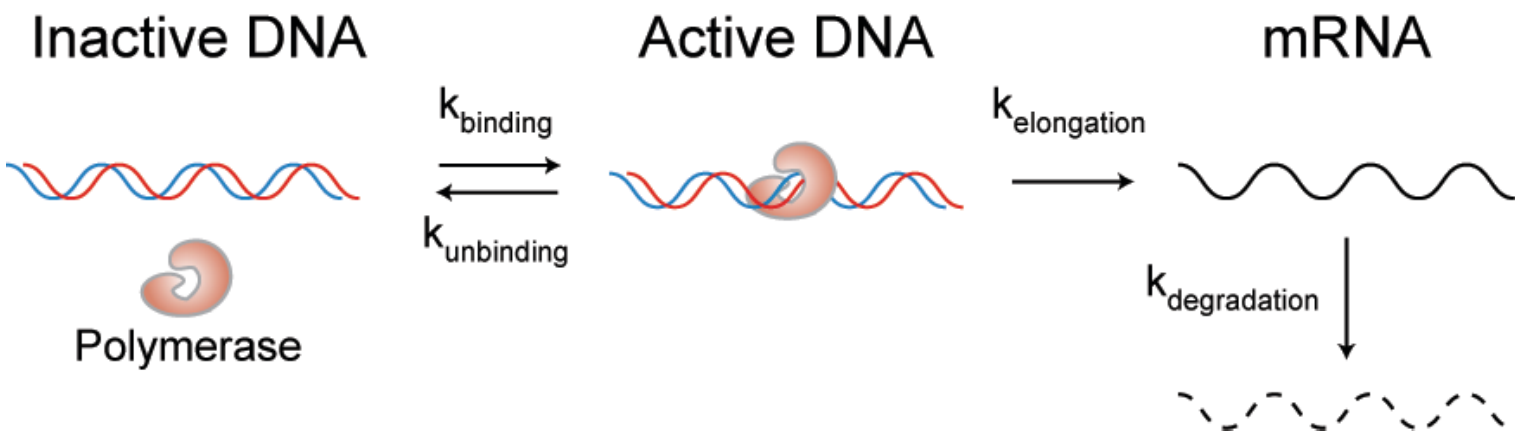
Parameterizing heterogeneity

Limitations of stochastic profiles

- Still not single-cell resolution transcriptomics
- Only make a binary decision as to whether a gene is heterogeneously regulated, but no information of the extent of heterogeneity

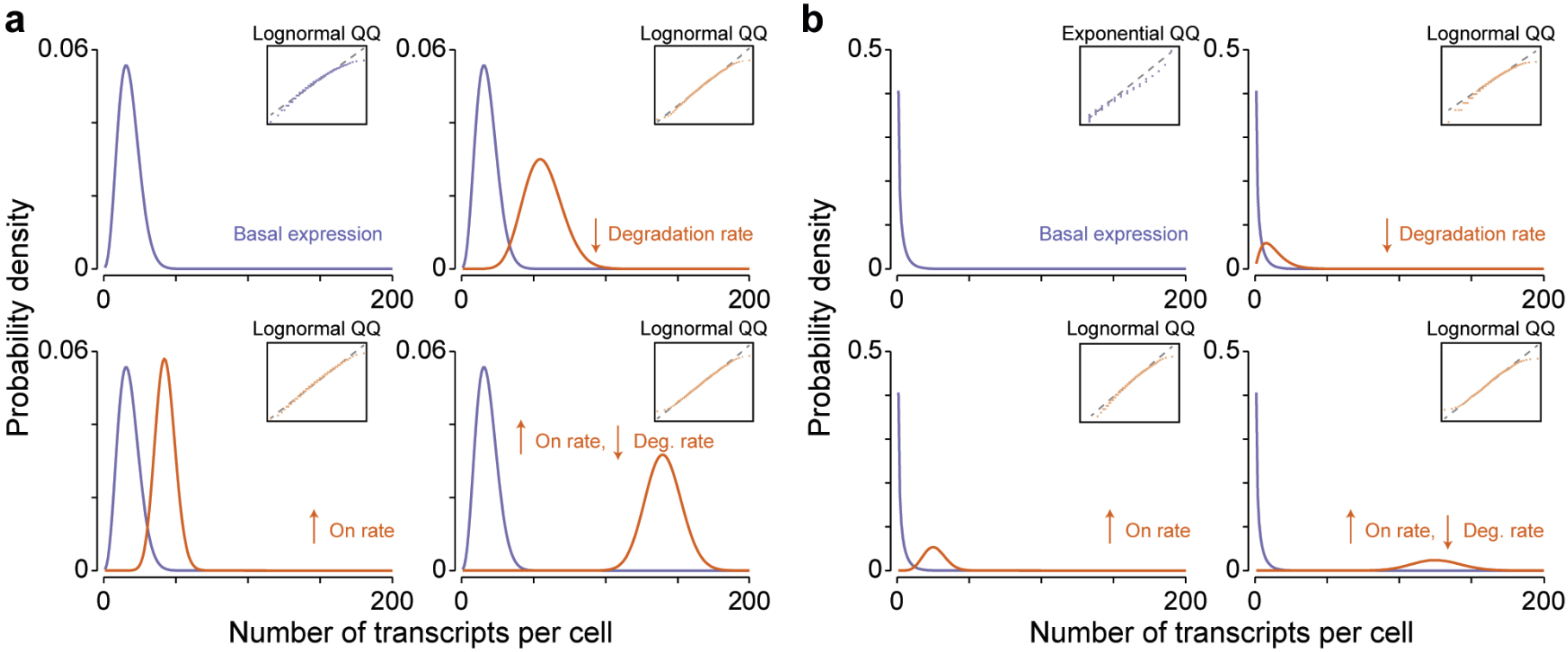
- Solution: use mathematical modeling to determine the single-cell expression distribution

