Data Introduction

- Main focus is classification of two type of leukemia:

  acute myeloid leukemia (AML)

  acute lymphoblastic leukemia (ALL)
- Initial leukemia data set consisted of 38 bone marrow samples (27 from ALL and 11 from AML)

AML Biology Background

- a cancer of the blood and bone marrow
- affects various white blood cells
- Leukemic cells accumulate in the bone marrow, replace normal blood cells and spread to the liver, spleen, lymph nodes, central nervous system, kidneys and gonads.
- the most common type of acute leukemia in adults

ALL Biology Background

- a malignant (clonal) disease of the bone marrow in which early lymphoid precursors proliferate and replace the normal hematopoietic cells of the marrow
- distinguished from other malignant lymphoid disorders by the immunophenotype of the cells
- Immunochemistry, cytochemistry, and cytogenetic markers also may aid in categorizing the malignant lymphoid clone.

Importance of Molecular Classification

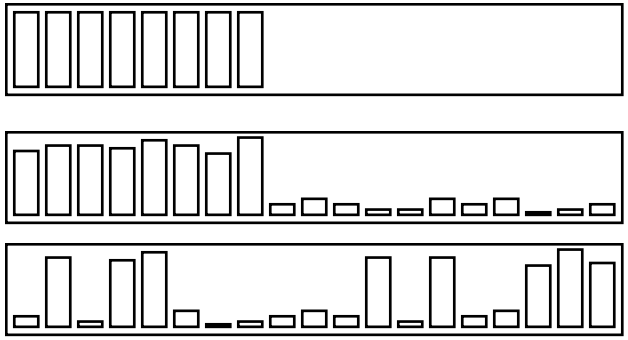
- Molecular marker help to diagnosis disease quickly and accurately
- Accurate classification help accurate diagnosis and pick up proper treatment
- Different types of cancer needs different treatments, most of the treatments are not replaceable
- Microarray and related statistical analysis would help to find good gene markers and set up accurate cancer classification

Clustering by Correlation

- Leukemia Study. 885 genes, 72 conditions
- Known partition of conditions into two clusters, AML and ALL
- How well does data agree with given partition?

Ideal Gene Expression

- Ideal gene for a partition
- Expression for one gene
- Expression for another
- Which one agrees with the partition?

