Modeling & Simulation (Computational Immunology)

Steven H. Kleinstein

YALE Pathology Informatics

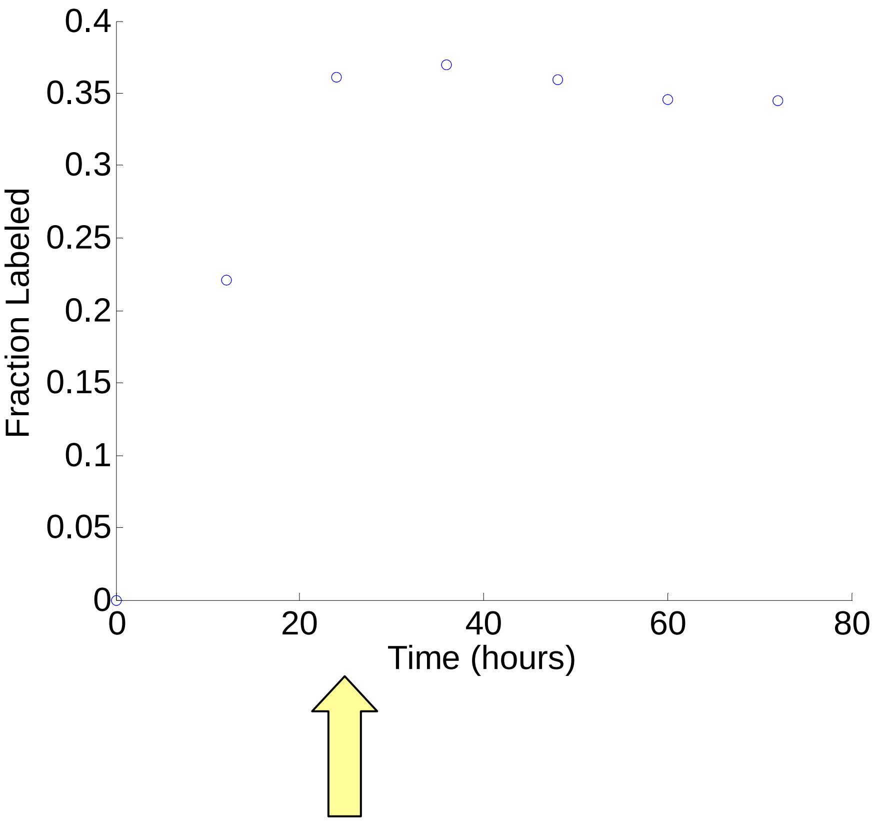
Department of Pathology
Yale University School of Medicine

steven.kleinstein@yale.edu

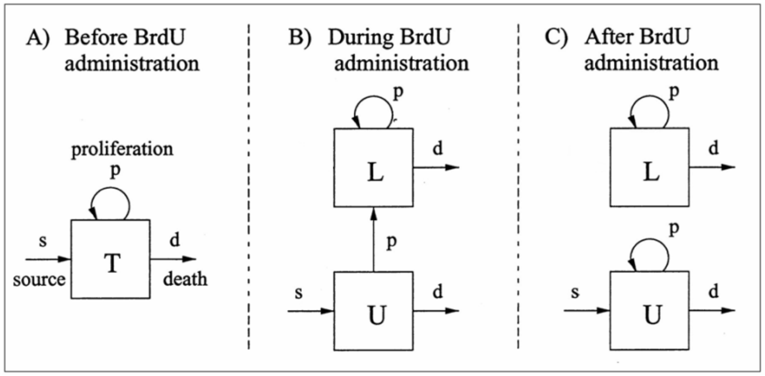
March 24, 2010

Simulated Experiment

Demonstrate full cycle of fitting model to data to estimate parameters



BrdU withdrawn


A) Before BrdU administration
proliferation
p
s
source
T
d
death
B) During BrdU administration
p
L
d
p
s
U
d
C) After BrdU administration
p
L
d
p
s
U
d