Single-cell gene expression and regulation



PNAS **111**, E626-35 (2014); *PLoS Biol* **4**(10), e309 (2006)

Probability models of gene regulation



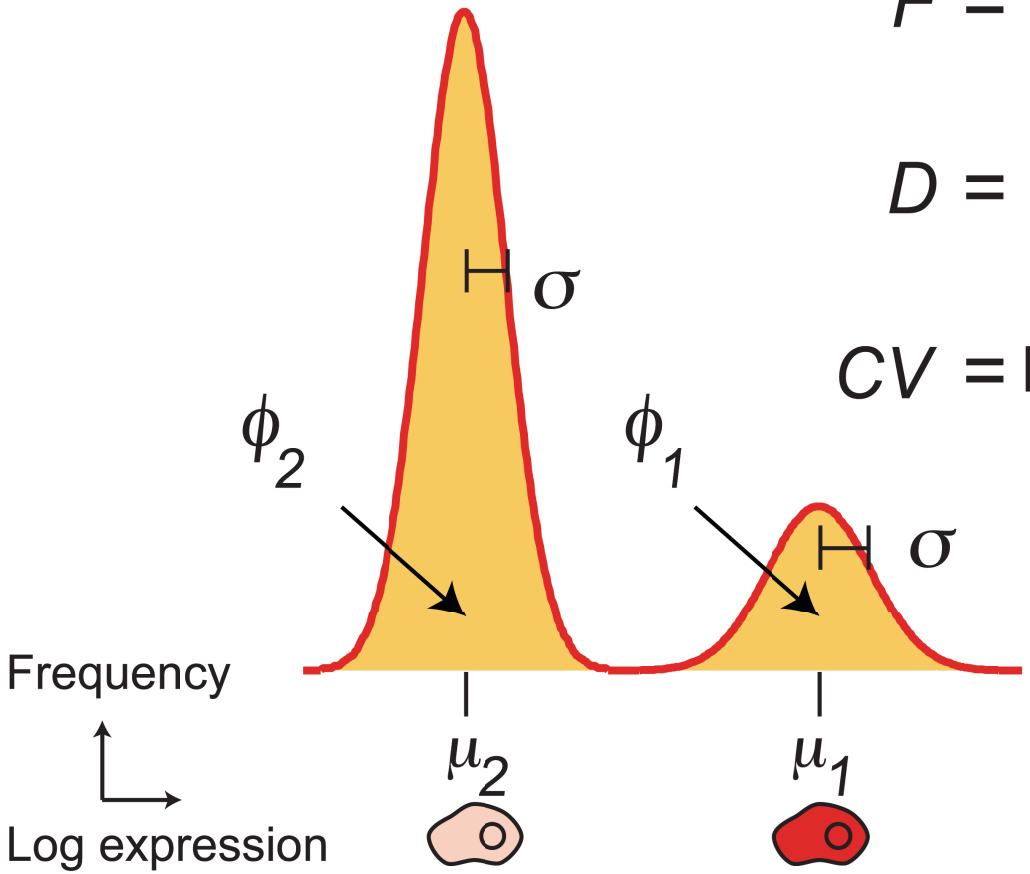
PNAS **111**, E626-35 (2014)