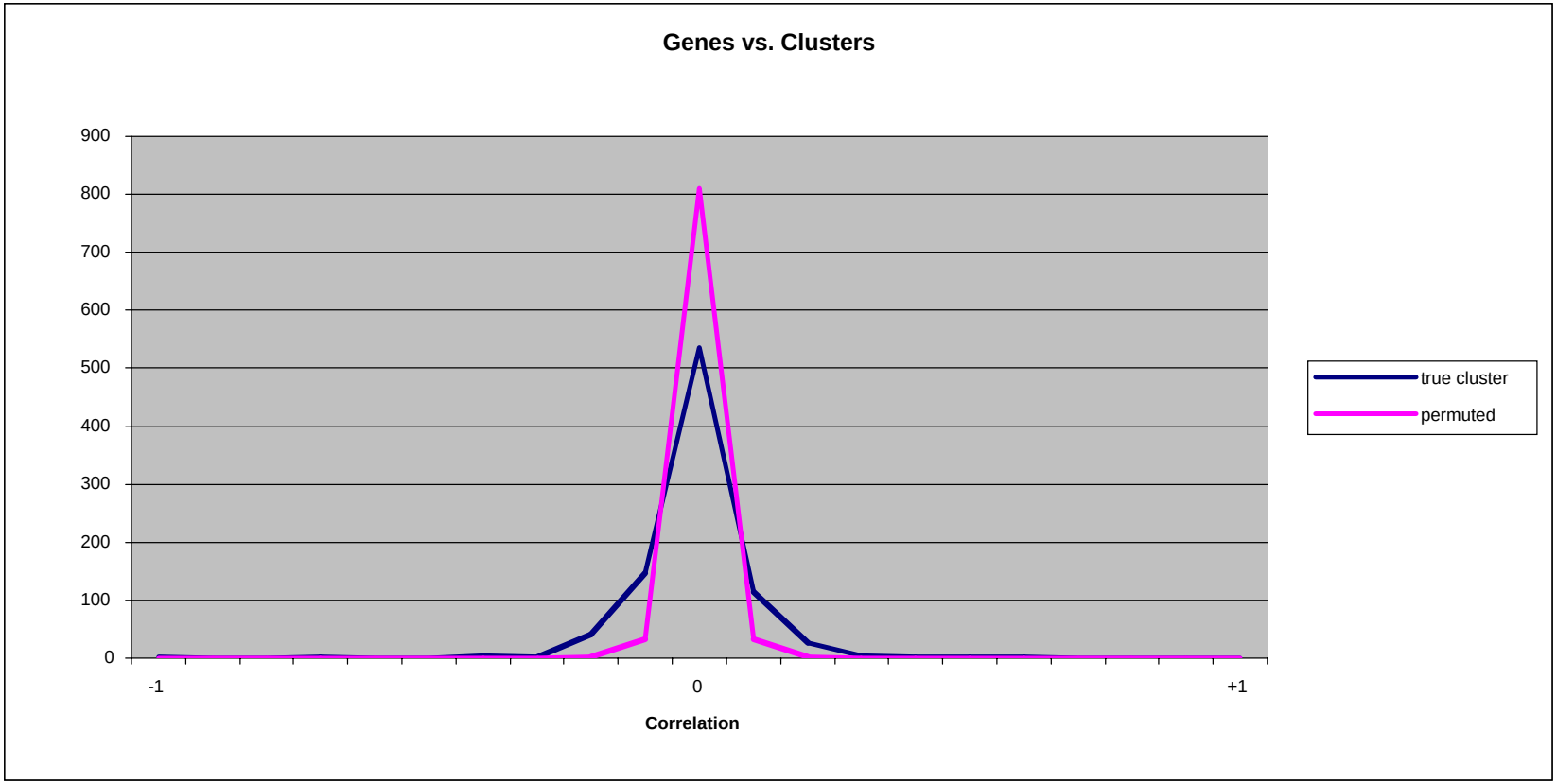


- Correlation $$P(g,c) = \frac{\mu_1(g) - \mu_2(g)}{\sigma_1(g) + \sigma_2(g)}$$

Judge partition from gene data

- Good partition $\Rightarrow$ lots of large correlations
