CANCER GENOMICS
Lecture 2: Probabilistic Methods for Mutation Detection

GENOME 541 Spring 2020
April 28, 2022

Gavin Ha, Ph.D.
Public Health Sciences Division
Human Biology Division

@GavinHa
gha@fredhutch.org
https://github.com/GavinHaLab
GavinHaLab.org

FRED HUTCH
CURES START HERE®

Outline

1. Primer on statistical modeling (cont'd)
    - Mixture models, inference and parameter estimation using the EM algorithm
2. Detecting Mutations in Cancer Genomes
    - Visualizing somatic vs germline SNVs
    - Sequencing read count data
3. Mixture Models for SNV Detection
    - SNV genotyping strategy
    - SNVMix probabilistic model and EM inference
    - Predicting somatic SNVs in cancer

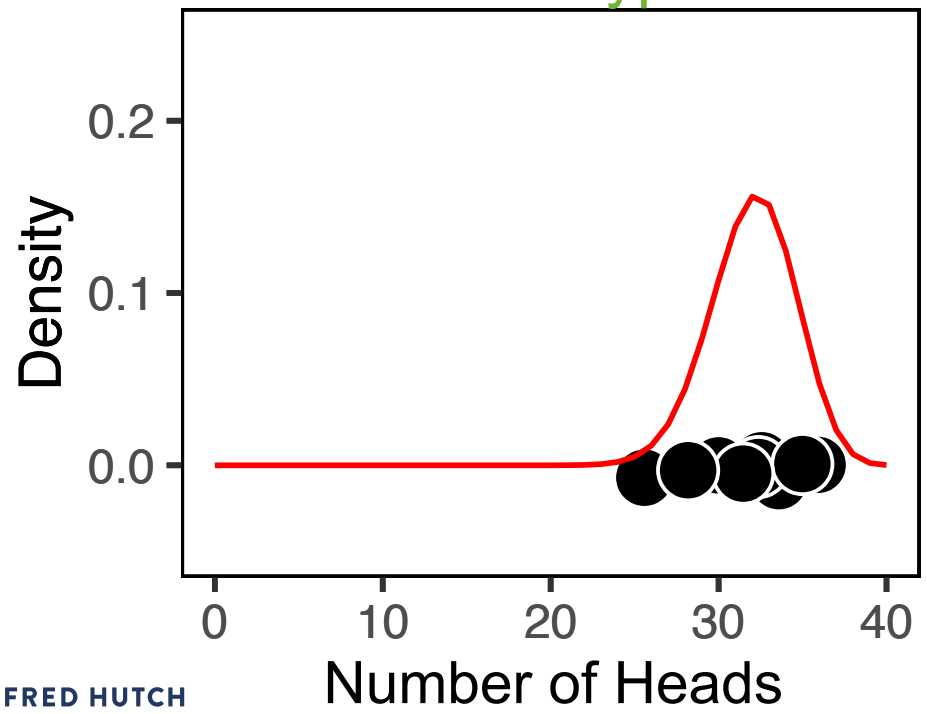
1. Primer on statistical modeling (cont'd)

- Probability
  - Unsupervised learning, probability rules & Bayes' theorem
  - Binomial distribution, Bayesian statistics
  - Beta-binomial model example
- **Mixture models, EM inference & parameter learning**
- References:
  - Murphy, K. (2012). Machine Learning: A Probabilistic Perspective. MIT Press. ISBN: 9780262018029
  - Bishop, C. M. (2006). Pattern Recognition and Machine Learning (Information Science and Statistics). Springer. ISBN: 0387310738

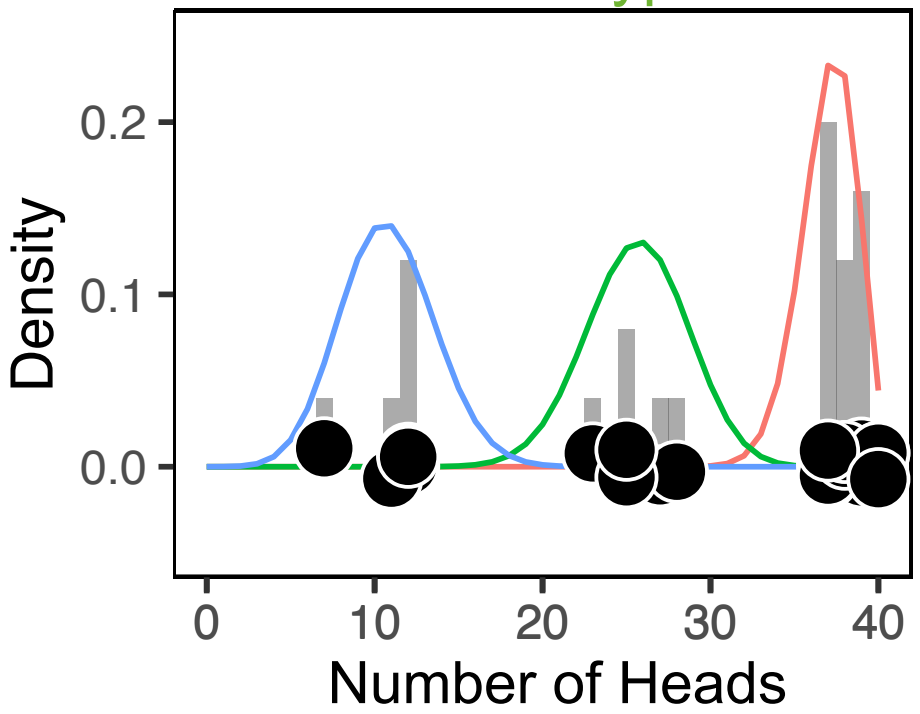
Mixture Model: Referee example with multiple coins

- Recall: There are T different referees who tossed the *same* coin $N = \{1, \ldots, N_T\}$ times and came up with counts of heads $\boldsymbol{x} = \{1, \ldots, x_T\}$.
- Now suppose there are **3 types of coins**: (1) probably fair, (2) unfairly favors heads, (3) unfairly favors tails denoted as $\{fair, heads, tails\}$.
- Each referee **draws one coin** (with replacement) from a hat containing these coin types mixed together.

1 coin type



3 coin types



Mixture Model: Referee example with multiple coins

- Recall: There are T different referees who tossed the *same* coin $N = \{1,\ldots,N_T\}$ times and came up with counts of heads $\boldsymbol{x} = \{1,\ldots,x_T\}$.
- Now suppose there are **3 types of coins**: (1) probably fair, (2) unfairly favors heads, (3) unfairly favors tails denoted as $\{fair, heads, tails\}$.
- Each referee **draws one coin from a hat** that contains a bunch of these coin types mixed together.

1. We don't know the proportion of each coin type in the hat.

2. We don't know which coin each referee drew from the hat.

3. We don't know the fairness (probability of heads) for each type of coin.

Referee	# of tosses (*N*)	# of heads (*x*)	Prop. of heads	Type of coin used?
Referee 1	40	25	0.63	?
Referee 2	42	35	0.83	?
Referee 3	39	27	0.69	?
Referee 4	x_T	N_T	x_T/N_T	?

Coin Type	Proportion in hat	Prob. of heads
"Fair"	?	?
"Heads"	?	?
"Tails	?	?

Mixture Model: Latent state model

1. What is the proportion of each coin type in the hat?

Find the probability for drawing a coin type.

- π_k is the probability of drawing coin type $k \in \{fair, heads, tails\}$
- $\boldsymbol{\pi} = (\pi_{fair}, \pi_{heads}, \pi_{tails})$ are the *mixture weights* where $\sum_{k=1}^{K} \pi_k = 1$

Coin Type	Proportion in hat	Prob. of heads
"Fair"	π_{fair}	?
"Heads"	π_{heads}	?
"Tails	π_{tails}	?

2. Which coin did each referee draw?

Define the latent variables.