Dichotomous gene expression model

$$F = \frac{\phi_1}{\phi_1 + \phi_2}$$

$$D = \frac{\mu_1}{\mu_2}$$

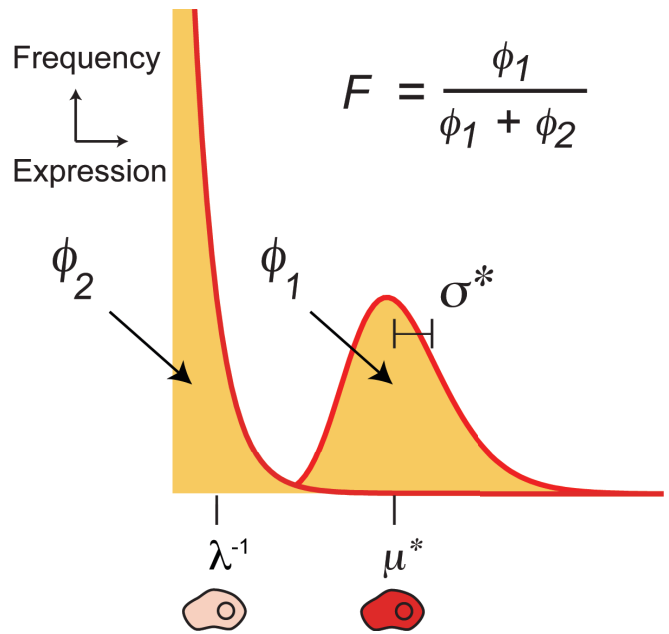
$$CV = \mathrm{Exp}(\sigma^2 + 1)$$



PNAS **111**, E626-35 (2014); *Nat Meth* **7**, 311-317 (2010)