- Bad partition $\Rightarrow$ not many large correlations
- But what's a large correlation?
- Get 'typical' distribution from permuted partitions

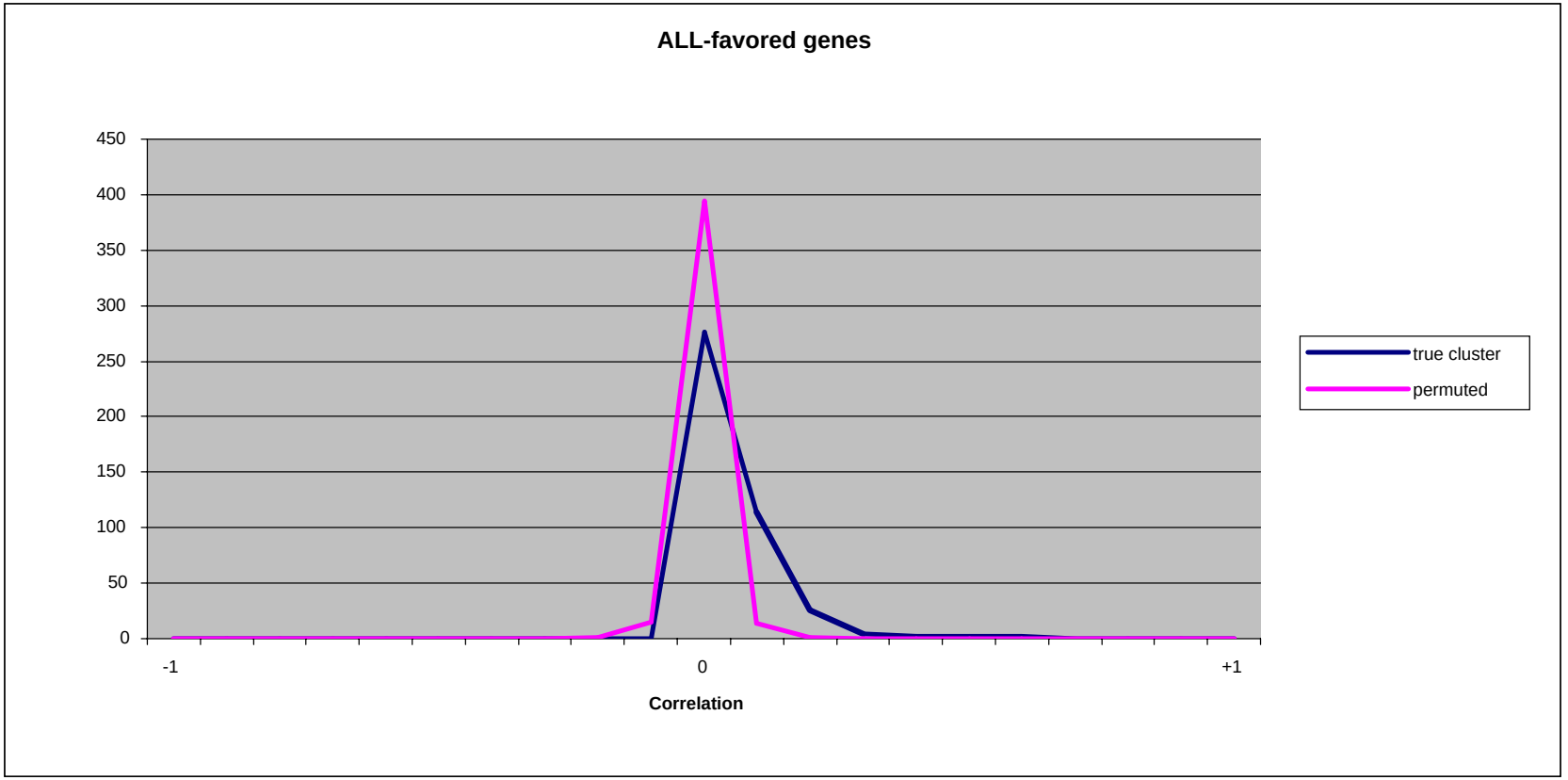
Distribution of correlation, complete set of genes