- Let $Z_i = k$ be the type of coin that referee i draws
- Z_i is called a ***latent variable*** and follows a *Categorical* distribution with parameter $\boldsymbol{\pi}$

$$p(Z_i = k \mid \pi_{1:K}) = Cat(Z_i = k \mid \pi_{1:K})$$

$$= \begin{cases} \pi_{fair} & \text{if } k = fair \\ \pi_{heads} & \text{if } k = heads \\ \pi_{tails} & \text{if } k = tails \end{cases}$$

Referee	Type of coin used?
Referee 1	Z_1
Referee 2	Z_2
Referee 3	Z_3
Referee T	Z_T

- The proportions $\pi_{1:K}$ of the coin types follows a Dirichlet distribution (conjugate prior)

$$p(\pi_{1:K} \mid \delta_{1:K}) = Dirichlet(\pi_{1:K} \mid \delta_{1:K})$$

Chapter 9 in Bishop (2006). Pattern Recognition and Machine Learning

Section 11.2 in Murphy (2012). Machine Learning: A Probabilistic Perspective. MIT Press

Mixture Model: Likelihood as a mixture of binomials

3. What is the fairness (prob. of heads) for each type of coin?
Find the probability of heads for each coin type.

Coin Type	Proportion in hat	Prob. of heads
"Fair"	π_{fair}	μ_{fair}
"Heads"	π_{heads}	μ_{heads}
"Tails	π_{tails}	μ_{tails}

- Recall: for a single coin, $p(x_i \mid N_i, \mu) = Bin(x_i \mid N_i, \mu)$
- Define the likelihood for a ***3-component mixture of binomials*** with 3 parameters, $\mu_{fair}, \mu_{heads}, \mu_{tails}$, one for each type of coin

$$p(x_i \mid Z_i = k, N_i, \mu_{1:K}) = Bin(x_i \mid N_i, \mu_k)$$

Observed likelihood

$$p(x_i \mid N_i, \mu_{1:K}, \pi_{1:K}) = \sum_{k=1}^{K} \pi_k Bin(x_i \mid N_i, \mu_k)$$

Mixture model

- Beta prior distribution $p(\mu_k \mid \alpha_k, \beta_k) = Beta(\mu_k \mid \alpha_k, \beta_k)$

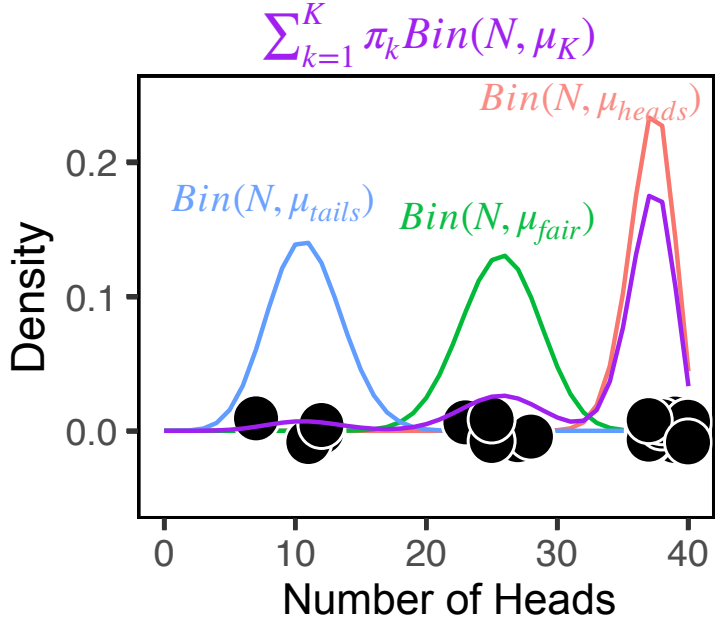
Log Likelihood Function of the Model

$$L(x_{1:T}, N_{1:T} \mid \mu_{1:K}, \pi_{1:K}) = \prod_{i=1}^{T} \sum_{k=1}^{K} \pi_k Bin(x_i \mid N_i, \mu_k)$$

Likelihood function

$$\ell = \sum_{i=1}^{T} \log \left(\sum_{k=1}^{K} \pi_k Bin(x_i \mid N_i, \mu_k) \right)$$

Log likelihood



Chapter 9 in Bishop (2006). Pattern Recognition and Machine Learning. Springer

Section 3.3, 3.4, 11.2 in Murphy (2012). Machine Learning: A Probabilistic Perspective. MIT Press

Mixture Model: Inference & parameter estimation using EM (1)

Expectation-Maximization: Inference and parameter training

Initialize parameters: $\pi_{1:K}$ and $\mu_{1:K}$

E-Step: compute "responsibilities" (inference)

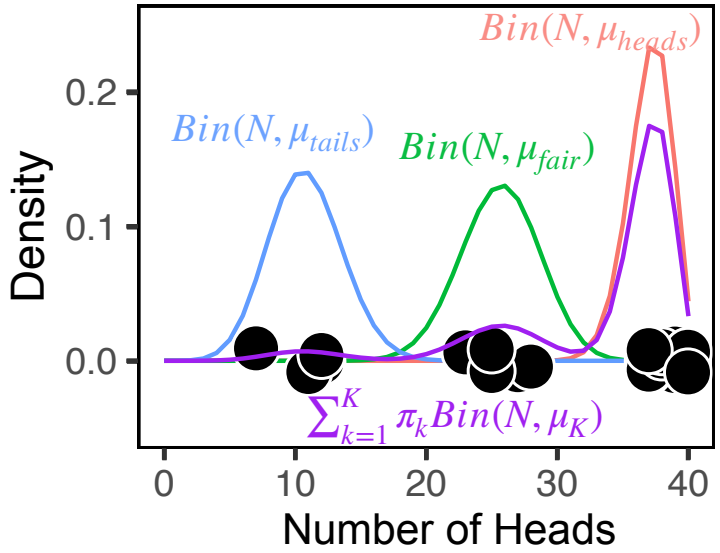
1. Which coin did each referee draw?
   (Posterior of the latent states $\gamma(Z_{1:T})$)
   - Soft-clustering: Referee i has a probability for using each of the coins.
   - Responsibilities: "coin that is responsible for generating observation x_i"

	Responsibilities		
Referee	Fair Coin	Heads Coin	Tails Coin
1	$\gamma(Z_1 = F)$	$\gamma(Z_1 = H)$	$\gamma(Z_1 = T)$
2	$\gamma(Z_2 = F)$	$\gamma(Z_2 = H)$	$\gamma(Z_2 = T)$
3	$\gamma(Z_3 = F)$	$\gamma(Z_3 = H)$	$\gamma(Z_3 = T)$
T	$\gamma(Z_T = F)$	$\gamma(Z_T = H)$	$\gamma(Z_T = T)$

M-Step: Update parameters (learning)

2. What is the proportion of each coin type in the hat? $\pi_{1:K}$
3. What is the fairness (prob. of heads) for each coin type? $\mu_{1:K}$

Iterate between E-Step and M-Step, check when log-likelihood ℓ (plus log priors) stops increasing.



Chapter 9 in Bishop (2006). Pattern Recognition and Machine Learning. Springer

Section 3.3, 3.4, 11.2 in Murphy (2012). Machine Learning: A Probabilistic Perspective. MIT Press

Mixture Model: Inference & parameter estimation using EM (2)

E-Step: compute responsibilities (inference)

1. What is the probability for a referee to draw each coin type? (Posterior of the latent states $Z_{1:T}$)

- Find the responsibilities given the current parameters

$$p(Z_i = k \mid x_i, N_i, \pi_{1:K}, \mu_{1:K}) = \frac{p(x_i \mid Z_i = k)p(Z_i = k)}{p(x_i)}$$
$$= \frac{\pi_k Bin(x_i \mid N_i, \mu_k)}{\sum_{k'=1}^{K} \pi_{k'} Bin(x_i \mid N_i, \mu_{k'})}$$
$$= \gamma(Z_i = k)$$

Bayes' Rule
Posterior distribution of the latent variables

Responsibilities
Matrix $T \times K$

- Responsibilities = "coin that is responsible for generating observation x_i"
- Soft-clustering: Referee i has a probability for using each of the coins.
- $\gamma(Z_{1:T})$ is a matrix of probabilities with dimensions $T \times K$

Mixture Model: Inference & parameter estimation using EM (3)

M-Step: Update parameters (learning)

2. What is the proportion of each coin type in the hat?

$$\hat{\pi}_k = \frac{\sum_{i=1}^{T} \gamma(Z_i = k) + \delta(k) - 1}{\sum_{j=1}^{K} \sum_{i=1}^{T} \{\gamma(Z_i = j) + \delta(j) - 1\}}$$

MAP for π

3. What is the fairness (prob. of heads) for each coin type?

$$\hat{\mu}_k = \frac{\sum_{i=1}^{T} \gamma(Z_i = k) x_i + \alpha_k - 1}{\sum_{i=1}^{T} \gamma(Z_i = k) N_i + \alpha_k + \beta_k - 2}$$

MAP for μ

Evaluate the log likelihood and log posterior: use updated parameters

Log posterior

$$\log \mathbb{P} = \sum_{i=1}^{T} \log \left(\sum_{k=1}^{K} \hat{\pi}_k Bin(x_i | N_i, \hat{\mu}_k) \right) + \log Dir(\hat{\boldsymbol{\pi}} | \boldsymbol{\delta}) + \sum_{k=1}^{K} \log Beta(\hat{\mu}_k | \alpha_k, \beta_k)$$

Log likelihood **Log priors**

Iterate between E-Step and M-Step:

- Stop EM when new $\log \mathbb{P}$ changes less than ϵ compared to previous EM iteration.

