genes that are in random association are in linkage equilibrium. When they are not they are in linkage disequilibrium
- With random mating and simplifying assumptions → i.e no mutations, no migration, large population size → we converge to linkage equilibrium (+ no selection.)

The rate of approach to LE depends on the rate of recombination in genotypes heterozygous for both genes. There are 2 types of double heterozygotes

- A_1B_1 / A_2B_2
- A_1B_2 / A_2B_1



Some Genetics

Given A_1B_1 / A_2B_2, there are four possible genotypes

1. A_1B_1
2. A_2B_2
3. A_1B_2
4. A_2B_1

By mendelian Segregation, the frequency of genomic type 1 = the frequency of genetic type 2, " " 3 = " " type 4. because genomic recombination rates must imply type 3 = type 4, vice versa.

Definition: The recombination fraction is proportional to the number of recombinatorial gametes (type 3 + type 4) produced by a double heterozygote.

$0 \leq \hat{r} \leq 0.5$ → 0.5 means different genes or VERY far apart.

Genes where $\hat{r} < 0.5$ are linked.

Altogether: type III = IV = $\frac{\hat{r}}{2}$

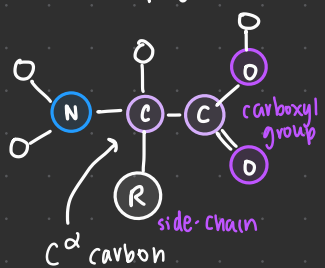
LECTURE 8: PROTEIN FOLDING

2/20/2025

Generally speaking, for a protein of length 50, there are an exponentially large number of ways to fold a protein. So how do we predict native structure?

PRIMER on PROTEINS

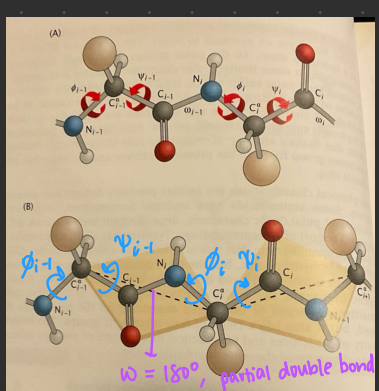
Proteins are polymers, sequences of individual peptides (of which there are 20).



proteins are either L-isomer or D-isomer, wher L is more common

L-isomer is when functional group points away if N-C-C were shown to be on the same plane.

Amino acids are marked by C-N peptide bonds → N-terminus (1) to c-terminus (n^{th}) these bonds form planes.



torsion angles ϕ and Ψ. rigidity in backbone planes between C^α_i's → this is due to the covalent bond nature within N-C-C

↓

"succession of planes that rotate about the planes between C^α indices. See (B) to the left.

Ramachandran Plots

The torsional angles vary → depending on angles we can classify as α-helices, β-strands due to the types of angles.

V → smaller L's tend to be α, ⌒, more obtuse reflect β-sheets.

Self Avoiding Walks / Contacts: → protein backbones are self-avoiding walks.

Models must not overlap while also considering the pairwise energies that result from interactions between peptides and even smaller, the component atoms.

Note: We want to minimize the energy of a system → this is when it is the most stable!

HP-model + Contact maps

A simple way to visualize this is the HP-model.

↳ we binarize all of the amino acids to be either H (hydrophobic), P (hydrophilic)

↳ we want to maximize the number of H-H contacts to minimize energy

example:  3 contacts!

vs  >> 1 contact

We can generalize this to using CONTACT maps!

We create graphical/matrix representations connecting and relating the individual amino acids in a particular protein sequence.

PIPELINE: Structure Similarity → Structure Alignment → Fold Recognition → Fold Alignment

Measuring Protein Similarity

How do we understand similarity for proteins of different sizes?

- RMSD → root-mean square distance
- Difference of distance matrices
- Contact map overlap
- Ad hoc scoring schemes

often done by choosing 100 aa regions for fixing length

Skipping Colin and Pranav's ★PERFECT POWERPOINTS★ ...

LECTURE 11 Markov Chain Monte Carlo Methods 3/04/25

3.1 Markov Chains + Markov Chain Monte Carlo (MCMC) Introduction

How do we compute the area of such an irregular shape?

→ 

Box the shape in a figure of area 1.

If we sample random pts, then by determining the ratio: $\frac{\text{pts in}}{\text{pts out}}$

We ≈ the area!!

Define: A **Markov Chain** has states $\{E_1, E_2, \ldots, E_S\}$ with timing $t = 1, 2, \ldots$

We also define a transition probability matrix **indexed by the states:**

→ sums to 1 along each row!

AXIOM 1: the **Markov Property** states that if at some point t the process is in state E_j, the probability that one timestep later we are at state E_i depends only on us currently being at E_j and not our previous trajectory to get to E_j.

Mathematically: $\mathbb{P}(X_k = E_k \mid X_{k-1} = E_{k-1}, \ldots, X_0 = E_0) = \mathbb{P}(X_k = E_k \mid X_{k-1} = E_{k-1})$

AXIOM 2: The transition probability is independent of t → **temporally homogenous.**

Definition: Where do we start our chain? $\lambda: \{\lambda_1, \ldots, \lambda_S\}$ where $\sum \lambda_i = 1$ is our initial state probability distribution.