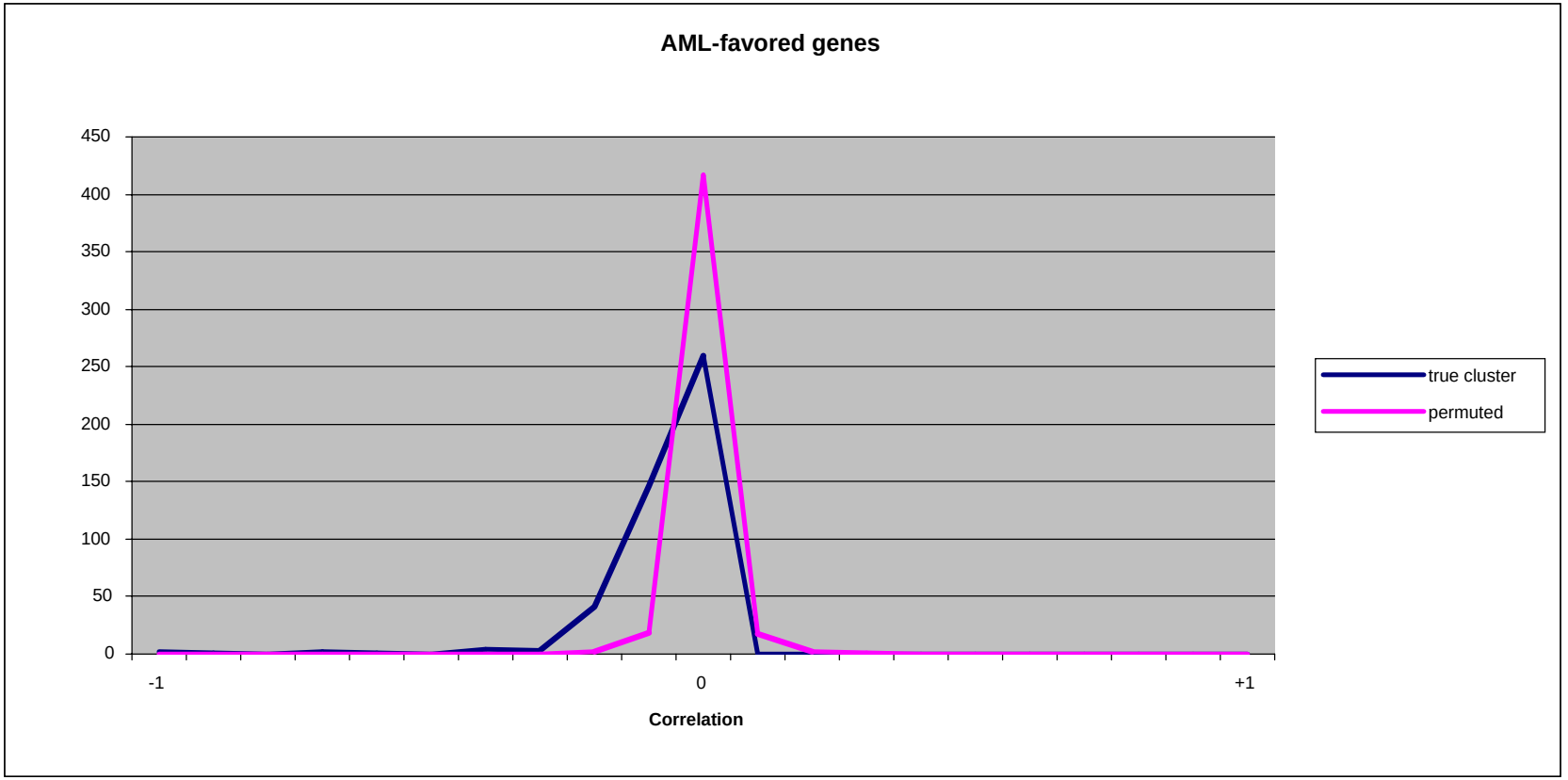
Favored genes

- Each gene has a cluster it favors
- Do genes maintain favoritism in permuted partitions?

Distribution of correlation, genes favoring ALL condition



Distribution of correlation, genes favoring AML condition



Genetic Algorithm for finding idealized expression pattern

- **Genetic Algorithm** is a stochastic search /optimization process that models evolution

Problem Description

- **Idealized expression pattern**
- We define an idealized expression pattern c = (c1, c2, .., cn), where ci = 1 or 0 according to whether the ith sample belongs to class ALL or class AML.
- A class distinction is represented by this idealized expression pattern c.
- Each gene is represented by an expression vector v(g) = (e1, e2, .., en), where ei denotes the expression level of gene g in ith sample in the initial set S of samples.

Correlation between a gene and a class distinction

- We define a correlation between a gene g and a class distinction c
- $P(g, c) = (M1(g) - M2(g)) / (SD1(g) + SD2(g))$, where M1(g), SD1(g), M2(g), SD2(g) denote the means and SDs of the log of the expression levels of gene g for the samples in class ALL and class AML, respectively.
- This reflects the difference between the classes relative to the SD within the classes.
- Large values of $|P(g, c)|$ indicate a strong correlation between the gene expression and the class distinction, while the sign of P(g, c) being positive or negative corresponds to g being more highly expressed in class ALL or class AML.

- We are trying to find **a class distinction *represented by an idealize expression pattern c* which has the strong correlation with gene expression *using the Genetic Algorithm***

Overview of the Algorithm

- Here is our genetic algorithm for finding fittest class distinction:
- create initial population of fixed size *p*;
- *do* {
- parents <- selection(population)
- population <- reproduction(parents)
- } *until* (stopping condition);
- report the answer;
- **[Genetic Algorithm for finding fittest class distinction]**
- Our genetic algorithm is ***generational*** genetic algorithm which replaces the whole population per generation (vs. ***steady-state*** genetic algorithm generates only one offspring per generation)

Problem Encoding:

- Each solution to our problem is represented by a chromosome, which is a binary string
- A chromosome corresponds to a class distinction
- Chromosome = class distinction(partition) = solution
- The number of genes in the chromosome = 72 where 72 is the sample size
- Each gene corresponds to a class type (in this case either class ALL or class AML)
- A gene has value 0 if the corresponding sample belongs to class ALL, and has value 1 if the corresponding sample belongs to class AML

Initialization

- Our algorithm creates *p* solutions at random
- We set the population size *p* to be 20

Parents Selection

- Each chromosome is selected as a parent with a probability that is proportional to its fitness value
- This is a very common parent selection scheme called ***proportional selection***

Fitness value:

We assign to each chromosome (= class distinction) a fitness value calculated as following:

fit(c) = sum (over row i) abs(m0(i) - m1(i)) / (sd0(i) + sd1(i))

where m0(i),sd0(i) are mean and standard deviation of the log of the samples in row i that are in type-ALL columns and m1(i),sd1(i) are mean and standard deviation of the log of the samples in row i that are in type-AML columns

Crossover and Mutation Operators

- We use 1 crossover point and 1% mutation rate which are the best ones from our experiments on the various crossover points and mutation rates

Replacement Scheme

- New generations were created from the previous generation. The two most fit strings are copied without modification into the next generation.

  They contribute to selection. After the full generation has been selected, any duplicates are thrown out, replaced by freshly random strings.

Stopping Condition

- We set the stopping condition as 10000 generations

Experimental Result

Convergence:

- Maximum fitness in the first generation was .855. This rose to about 1.5 in the first 100 generations, and was around 1.75 by 250 generations. After that, improvement was pretty slow. The 2.00 barrier was exceeded around generation 2,000, and the 2.011 score (the top score) was achieved at generation 2,232

Conclusion:

- The program discovered a partition that scored within .15 percent of the 'true' partition. The 'true' partition had a score of 2.014, the program's partition scored 2.011. The partitions disagreed in seven places: samples 35, 54, 60, 66, 67, and 72.