Chapter 9 in Bishop (2006). Pattern Recognition and Machine Learning. Springer

Section 3.3, 3.4, 11.2 in Murphy (2012). Machine Learning: A Probabilistic Perspective. MIT Press

Bayesian statistics: Posterior for Beta-Binomial Model (2)

Beta-Binomial Model: Posterior distribution

- To estimate the model parameter μ in a Bayesian framework, we compute the ***posterior***, $p(\mu \mid \boldsymbol{x})$

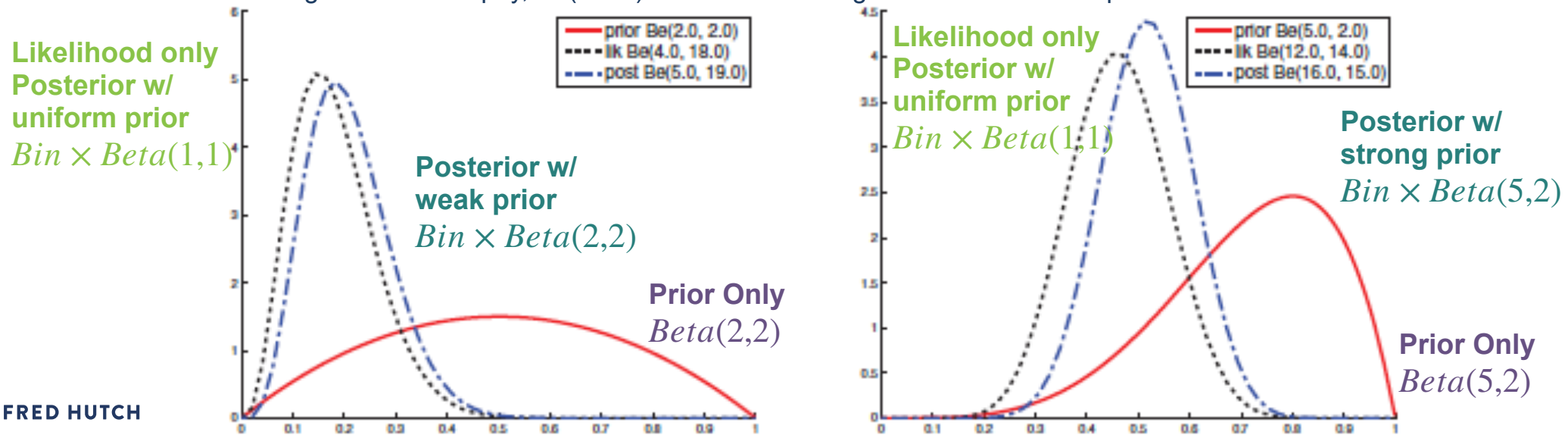
$$p(\mu \mid x_i) \propto Bin(x_i \mid N_i, \mu) \times Beta(\mu \mid \alpha, \beta)$$

- Beta is a ***conjugate prior*** for the binomial; *the product of binomial and Beta has the form of a Beta*

$$p(\mu \mid x_i) \propto Bin(x_i \mid N_i, \mu) \times Beta(\mu \mid \alpha, \beta) = Beta(\mu \mid x_i + \alpha, N_i - x_i + \beta)$$

Likelihood **Prior** **Posterior**

Figure 3.6 in Murphy, K. (2012). Machine Learning: A Probabilistic Perspective. MIT Press



Algorithm 1 Binomial Mixture Model Inference and Learning using EM

Inputs:
  Data: $x_{1:T}, N_{1:T}$
  Initial parameters: $\pi_{1:K}^{(0)}$, $\mu_{1:K}^{(0)}$,
  Hyperparameters: $\delta_{1:K}$, $\alpha_{1:K}$, $\beta_{1:K}$
Initialize:
  $\pi_{1:K} \leftarrow \pi_{1:K}^{(0)}$, $\mu_{1:K} \leftarrow \mu_{1:K}^{(0)}$
  `logP` $\leftarrow -Inf$
Compute the observed likelihood using initial parameters:
  `lik` $\leftarrow$ `compute.binom.lik()`
while `converged = false` **do**
  E-Step: Compute responsibilities:
    $\gamma(Z_{1:T}) \leftarrow$ `compute.responsibilities()`
  M-Step: Update parameters:
    $\hat{\pi}_{1:K} \leftarrow$ `update.pi()`
    $\hat{\mu}_{1:K} \leftarrow$ `update.mu()`
  Assign updated parameters:
    $\pi_{1:K} \leftarrow \hat{\pi}_{1:K}$, $\mu_{1:K} \leftarrow \hat{\mu}_{1:K}$
  Re-compute the observed likelihood using updated parameters:
    `obs.lik` $\leftarrow$ `compute.binom.lik()`
  Compute the log-likelihood:
    `loglik` $\leftarrow$ `compute.loglik()`
  Compute log Posterior:
    `logP[curr.iter]` $\leftarrow$ `compute.log.posterior()`
  if (`logP[curr.iter] - logP[prev.iter] <` ϵ) **then**
    `converged = true`
  end if
  `logP[prev.iter]` $\leftarrow$ `logP[curr.iter]`
end while
return Responsibilites $\gamma(Z_{1:T})$, Converged parameters $\hat{\pi}_{1:K}$, $\hat{\mu}_{1:K}$

Mixture Model: Inference & parameter estimation using EM (extra slide 1)

Incomplete data log likelihood

$$L(x_{1:T}, N_{1:T} | \mu_{1:K}, \pi_{1:K}) = \prod_{i=1}^{T} \sum_{k=1}^{K} \pi_k Bin(x_i | N_i, \mu_k)$$

- The incomplete data log likelihood (plus the priors) is used to monitor EM convergence

Expected complete data log likelihood

Complete data likelihood

$$L(\mu_{1:K}, \pi_{1:K} | x_{1:T}, Z_{1:T}, N_{1:T}) = \prod_{i=1}^{T} \prod_{k=1}^{K} \pi_k Bin(x_i | N_i, \mu_k)^{\mathbb{I}(Z_i=k)}$$

Complete data log likelihood

$$\ell(\mu_{1:K}, \pi_{1:K} | x_{1:T}, Z_{1:T}, N_{1:T}) = \sum_{i=1}^{T} \sum_{k=1}^{K} \mathbb{I}(Z_i = k) \left\{ \log \pi_k + \log Bin(x_i | N_i, \mu_k) \right\}$$

Expected complete data log likelihood

$$Q = \mathbb{E}\left[\ell(\mu_{1:K}, \pi_{1:K} | x_{1:T}, Z_{1:T}, N_{1:T})\right] = \sum_{i=1}^{T} \sum_{k=1}^{K} \mathbb{E}\left[\mathbb{I}(Z_i = k)\right] \left\{ \log \pi_k + \log Bin(x_i | N_i, \mu_k) \right\}$$

$$= \sum_{i=1}^{T} \sum_{k=1}^{K} \gamma(Z_i = k) \left\{ \log \pi_k + \log Bin(x_i | N_i, \mu_k) \right\}$$

- The expected complete data log likelihood is in the M-Step for updating parameters.

Mixture Model: Inference & parameter estimation using EM (extra slide 2)

M-Step: Update the parameters given the responsibilities

$$\mathbb{P}(\pi_{1:K}, \mu_{1:K}) = Dir(\boldsymbol{\pi} \mid \boldsymbol{\delta}) \prod_{k=1}^{K} Beta(\mu_k \mid \alpha, \beta)$$ **Priors**

$$\mathcal{O} = Q + \log \mathbb{P}(\pi_{1:K}, \mu_{1:K})$$ **Complete data log likelihood + log priors**

- The object function $\mathcal{O}$ is used to obtain the update equations for $\pi_{1:K}$ and $\mu_{1:K}$

$$\frac{\partial \mathcal{O}}{\partial \mu_k} = 0, \text{ find } \hat{\mu}_k \text{ and } \frac{\partial \mathcal{O}}{\partial \pi_k} = 0, \text{ find } \hat{\pi}_k$$

EM Convergence: after each iteration, monitor the log posterior

$$\ell = \sum_{i=1}^{T} \log \left(\sum_{k=1}^{K} \pi_k Bin(x_i \mid \mu_k, N_i) \right)$$ **Incomplete Data Log likelihood**

$$\log \mathbb{P}(\pi_{1:K}, \mu_{1:K} \mid x_{1:T}) = \ell + \log \mathbb{P}(\pi_{1:K}, \mu_{1:K})$$ **Log posterior**

- If the log posterior, $\log \mathbb{P}(\pi_{1:K}, \mu_{1:K} \mid x_{1:T})$, stops increasing by ϵ, then EM is converged.
- If not using a Bayesian framework, then use the log likelihood, ℓ, to monitor convergence.