Aperiodicity: There is no such state such that we revisit it at every t_0 (multiple) steps. If there are guaranteed cycles, we will not visit all the nodes or approximate the stationary distribution.

Define: the **stationary distribution** π is the probability we are in any state i. Suppose we have a transition matrix $P \rightarrow$ if $\pi_j = \sum_k \pi_k P_{kj}$ at any timestep we observe that π does not change. It is stationary.

Mathematically: $\pi = \pi P$ must hold!!

irreducibility dictates that every state can be visited from every other state in some number of steps.

☆ If we have **disconnected components:**

we will actually end up with **two distinct stationary distributions!** (think about why that might be :)

Theorem: Every finite and irreducible Markov Chain has a **unique stationary distribution**

If π is the SD of $M = (P, \lambda)$ then

$$\pi = \pi P \qquad \sum_{k=1}^{S} \pi_k = 1$$

We can use linear algebra to solve $\pi = \pi P$ to solve for π.

3.2 MCMC ALGORITHMS

Theorem Let $(X_0, \ldots, X_n)$ be irreducible and aperiodic, MC with state space S and trans. mat. P. Our estimate $\tilde{\pi}^{(n)}$:

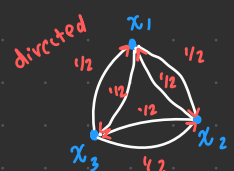
$$\tilde{\pi}^{(n)} \to \pi \quad \text{as} \quad n \to \infty.$$

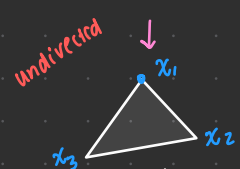
We essentially will observe the true underlying distribution as we sample a large number of samples.

Definition: Let MC, S state set, P trans. mat., A distribution π on S is reversible if $\forall ij \quad \pi_i P_{ij} = \pi_j P_{ji}$

LECTURE 12 MCMC Continued

RANDOM WALKS on GRAPHS





Assume = IP to transition to any arbitrary neighbor

These are our transition probabilities → so let's move it to a graph!

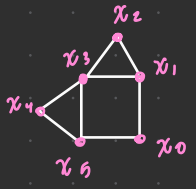
Graph $G = (V, E)$ → Vertices and Edges

A random walk on a graph is a markov chain with state set $V = \{v_1, \ldots, v_k\}$ and the following transition mechanism:

- If at v_i, it moves at time $t+1$ to one of neighbors w. equal IP.
- This is then a function of degree (v_i) = the number of neighbors.

$$P_{ij} = \begin{cases} 1/d_i, & \text{if } i,j \text{ are neighbors} \\ 0 & \text{otherwise.} \end{cases}$$

Extended Example



Then the stationary distribution $\pi = \left[\frac{d_0}{d}, \frac{d_1}{d}, \ldots, \frac{d_5}{d}\right]$ for $d = \sum d_i$

Intuitively, the nodes with the most neighbors will be visited the most number of times.

Markov Chain Monte Carlo

Given a probability distribution $\pi = [\pi_1, \pi_2, \ldots, \pi_K] \to \sum \pi_i = 1$ over a state space S.

How do we simulate a random object over the distribution π?

the "simple solution":

Let $X = \Psi(U)$ where U is uniform $[0,1]$ and the function Ψ:

$$\Psi(x) = \begin{cases} g_1 : & x : [0, \varphi(s_1)] \\ g_2 : & x : (\varphi(s_1), \varphi(s_1) + \varphi(s_2)] \\ \vdots & \\ s_K : & x : \left(\sum_{i=1}^{k-1} \varphi(s_i), \sum_i^K \varphi(s_i)\right] \end{cases}$$

→ a piecewise function that takes the "INVERSE CDF" method

"INVERSE CDF METHOD": an algorithm (learn more about this in APMA 1690!)

- Input: a given CDF $F(t)$ + a sample size n (the number of random variables)
- Output: A series of "iid" random variables.

1) Generate (pseudo) iid random variables $U_1, \ldots, U_n$ from Unif(0,1)
2) for $i = 1, 2, \ldots$ $X_i = \inf\{t \le \mathbb{R} \mid F(t) \ge U_i\}$

Proof of Thm: (prvs page)

Assume F^{-1} exists, then $G = F^{-1}$ (that is why we choose smallest t in Thm).

$$F(t) = \mathbb{P}\{w \in \Omega \mid X(w) \le t\} = \mathbb{P}\{w \in \Omega \mid F^{-1}(U(w)) \le t\} = \mathbb{P}\{w \in \Omega \mid U(w) \le F(t)\}$$

$= F(t)$ ✓ this proof is ONLY possible with the assumption $F^{-1} = G$.

this becomes impractical! Especially when K very large.

Metropolis Algorithm

We want to simulate a given π over a set of states. The graph that represents the states must:

- the graph must be connected and irreducible
- each vertex should not be the endpoint of too many edges.

We then define

IP of choosing given = IP to transition to an arbitrary edge

$$P_{ij} = \begin{cases} \frac{1}{d_i} \min\left\{\frac{\pi_j}{\pi_i}\frac{d_i}{d_j}, 1\right\} & \text{if } i \neq j \\ 0 & \text{if not neighbors} \\ 1 - \sum_{\text{neighbor}(I)} \frac{1}{d_i} \min\{\ldots\} & \text{if } i = j \end{cases}$$

debiasing term

Now we will prove that the