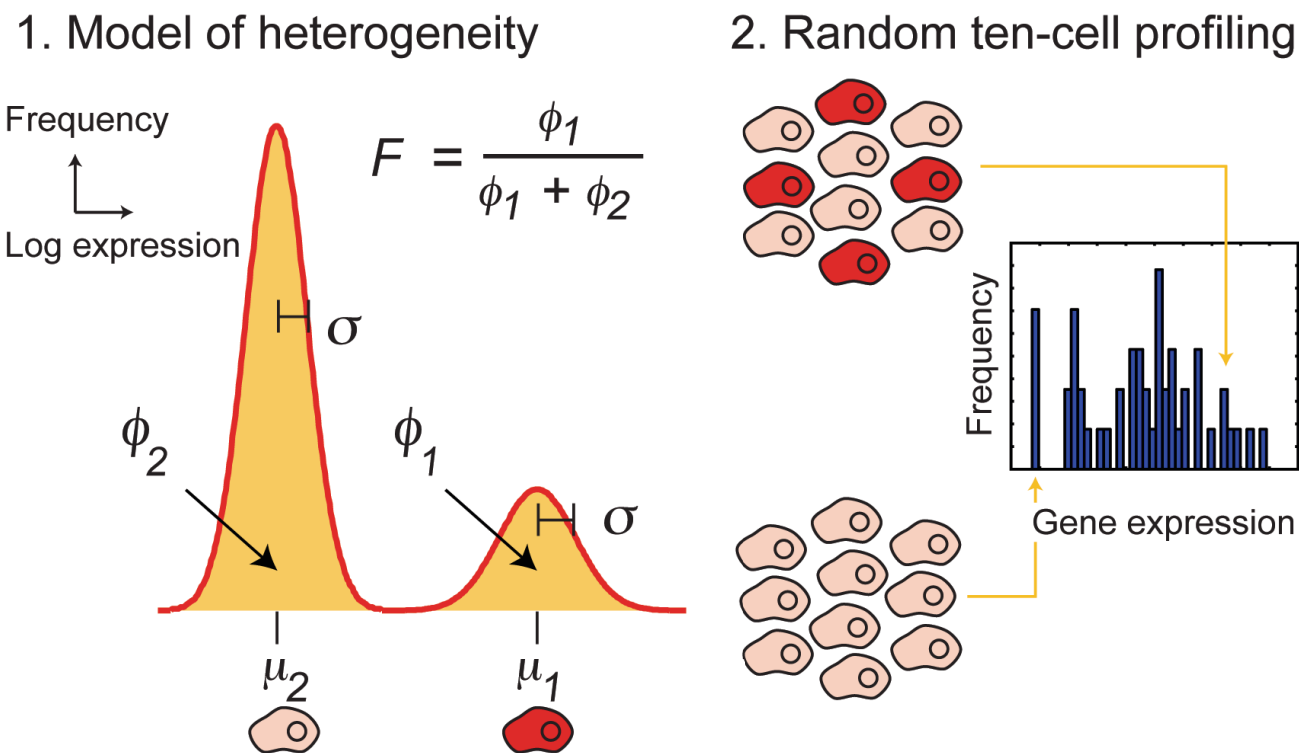
Dichotomous gene expression model

- Also have a model where the CVs are different for each lognormal population (rLN-LN)
- Also have a model where low-expression population is exponentially distribution



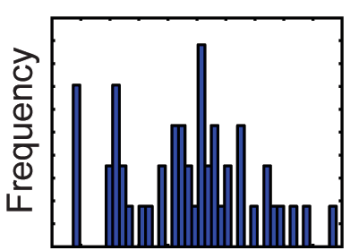
PNAS **111**, E626-35 (2014)

Parameterizing heterogeneity with Maximum-Likelihood Inference



PNAS **111**, E626-35 (2014)

Parameterizing heterogeneity with Maximum-Likelihood Inference



3. Maximize likelihood of pdf f(x)

measurements

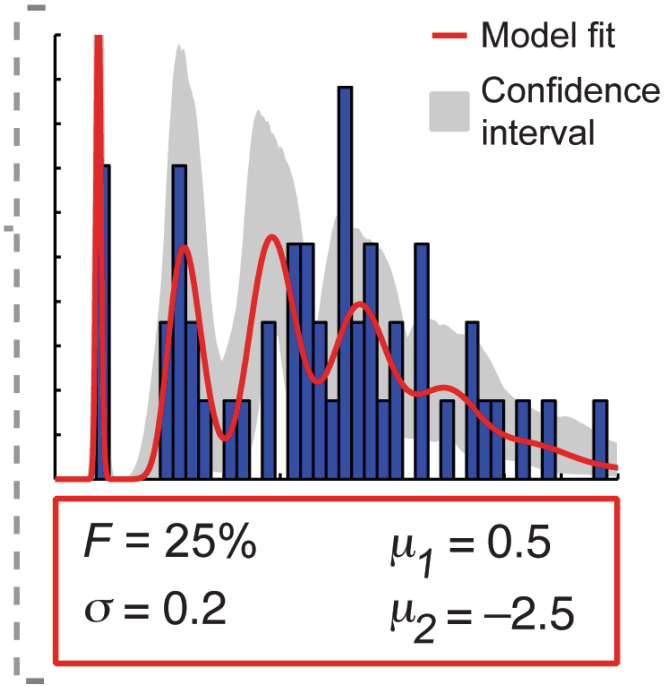
$$\text{Max} \sum_{i=1} \log f(x_i \mid \mu_1, \mu_2, F, \sigma)$$

$$f(x) = \sum_{j=0}^{10} \binom{10}{j} F^{j} (1 - F)^{10-j} z_{j,\,10-j}(x)$$

$z_{j,\,10-j}(x)$ is pdf of $Z_1 + \ldots + Z_{10}$, where

$$Z_c \overset{\text{ind}}{\sim} \begin{cases} LN(\mu_1, \sigma^2) & 1 \le c \le j \\ LN(\mu_2, \sigma^2) & j < c \le 10 \end{cases}$$