Mixture Models: Online Tutorial and Resource

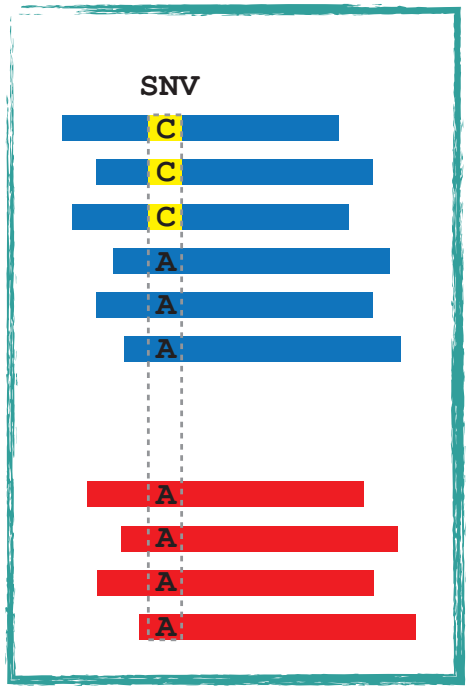
fiveMinuteStats (https://stephens999.github.io/fiveMinuteStats/)

by **Dr. Matthew Stephens**, Professor in Statistics & Human Genetics at University of Chicago

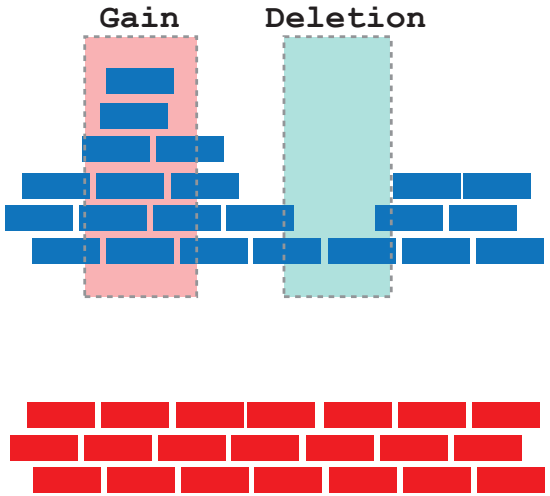
1. Introduction to mixture models with probabilistic derivations and R code
    - Examples with Bernoulli and Gaussian models
    - https://stephens999.github.io/fiveMinuteStats/intro_to_mixture_models.html
2. Introduction to EM with Gaussian Mixture Model example and R code
    - https://stephens999.github.io/fiveMinuteStats/intro_to_em.html

2. Detecting Mutations in Cancer Genomes

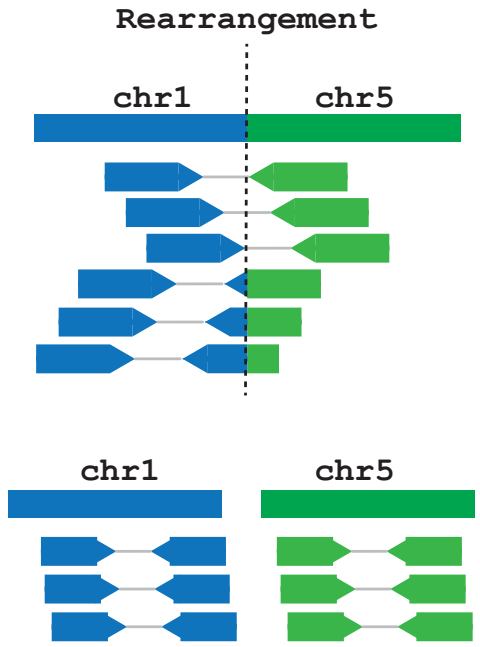
Mutations (SNV, INDEL)



Copy Number Alterations



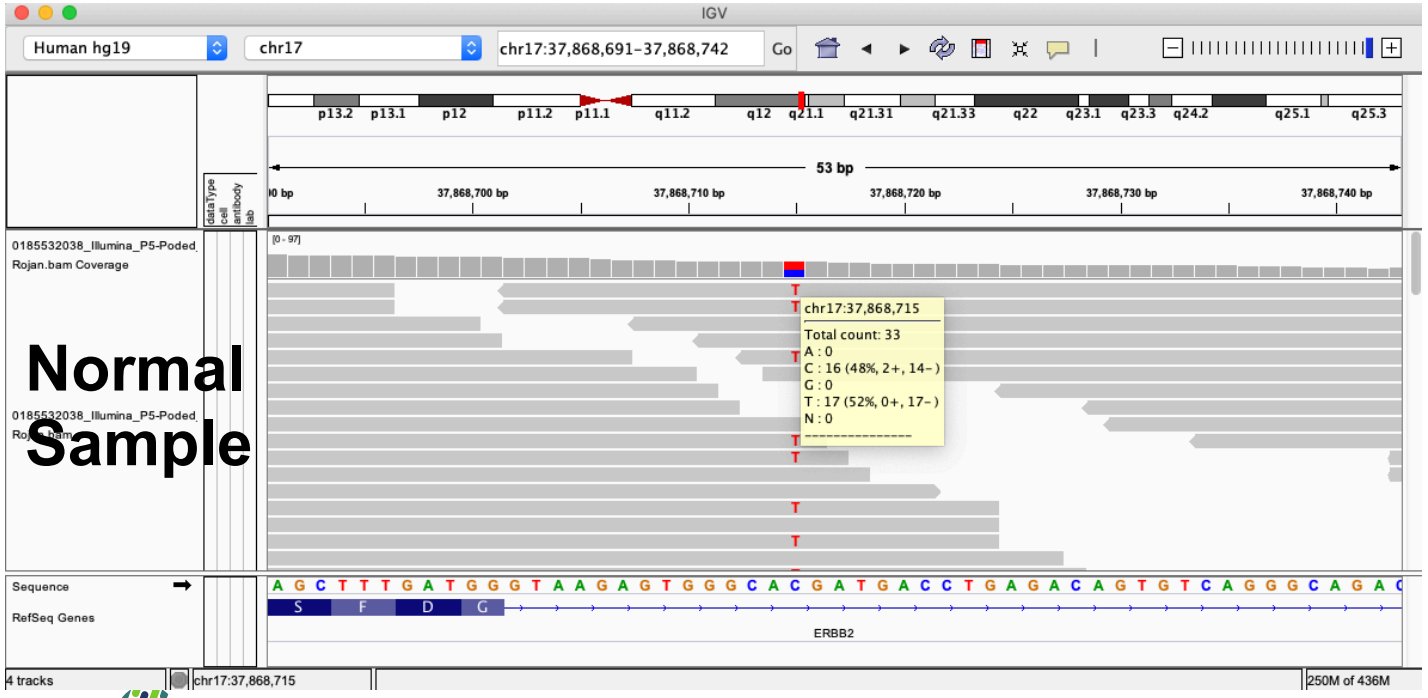
Structural Variants



Visual inspection using IGV: Germline SNVs

Integrative Genomics Viewer (https://software.broadinstitute.org/software/igv)

- ~1.5 to 2 million **SNPs** per individual
- Identify SNPs from normal peripheral blood mononuclear cells (PBMC)



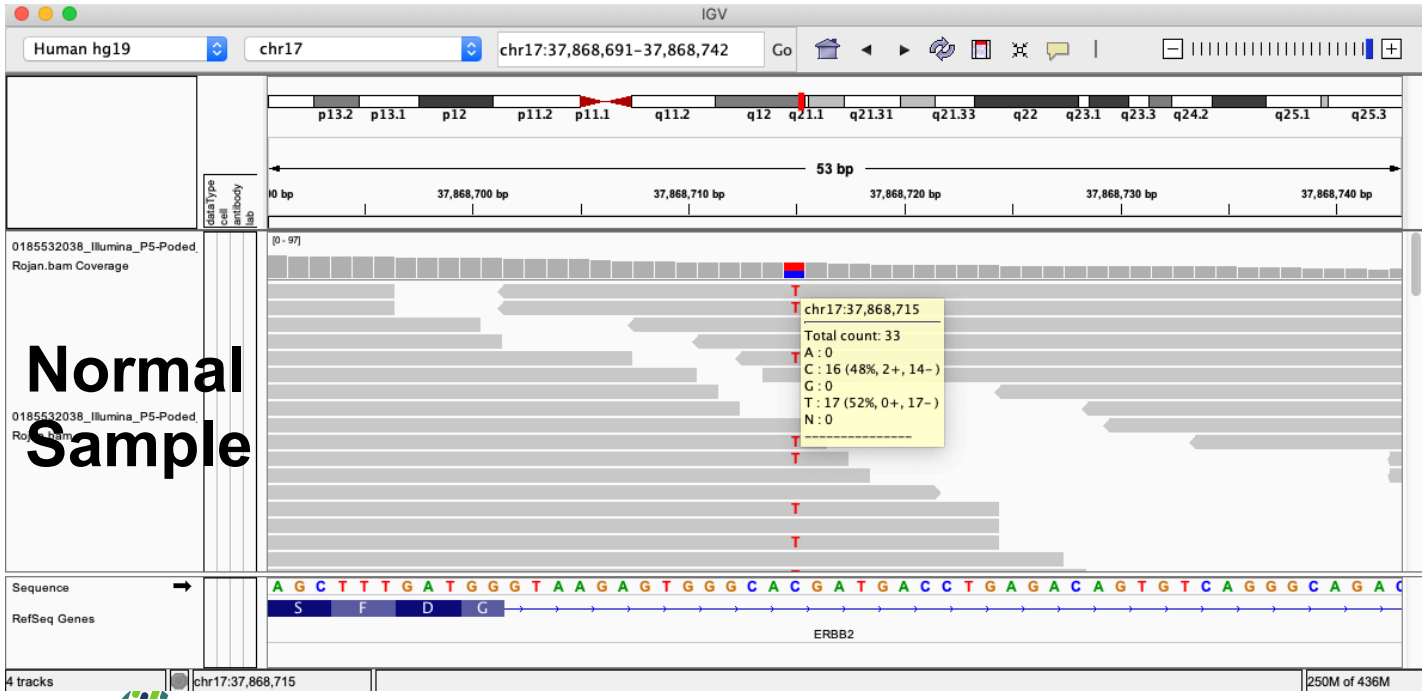
Heterozygous SNP with 17 reads containing the variant and having depth 33 reads