<u>Parameters used to create synthetic data</u>

s = 0.003 per hour

p = 0.01 per hour

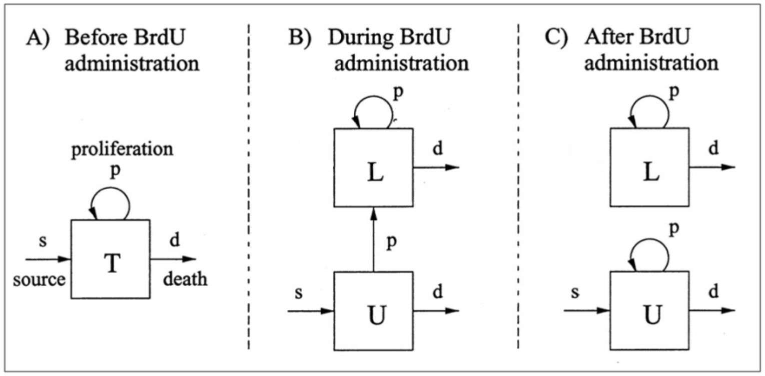
d = p + s (to achieve steady state)

Random noise added to each data point

How can we estimate flow/proliferation/death rates?

Simulating the BrdU Labeling Model

Use integration functions (e.g., ode45 in MATLAB)


A) Before BrdU administration
proliferation
p
s
source
T
d
death
B) During BrdU administration
p
L
d
p
s
U
d
C) After BrdU administration
p
L
d
p
s
U
d

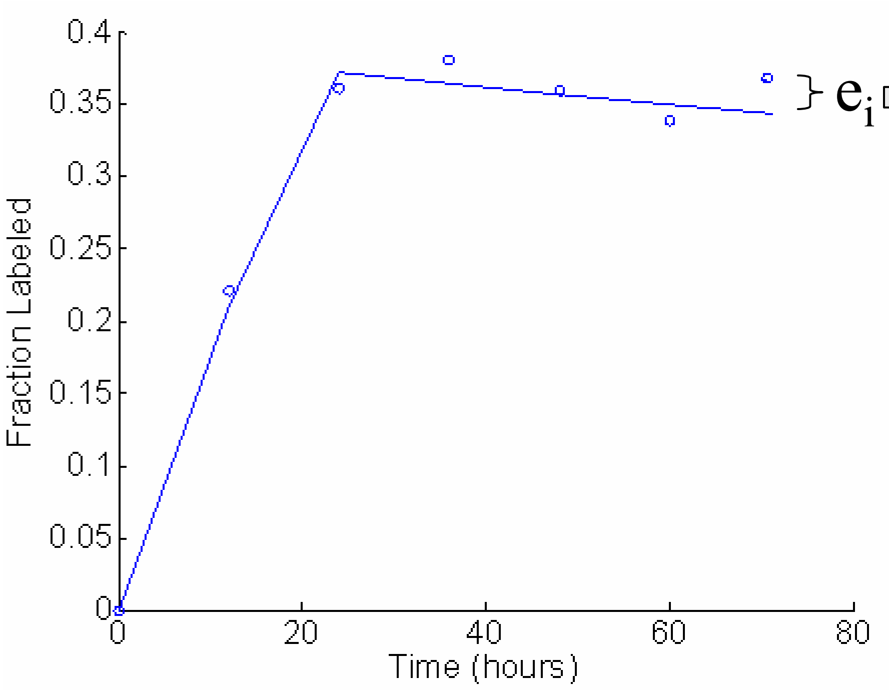
```
Yin = [1 0];        % Initial Conditions [unlabeled labeled]
pr = [s p d tau];  % Model Parameters
t = [0,12,24,36,48,60,72];   % Times to evaluate
[T,Y] = ode45(@fode,t,Yin,opts,pr);
fl = Y(:,2) ./ sum(Y,2);  % Fraction labeled
```

```
function dy = fode(t, y, pr)
s = pr(1); p = pr(2); d = pr(3); tau = pr(4);
U = y(1); L = y(2);
dy = zeros(2,1); % Vector of derivatives
if (t<tau)  % During BrdU Administration (B)
    dy(1) =  s - p.*U - d.*U;                  % dbU/dt
    dy(2) =  2.*p.*U + p.*L - d.*L;        % dbL/dt
else        % After BrdU Administration (C)
    dy(1) =  s + p.*U - d.*U;                  %dbU/dt
    dy(2) =        p.*L - d.*L;                %dbL/dt
end
```