4. Estimate parameters

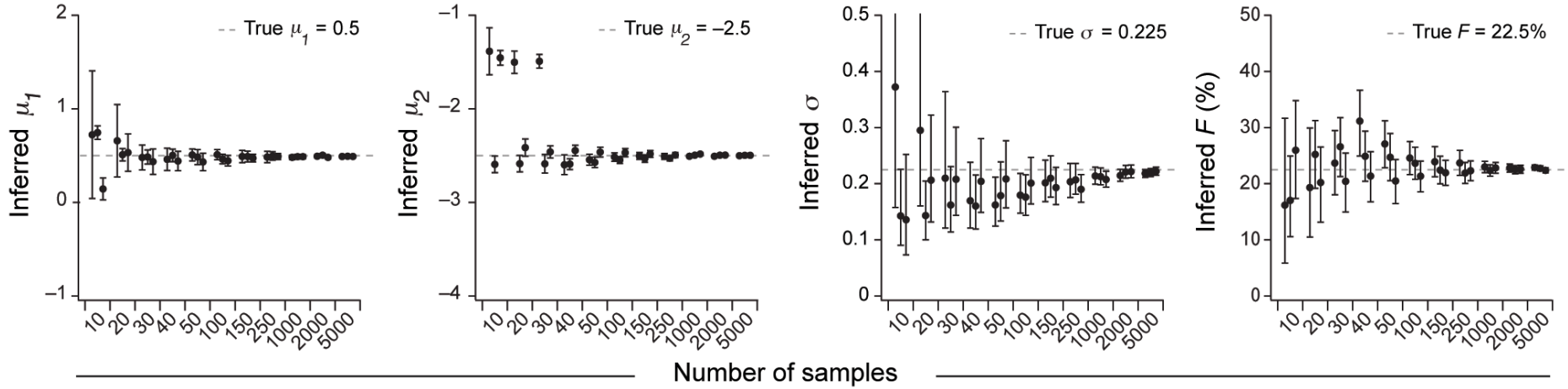


PNAS **111**, E626-35 (2014)

Maximum-likelihood estimation (MLE)

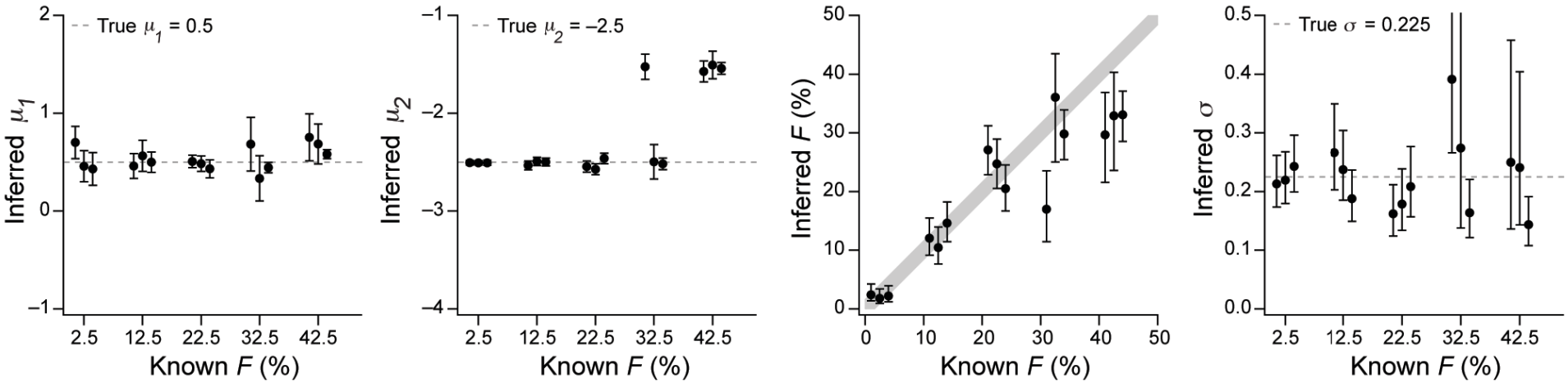
- Estimate parameters of a statistical model from a given data set
- Calculate the likelihood of the data
  - Probability of observing sample values given a parameter
- Maximizing the likelihood identifies the best parameter set of the model for the data