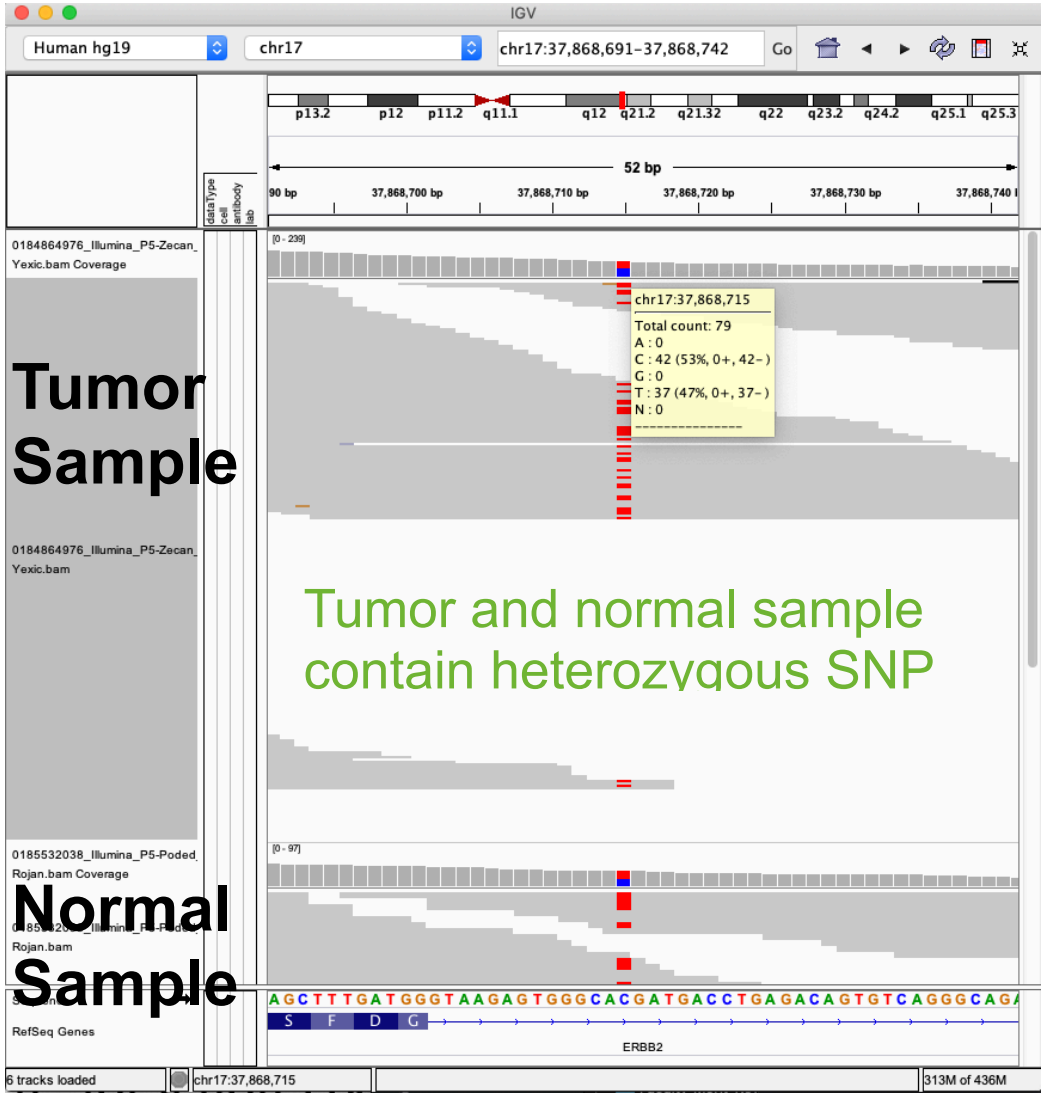
17/33 (48%) variant allele fraction (VAF)

Visual inspection using IGV: Germline SNVs

Integrative Genomics Viewer (https://software.broadinstitute.org/software/igv)

- ~1.5 to 2 million **SNPs** per individual
- Identify SNPs from normal peripheral blood mononuclear cells (PBMC)

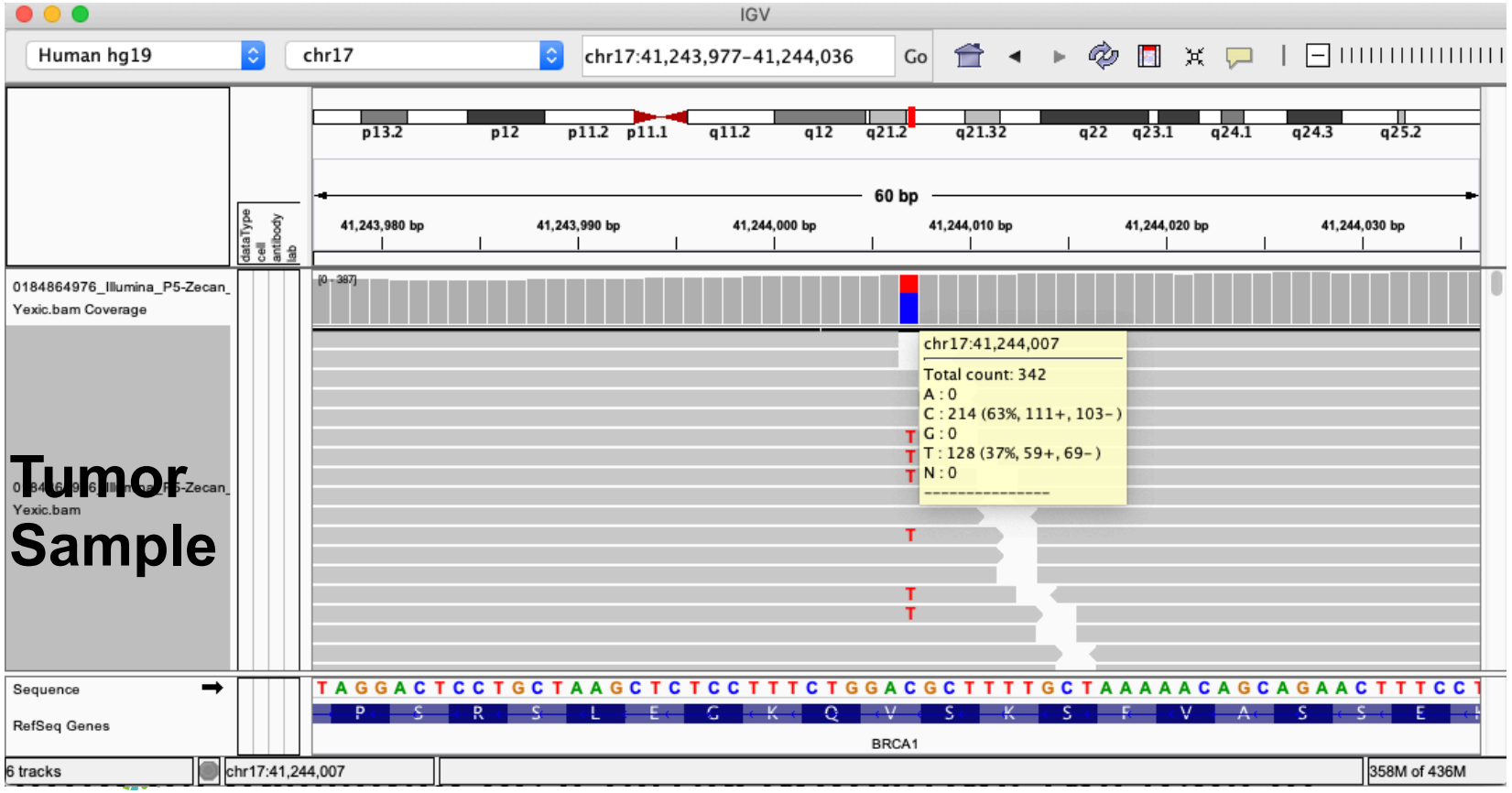




FRED HUTCH

Visual inspection using IGV: Somatic SNVs

- Somatic **SNV** requires comparing case (tumor) with control (PBMC)
- On the order of 10 to 10^4 number of mutations