Simple models can be solved analytically -- faster

Fitting the Model to Experimental Data

Compare simulation and experiment using least-squares objective



Least-squares objective function

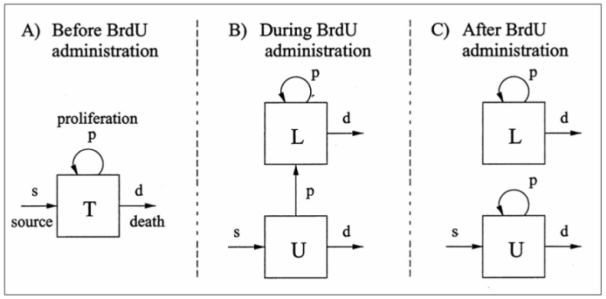
$$E = \sum_i \frac{(y_i - \hat{y}_i)^2}{VAR(y_i)}$$

Find parameters to minimize objective

Many options for how to optimize the fit

Fitting Models to Data in MATLAB

Several optimization functions available in many programming languages


A) Before BrdU administration
proliferation
p
s
source
T
d
death
B) During BrdU administration
p
L
d
p
s
U
d
C) After BrdU administration
p
L
d
p
s
U
d

```
pri = [.01 .01];  %Initial guess for parameter values to be fitted [s p]
[pr,fval,exitflag] = lsqnonlin (@efun,pri,[],[],options,fl_observed,t,tau);
s = pr(1); p = pr(2); % Optimal parameter values
```