Sample-size requirements for inference



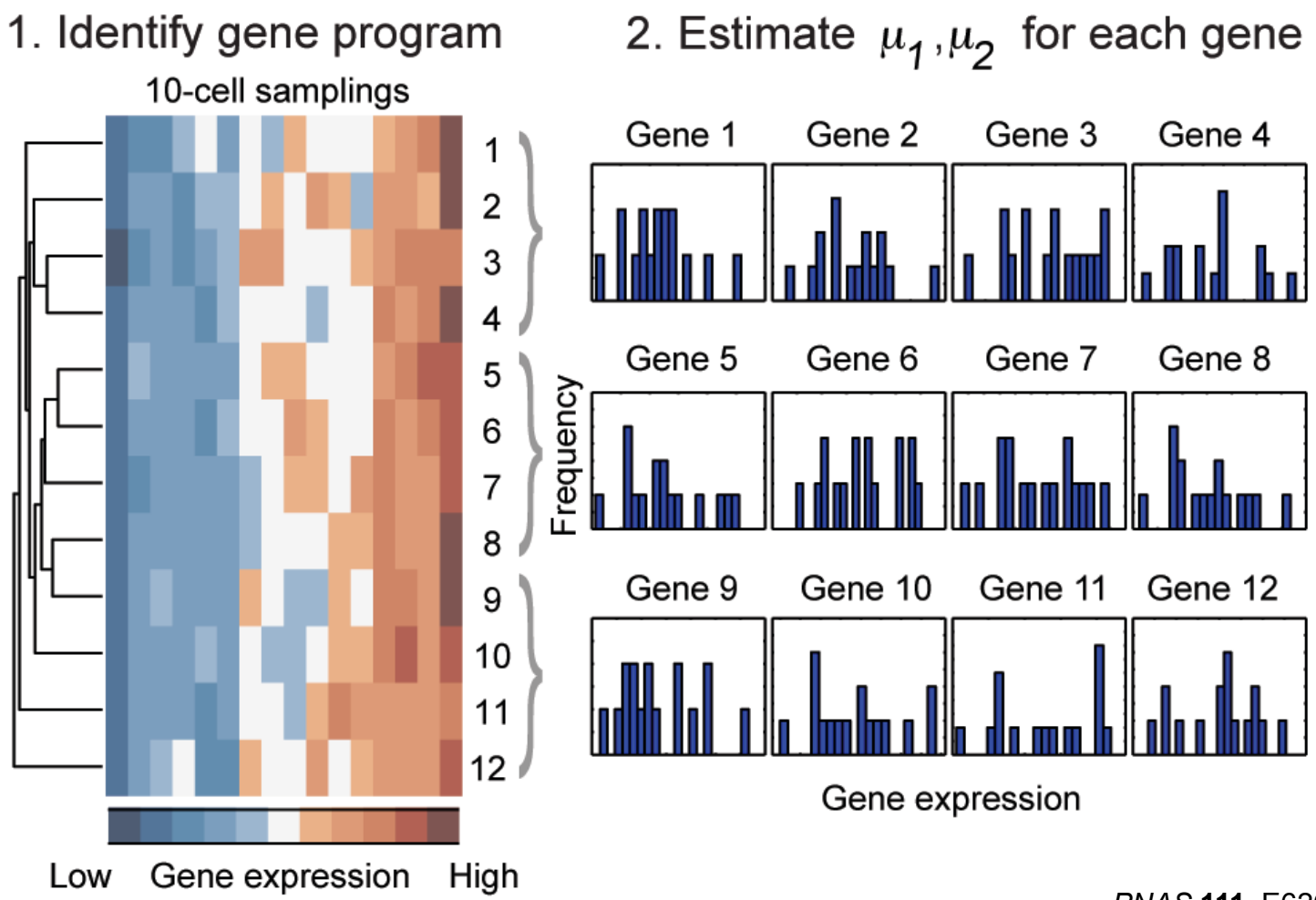
PNAS **111**, E626-35 (2014)