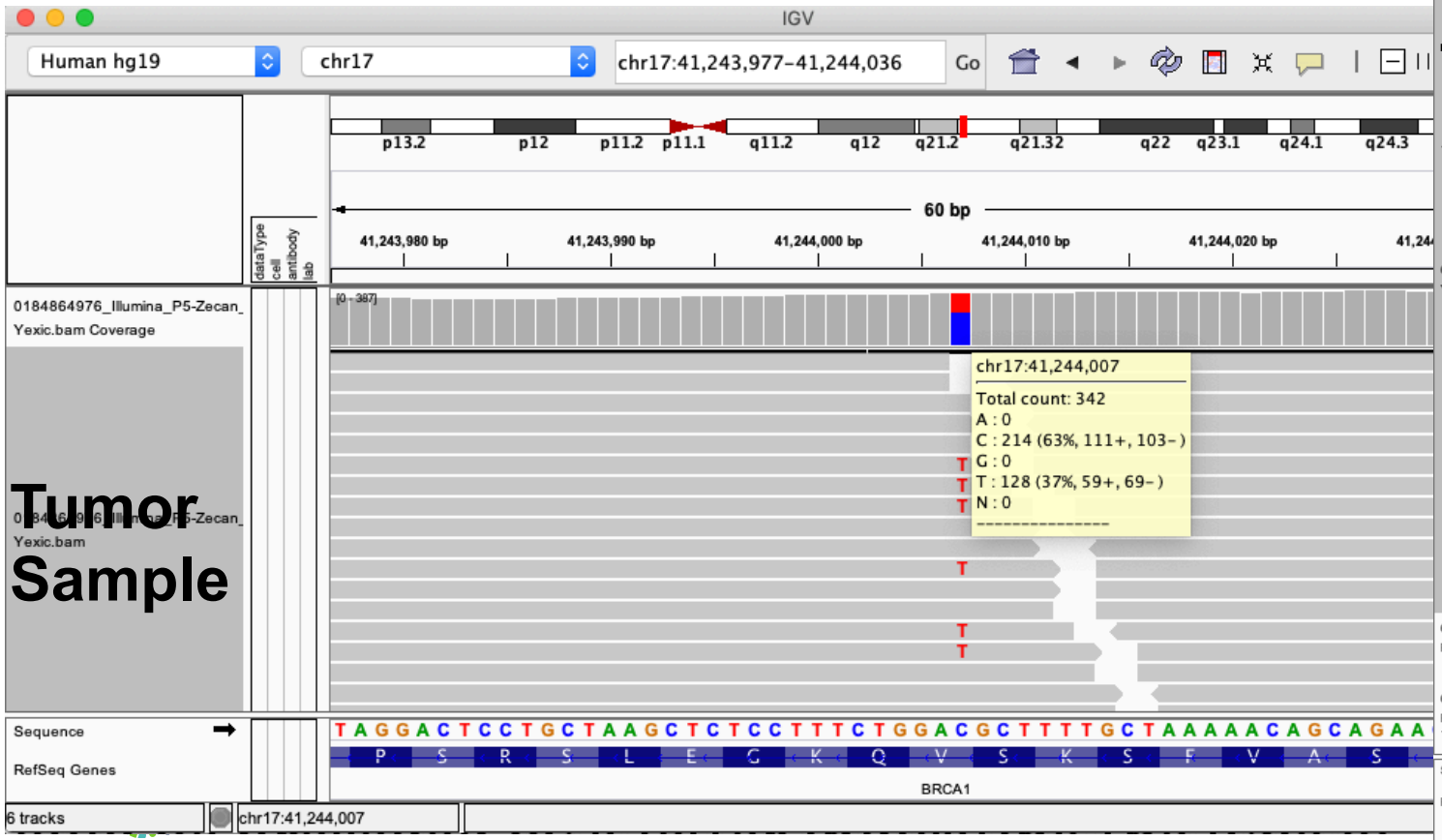


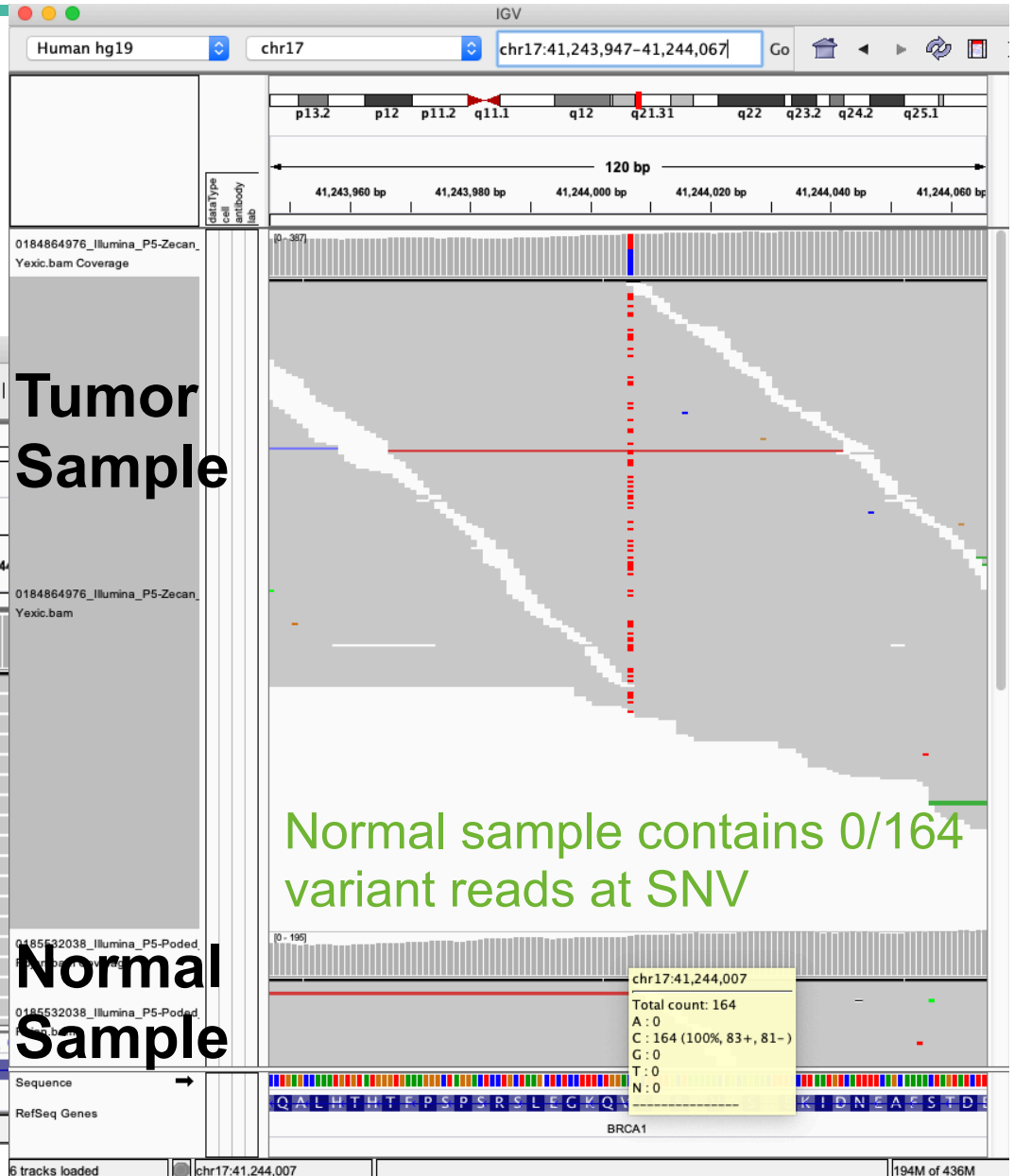
Potential SNV with 128/342 (37%) VAF

p.V1181I

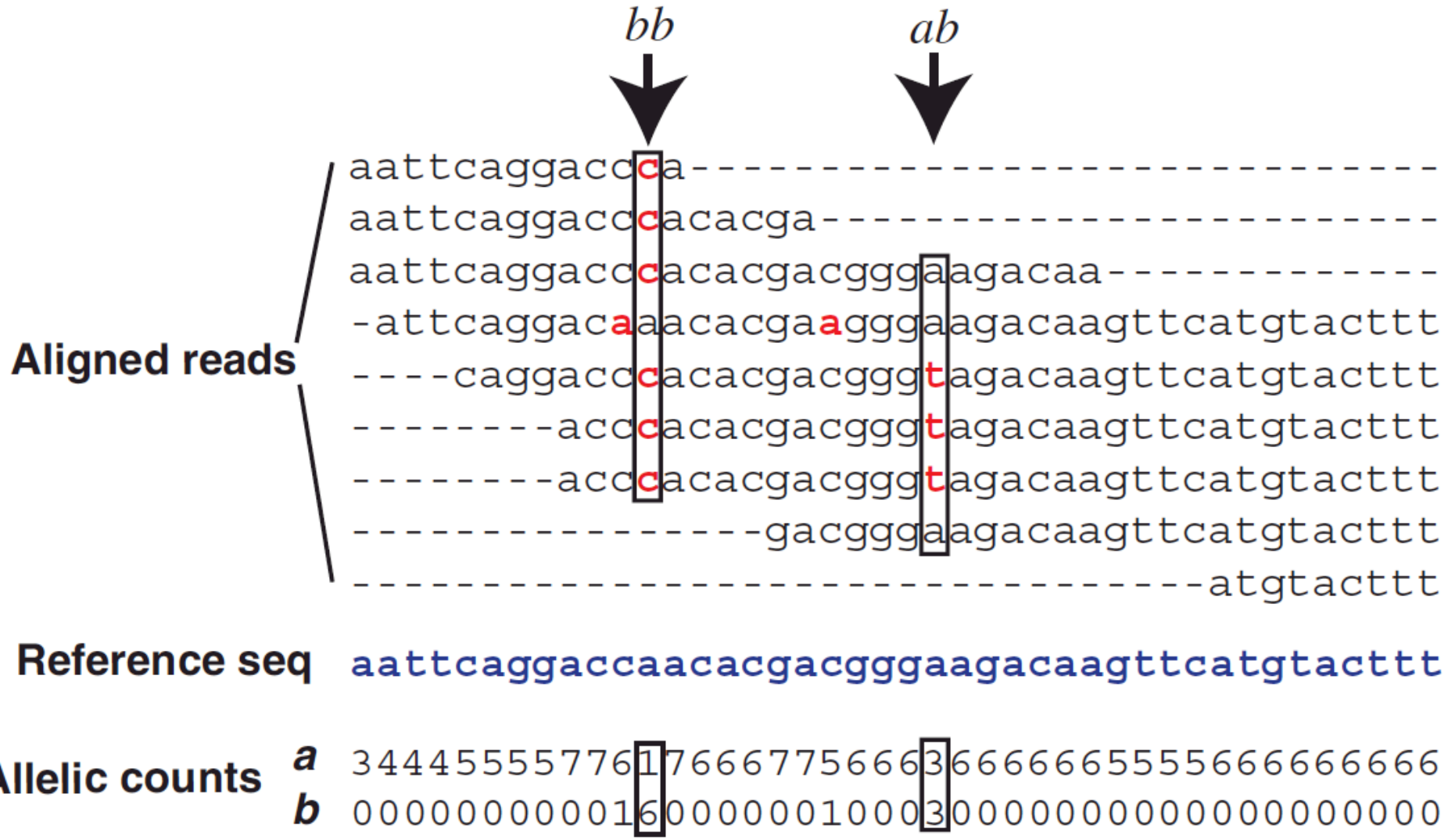
Visual inspection using IGV: Somatic SNVs