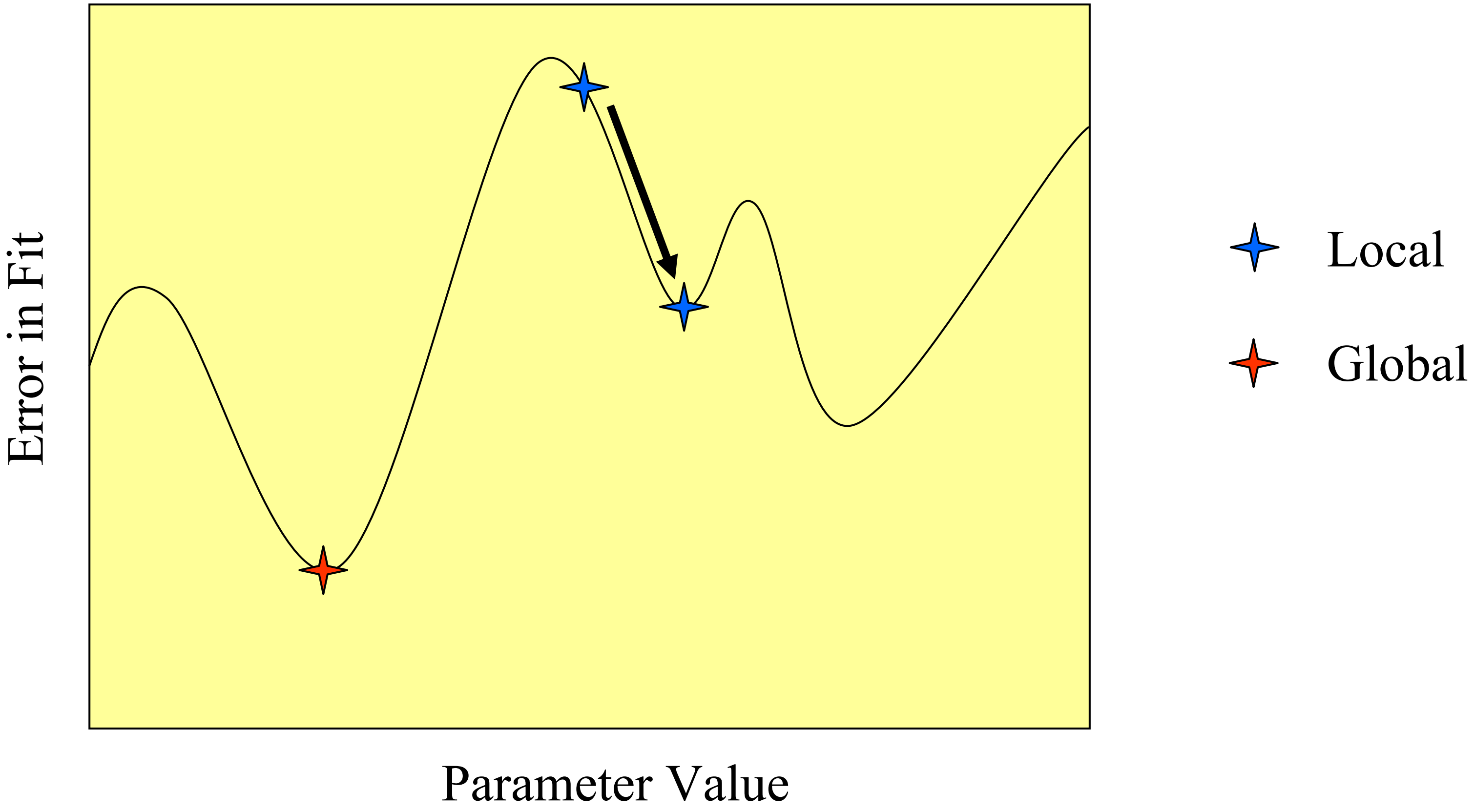
Optional parameters

```
function error = efun (pr,fl_observed,t,tau)
  s = pr(1); p = pr(2); d = s+p;                 % Assume steady-state
  [fl_predicted] = labelBrdU(s,p,d,tau,t);       % Function that simulates model
  error = sum((fl_predicted-fl_observed).^2);    % Least-squares objective
```

lsqnonlin, fminsearch, fmincon, fminbnd

Local and Global Optimization

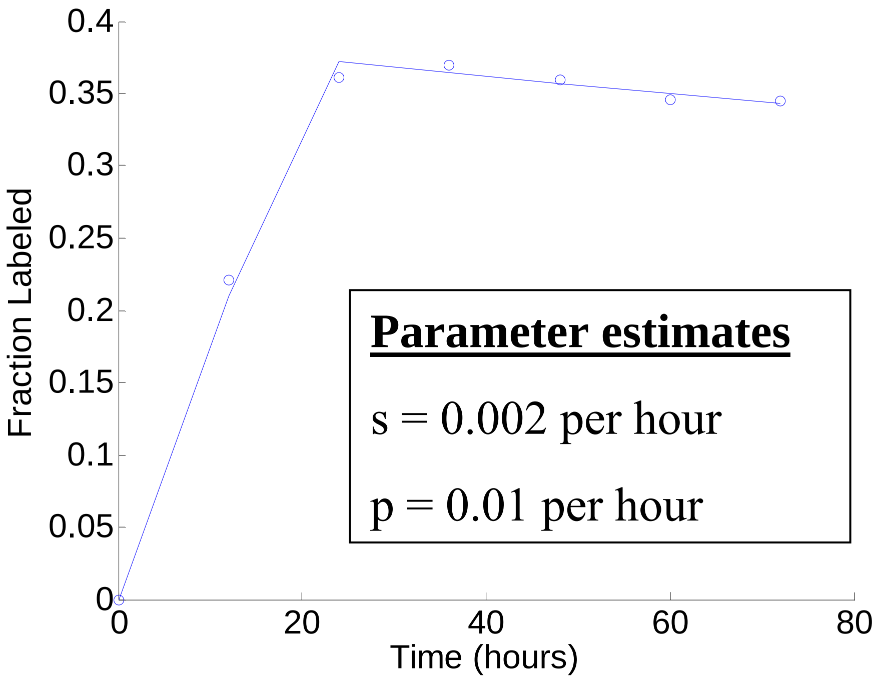
Local optimization techniques find optimal fit around given starting point