Parameter sensitivity in LN-LN estimation



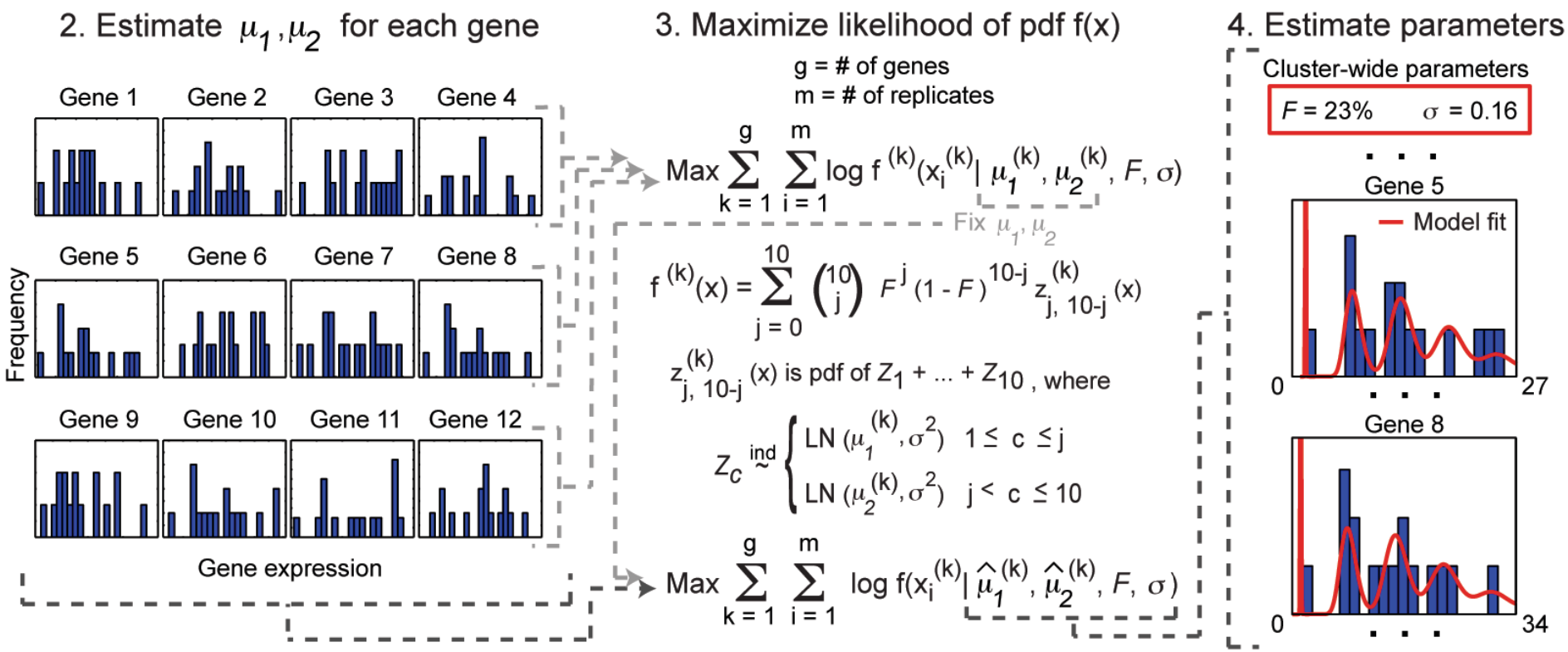
PNAS **111**, E626-35 (2014)

Globally parameterizing heterogeneities



PNAS **111**, E626-35 (2014)

Globally parameterizing heterogeneities



NOTE: DATA IS NOT LOG-TRANSFORMED

PNAS **111**, E626-35 (2014)

General work flow

- Input in data
  - Single-gene (e.g., qPCR) or cluster of genes
  - Not log transformed
- Perform inference with three mixture models
  - LN-LN, rLN-LN, EXP-LN
- Use Bayes' Information Criterion (BIC) to determine best model for data

Installing R package

- stochprofML package on CRAN
- Install package from source file on thumbdrive or by downloading from CRAN