- Somatic **SNV** requires comparing case (tumor) with control (PBMC)
- On the order of 10 to 10^4 number of mutations





Single Nucleotide Variant (SNV) Calling: Single Sample



FRED HUTCH

Goya et al. *Bioinformatics* **26**:703-36 (2010)

SNV Variant Allele Fraction and Genotypes

Variant Allele Fraction (VAF) Analysis

Genotypes: AA, AB, BB

Genotype	AA	AB	BB
Allelic Fraction	~1.0	~0.5	~0

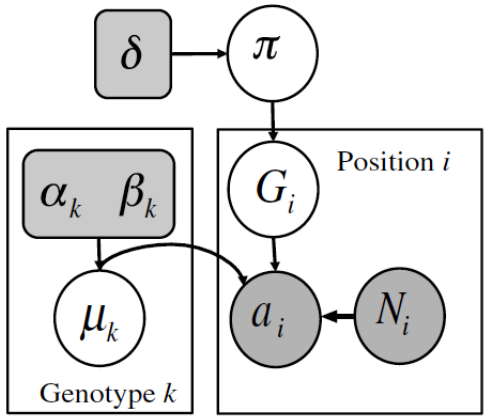
- **Allelic Fraction** is defined as the fraction of reference reads, $\frac{A}{N}$, where depth $N = A + B$
- Values in the table are the *expected* proportions of *reference reads* for each genotype
- Why might the observed allelic fractions be different than the expected values?

3. Mixture Model for SNV Detection

- SNVMix probabilistic model and EM inference
- Predicting somatic SNVs in cancer

References:

- Goya et al. **SNVMix**: predicting single nucleotide variants from next-generation sequencing of tumors. *Bioinformatics* **26**:730-36 (2010)
- Roth et al. **JointSNVMix**: a probabilistic model for accurate detection of somatic mutations in normal/tumour paired next-generation sequencing data. *Bioinformatics* **28**:907-13 (2012)



SNVMix1 model

Mapping the Referee Example to Mutation Calling

Referee Coin Toss Example

Data

Referees $1,\ldots,T$

For each Referee i

- Coin Tosses: N_i
- Count of heads: x_i
- Count of tails: $N_i - x_i$

Parameters

Probability to draw coins: π_{fair}, π_{heads}, π_{tails}

Probability of heads for 3 types of coins

$$\mu_{fair}, \mu_{heads}, \mu_{tails}$$

Responsibilities

Probability that Referee i used coin k: $\gamma(Z_i = k)$

Mutation Calling from Sequencing Data

Data

Genomic loci $1,\ldots,T$

For each locus i

- Depth (total reads): N_i
- Count of reference base: x_i
- Count of variant base: $N_i - x_i$

Parameters

Probability of genotypes: π_{AA}, π_{AB}, π_{BB}

Probability of reference base for 3 genotypes:

$$\mu_{AA}, \mu_{AB}, \mu_{BB}$$

Responsibilities

Probability that locus i has genotype k: $\gamma(Z_i = k)$

SNVMix: Probabilistic Model

Sequence Data

There are T different genomic loci with read depths $N = \{1,\ldots,N_T\}$ and reference base counts $\boldsymbol{x} = \{1,\ldots,x_T\}$.
There are $K = 3$ different possible genotypes $AA,\ AB,\ BB$

Mixture Model Setup

1. The **probabilities for the genotypes** are $\pi_{AA},\ \pi_{AB},\ \pi_{BB}$
2. Thus, a specific genotype $k \in AA,\ AB,\ BB$ can be assigned to the **latent state** Z_i at locus i with these probabilities

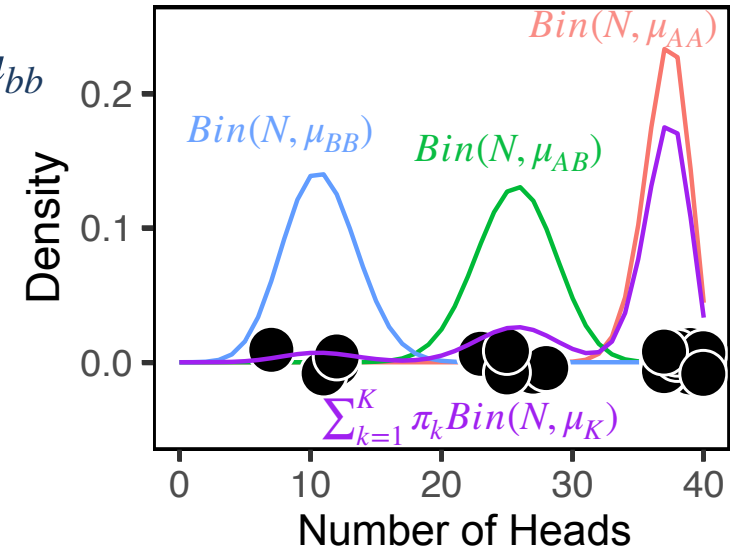
$$p(Z_i = k \mid \pi_{1:K}) = \begin{cases} \pi_{AA} \text{ if } k = AA \\ \pi_{AB} \text{ if } k = AB \\ \pi_{BB} \text{ if } k = BB \end{cases}$$

3. The **probability of observing a reference base** for the genotypes are $\mu_{aa},\ \mu_{ab},\ \mu_{bb}$
4. The likelihood is a ***3-component mixture of binomials***

$$p(x_i \mid N_i, \mu_{1:K}, \pi_{1:K}) = \sum_{k=1}^{K} \pi_k Bin(x_i \mid N_i, \mu_k)$$

5. The **priors** for genotype $k \in \{aa,\ ab,\ bb\}$ in the model are

$$p(\pi_{1:K} \mid \delta_{1:K}) = Dirichlet(\pi_{1:K} \mid \delta_{1:K})$$
$$p(\mu_k \mid \alpha_k, \beta_k) = Beta(\mu_k \mid \alpha_k, \beta_k)$$



Goya et al. *Bioinformatics* **26**:703-36 (2010)

SNVMix: Inference & parameter estimation using EM (revisited)

E-Step: compute responsibilities

1. What is the probability of locus i having genotype k?

$$\gamma(Z_i = k) = \frac{\pi_k Bin(x_i | N_i, \mu_k)}{\sum_{j=1}^{K} \pi_j Bin(x_i | N_i, \mu_j)}$$

Responsibilities
Matrix $T \times K$

M-Step: update parameters

2. What is the probability of genotype k?

$$\hat{\pi}_k = \frac{\sum_{i=1}^{T} \gamma(Z_i = k) + \delta(k) - 1}{\sum_{j=1}^{K} \left\{ \sum_{i=1}^{T} \gamma(Z_i = j) + \delta(j) - 1 \right\}}$$

MAP for π

3. What is the probability of observing a reference base for genotype k?

$$\hat{\mu}_k = \frac{\sum_{i=1}^{T} \gamma(Z_i = k) x_i + \alpha_k - 1}{\sum_{i=1}^{T} \gamma(Z_i = k) N_i + \alpha_k + \beta_k - 2}$$

MAP for μ

Evaluate the log likelihood and log posterior: use updated parameters

$$\log \mathbb{P} = \sum_{i=1}^{T} \log \left(\sum_{k=1}^{K} \hat{\pi}_k Bin(x_i | \hat{\mu}_k, N_i) \right) + \log Dir(\hat{\pi}_k | \delta_k) + \sum_{k=1}^{K} \log Beta(\hat{\mu}_k | \alpha_k, \beta_k)$$

Log posterior

Iterate between E-Step and M-Step: stop when $\log \mathbb{P}$ changes less than ϵ compared to previous EM iteration.