Global optimization attempts to avoid local minima

Optimal Parameter Estimates

Least-squares fit using lsqnonlin in MATLAB

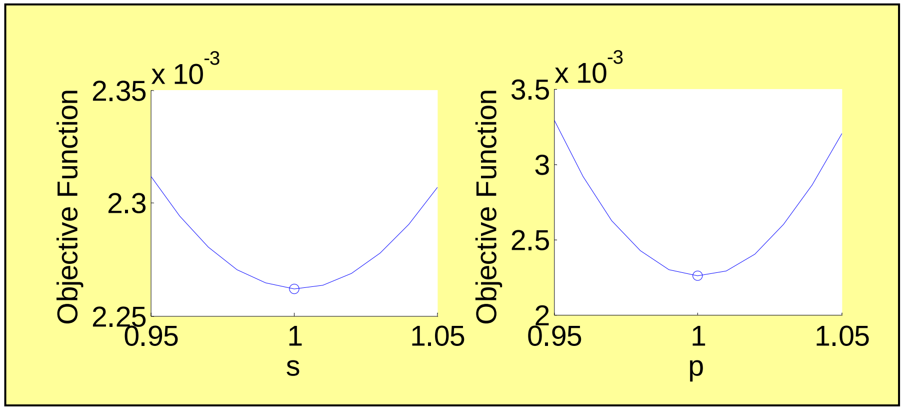


Parameter estimates

s = 0.002 per hour

p = 0.01 per hour

Plot local curvature to check minimization…



Recall, parameters used to create data:

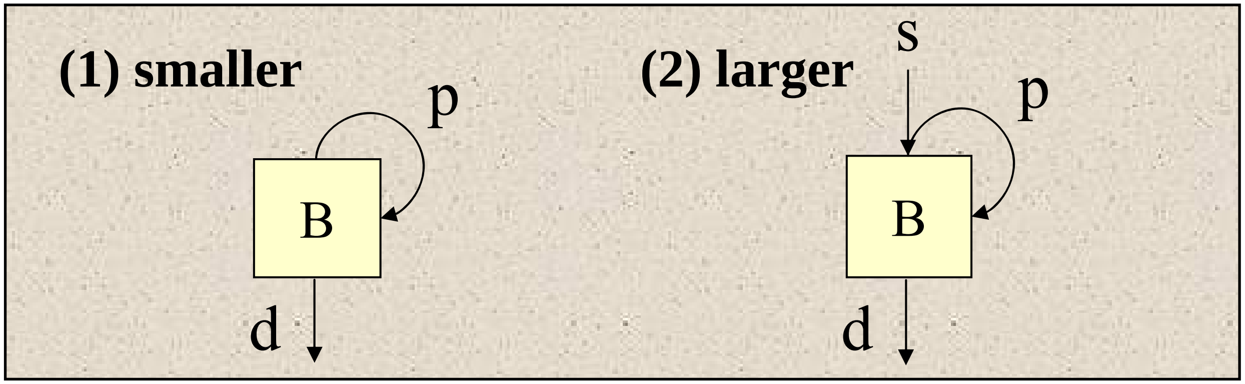
s = 0.003 per hour

p = 0.01 per hour

d = p + s (to achieve steady state)

Is inflow necessary to fit the data? Can we use simpler model?