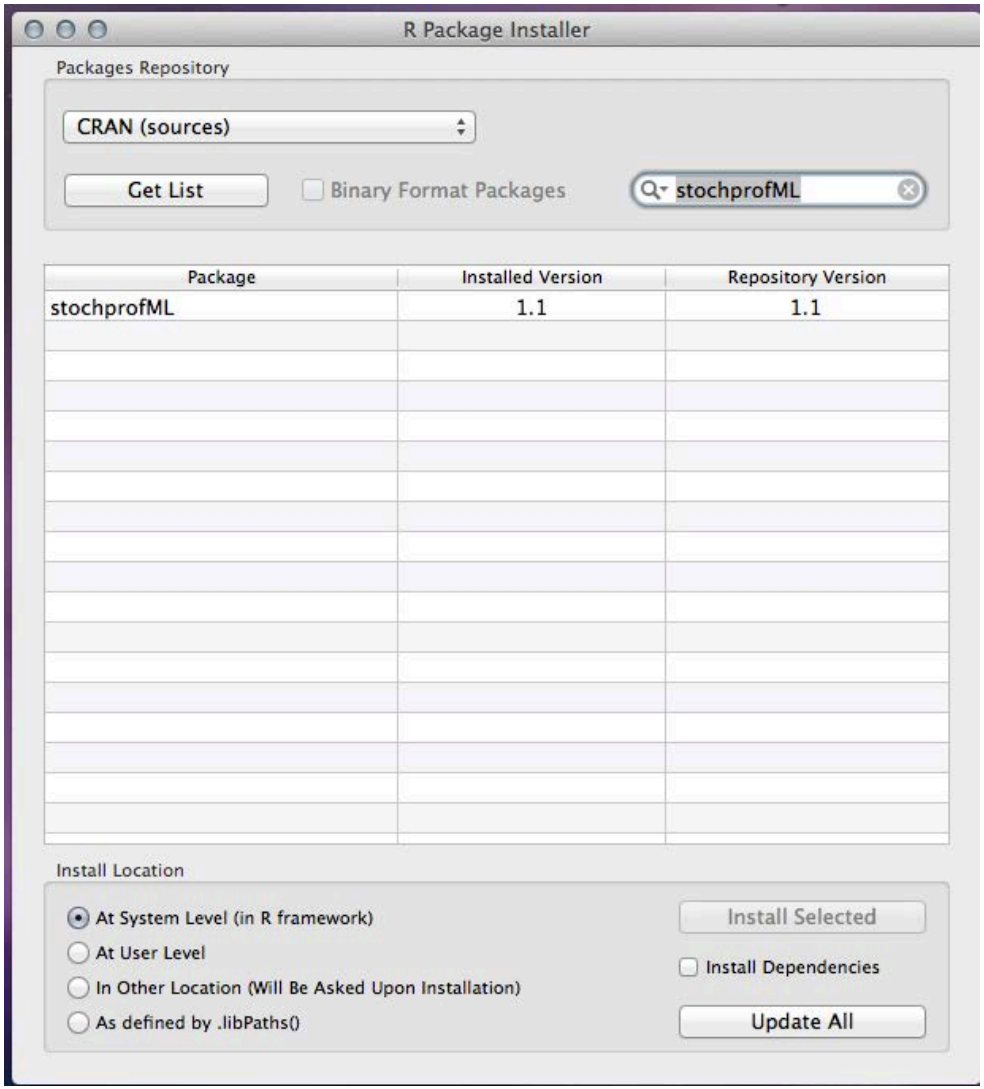


Image using R-Studio

Using R package

- Monte Carlo simulation variables to define
  - Model to use
    - LN-LN, rLN-LN (will take longer to run), EXP-LN (will take very long to run)
  - Number of cells sampled
  - Number of samples
  - F, μ_1, μ_2, σ

Using R package

- Use function **stochasticProfilingData()** to simulate data

- Use function **stochasticProfilingML()** to perform maximum likelihood inference for data (both simulated and real)

Exercises and discussion

- Generate simulated data and perform inference for LN-LN model
  - rLN-LN will take ~twice as long, EXP-LN will take ~1 hr
- Infer SOD2 expression frequency for LN-LN model
  - Data is stored in the package as the vector “sod2”
- During overflow or free time
  - Infer parameters for a cluster of 4 simulated genes

Key R functions

- r.sum.of.mixtures.**MODEL**(k, n, *F*, mu, sigma)
  - Simulate data
    - k = # of samples
    - n = # of cells
- stochprof.loop
  - Main function, takes in several arguments (see help file) and initiates inference algorithm

Using R package

samples <- c(**NUMBER OF SAMPLES**)
numberofcells <- c(**NUMBER OF CELLS PER SAMPLE**)
numberofpopulations <- c(**NUMBER OF SUBPOPULATIONS**)
p.vector <- c(***F*, 1 - *F***)
mu.vector <- c(μ_1, μ_2)
sigma.vector <- c(σ, σ) #sigmas are different in rLN-LN model
data <- r.sum.of.mixtures.LNLN(samples, numberofcells, p.vector, mu.vector, sigma.vector)
#data <- **qPCR data OR matrix of profiles where columns are genes, rows are samples**
data <- matrix(data, nrow = samples, ncol = 1) #columns are gene, rows are samples
stochprof.loop("LN-LN", data, numberofcells, numberofpopulations)