Chapter 9 in Bishop (2006). Pattern Recognition and Machine Learning. Springer

Section 3.3, 3.4, 11.2 in Murphy (2012). Machine Learning: A Probabilistic Perspective. MIT Press

SNVMix: Calling somatic SNVs from genotype inference

Genotypes: AA, AB, BB

Responsibilities			
Locus	**AA**	**AB**	**BB**
1	$\gamma(Z_1 = AA)$	$\gamma(Z_1 = AB)$	$\gamma(Z_1 = BB)$
2	$\gamma(Z_2 = AA)$	$\gamma(Z_2 = AB)$	$\gamma(Z_2 = BB)$
3	$\gamma(Z_3 = AA)$	$\gamma(Z_3 = AB)$	$\gamma(Z_3 = BB)$
T	$\gamma(Z_T = AA)$	$\gamma(Z_T = AB)$	$\gamma(Z_T = BB)$

- To call a variant for each locus i, we can apply a threshold on the responsibilities $\gamma(Z_i)$
- We can sum $\gamma(Z_i = AB)$ and $\gamma(Z_i = BB)$ to get the overall probability (either genotype AB or BB) that locus i is a variant containing the non-reference allele (B)
- Additional steps required for filtering and determining if variant is somatic vs germline
  - Minimum 3 variant reads ($N_i - x_i$) is typically required
  - Account for mapping and base qualities of sequenced reads (i.e. SNVMix2)
  - Compare locus i in tumor sample to (1) matched normal sample, (2) germline databases

SNV Genotyping Callers

Variant Allele Fraction Analysis

- Single sample

Genotypes: AA, AB, BB

Homozygous Reference (not SNV)

Heterozygous Variant (**Het SNV**)

Homozygous Variant (**Homd SNV**)

- Joint tumor-normal

Joint Genotypes:

$g_N \backslash g_T$	AA	AB	BB
AA	0.01	0.95	0.00
AB	0.00	0.04	0.00
BB	0.00	0.00	0.00

- Cohort level or panel: Machine Learning (supervised)

Variant caller	Type of variant	Single-sample mode	Type of core algorithm
BAYSIC [48]	SNV	No	Machine learning (ensemble caller)
CaVEMan [34]	SNV	No	Joint genotype analysis
deepSNV [38]	SNV	No	Allele frequency analysis
EBCall [37]	SNV, indel	No	Allele frequency analysis
FaSD-somatic [31]	SNV	Yes	Joint genotype analysis
FreeBayes [44]	SNV, indel	Yes	Haplotype analysis
HapMuC [42]	SNV, indel	Yes	Haplotype analysis
JointSNVMix2 [30]	SNV	No	Joint genotype analysis
LocHap [43]	SNV, indel	No	Haplotype analysis
LoFreq [36]	SNV, indel	Yes	Allele frequency analysis
LoLoPicker [39]	SNV	No	Allele frequency analysis
MutationSeq [45]	SNV	No	Machine learning
MuSE [40]	SNV	No	Markov chain model
MuTect [35]	SNV	Yes	Allele frequency analysis
SAMtools [8]	SNV, indel	Yes	Joint genotype analysis
Platypus [41]	SNV, indel, SV	Yes	Haplotype analysis
qSNP [24]	SNV	No	Heuristic threshold
RADIA [26]	SNV	No	Heuristic threshold
Seurat [33]	SNV, indel, SV	No	Joint genotype analysis
Shimmer [25]	SNV, indel	No	Heuristic threshold
SNooPer [47]	SNV, indel	Yes	Machine learning
SNVSniffer [32]	SNV, indel	Yes	Joint genotype analysis
SOAPsnv [27]	SNV	No	Heuristic threshold
SomaticSeq [46]	SNV	No	Machine learning (ensemble caller)
SomaticSniper [28]	SNV	No	Joint genotype analysis
Strelka [17]	SNV, indel	No	Allele frequency analysis
TVC [97]	SNV, indel, SV	Yes	Ion Torrent specific
VarDict [18]	SNV, indel, SV	Yes	Heuristic threshold
VarScan2 [9]	SNV, indel	Yes	Heuristic threshold
Virmid [29]	SNV	No	Joint genotype analysis

JointSNVMix. Roth et al. *Bioinformatics* **28**:907-13 (2012)

Xu. *Comput Struct Biotechnol.* **16**:15-24 (2018)

Somatic SNV Detection using Joint Inference from Tumor-Normal Pairs

1.Latent variable state space

- 9 genotype pairs (k_n, k_t)
- $n, t \in \{AA, AB, BB\}$

Tumor, t

Normal, n \ k_t	AA	AB	BB
AA	0.01	0.95	0.00
AB	0.00	0.04	0.00
BB	0.00	0.00	0.00

2.Probability of the genotypes

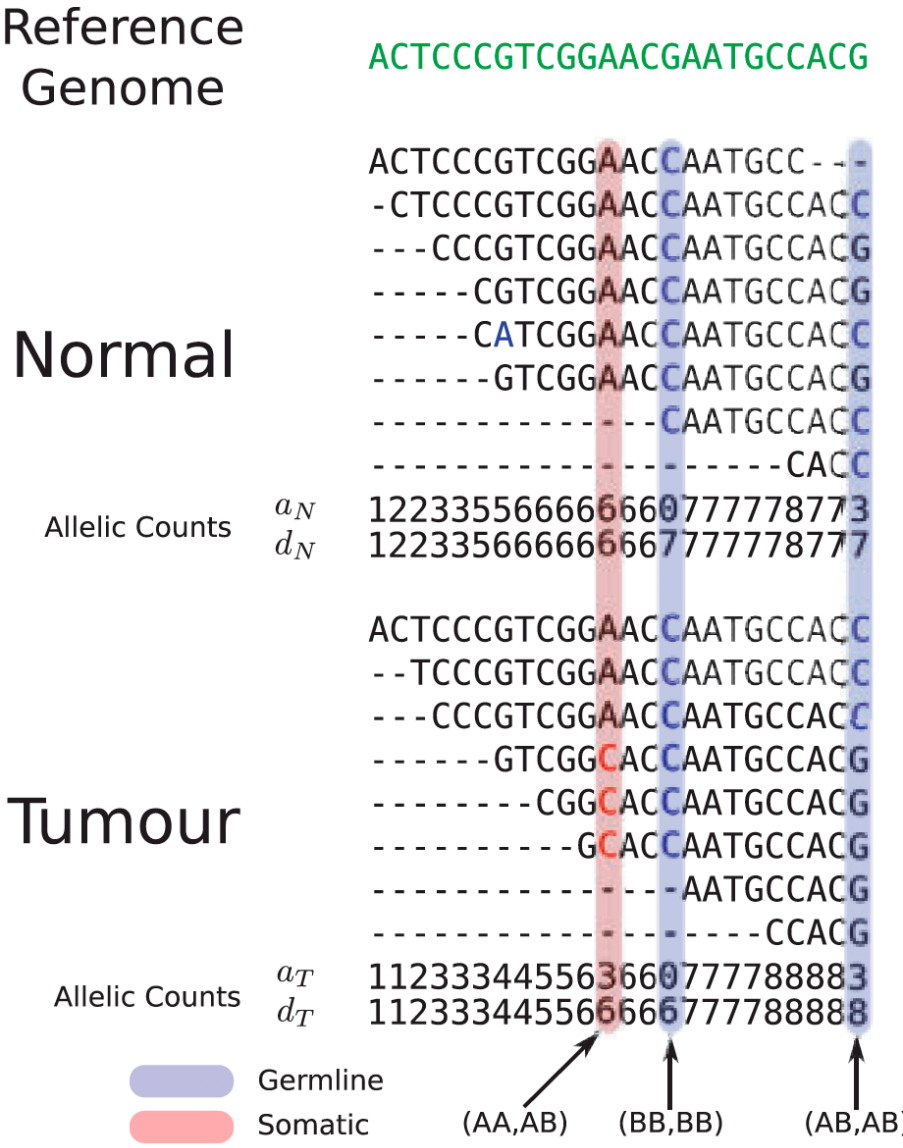
- 9 mixture weights $\pi_{(k_n,k_t)}$

3.Joint binomial mixture model

- 9-component mixture model

$$p(x_i^n, x_i^t \mid N_i^n, N_i^t, \mu_{1:K}^n, \mu_{1:K}^t) = \sum_{k_n=1}^{K} \sum_{k_t=1}^{K} \pi_{(k_n,k_t)} Bin(x_i^n \mid N_i^n, \mu_{k_n}^n) Bin(x_i^t \mid N_i^t, \mu_{k_t}^t)$$

- with 9 parameter tuples (μ^n, μ^t)



Roth et al. *Bioinformatics* **28**:907-13 (2012)

Homework #5: Single-nucleotide Genotype Caller

Implement a standard binomial mixture model described in Lecture 2.

- Learn the parameters and infer the genotypes
- Annotate the mutation status for a set of genomic loci.
- Expected outputs for each question will be provided so that you can check your code.
- RStudio Markdown and Python Jupyter Notebook templates provided.

Due: May 5th, 2022

Virtual Office Hours with Anna-Lisa Doebley (adoebley@uw.edu)

- May 3 @ 3pm (*new time*)
- May 6 @ 2pm
- Zoom Meeting ID: 446 356 7725 (Passcode: GS541)