Is inflow (s) significant?



Residual Sum of Squares $\boxed{RSS = \sum_i (y_i - \hat{y}_i)^2}$

$$F = \frac{\left(\text{RSS}_{\text{smaller}} - \text{RSS}_{\text{larger}}\right) / \left(df_{\text{smaller}} - df_{\text{larger}}\right)}{\text{RSS}_{\text{larger}} / df_{\text{larger}}}$$

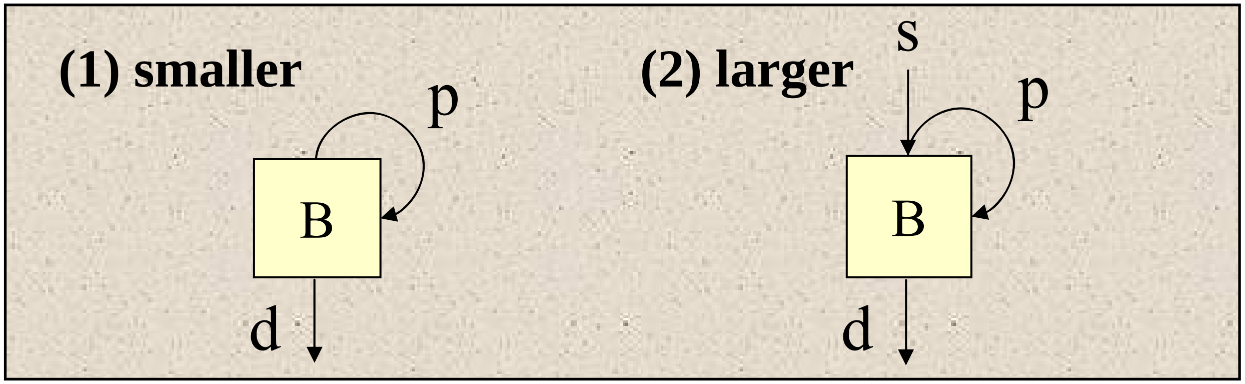
} Reduction in RSS per extra parameter

} Measure of 'noise' in model

Degrees of Freedom $\boxed{df = \#observations - \#\,parameters}$

F distribution with ($\text{df}_{\text{smaller}}$-$\text{df}_{\text{larger}}$, $\text{df}_{\text{larger}}$) degrees of freedom

Is inflow (s) significant?



$$F = \frac{\left(\mathrm{RSS}_{\mathrm{smaller}} - \mathrm{RSS}_{\mathrm{larger}}\right) / \left(df_{\mathrm{smaller}} - df_{\mathrm{larger}}\right)}{\mathrm{RSS}_{\mathrm{larger}} / df_{\mathrm{larger}}}$$

Reduction in RSS per extra parameter

Measure of ‘noise’ in model

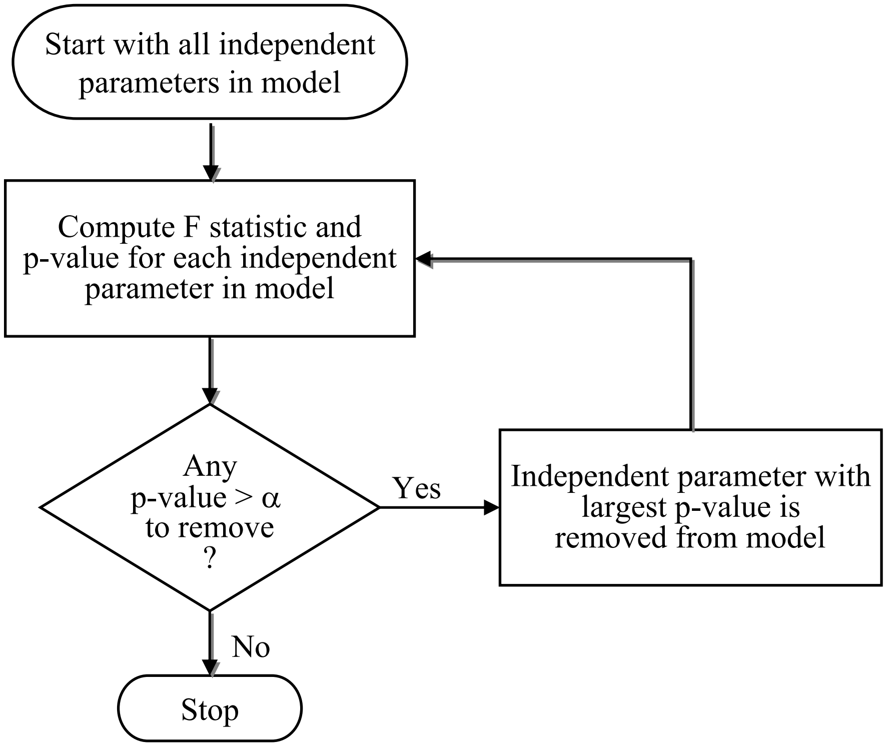
	Observations	Parameters	RSS	F test (1-fcdf in MATLAB)
(1) No flow (s=0)	6	1	9.38e-7	
(2) Including flow	6	2	0.95e-7	**53.1 (p<0.0004)**

Inflow (s) is important to explain observations

Building models with variable selection

F statistic determines if variable added or deleted from model

Backward Elimination


Start with all independent parameters in model
Compute F statistic and p-value for each independent parameter in model
Any p-value > α to remove ?
Yes
Independent parameter with largest p-value is removed from model
No
Stop

Other Variations:

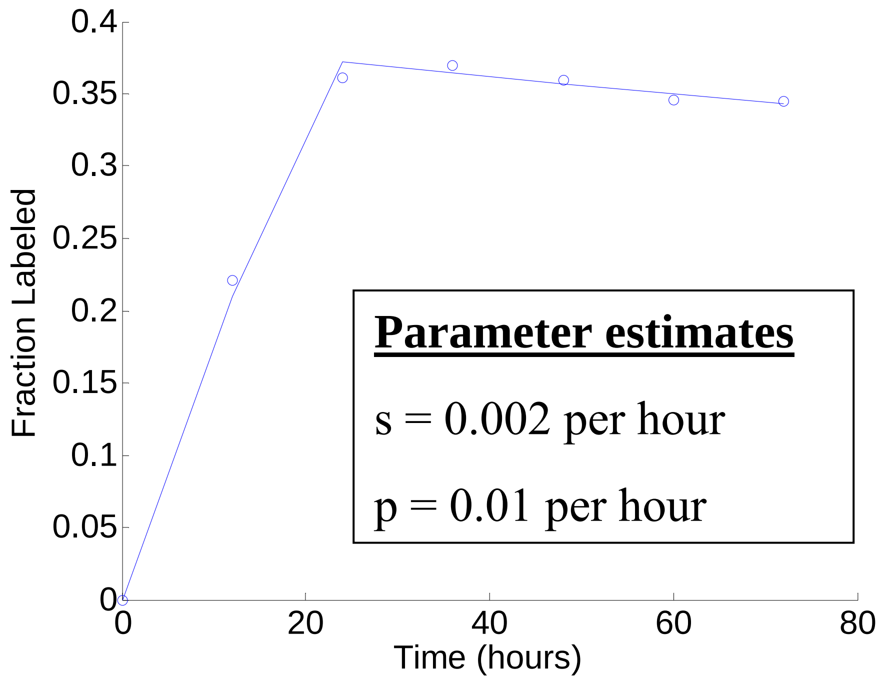
Forward selection: adds variables one at a time as long as significant F test.

Stepwise procedure: allows for removal of a parameter at each step

No guarantee that globally optimal model with be found (need all subsets, but prohibitive for large parameter space)

How much confidence to put in estimate?

Construct confidence intervals for model parameters



Parameter estimates

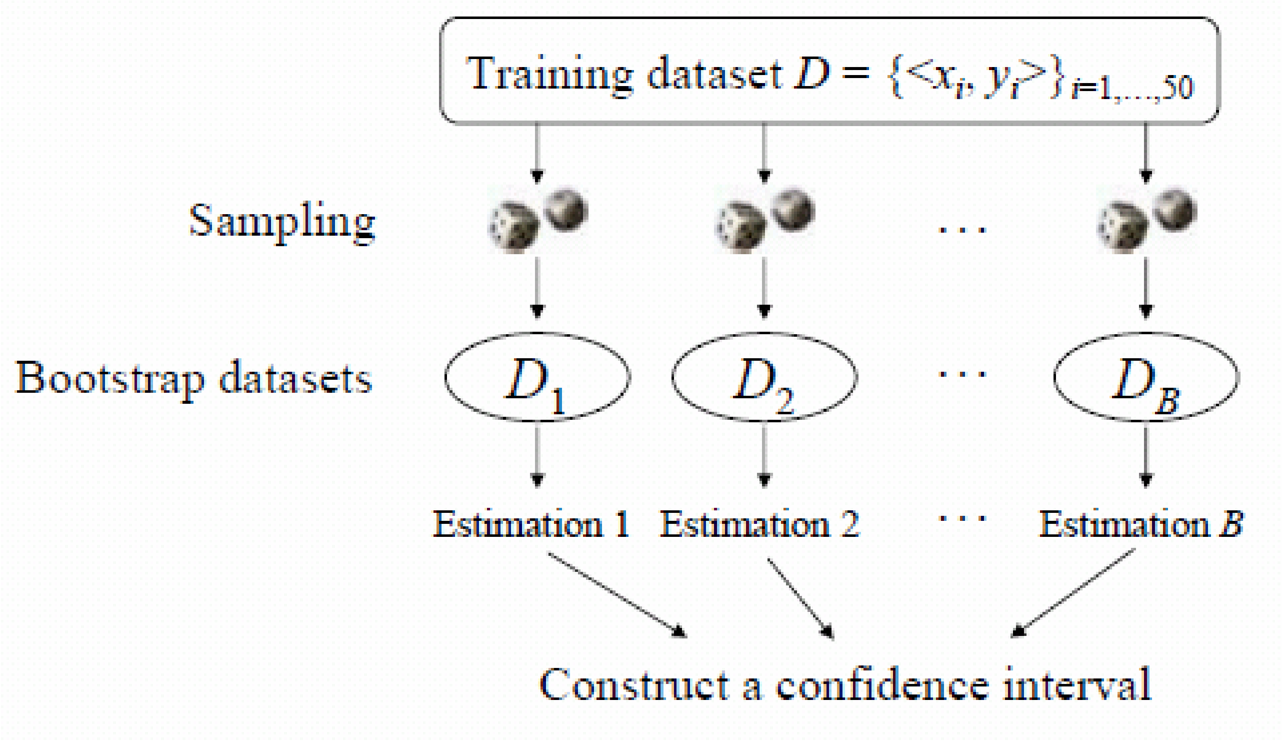
s = 0.002 per hour

p = 0.01 per hour

Estimate uncertainty given limited number of experimental observations

Bootstrap Methods

Estimating generalization error based on "resampling“:
Randomly draw datasets with replacement from training data



Training dataset $D = \{<x_i, y_i>\}_{i=1,\dots,50}$

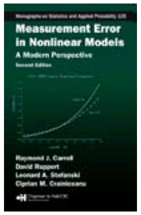
Sampling

Bootstrap datasets

D_1 D_2 . . . D_B

Estimation 1 Estimation 2 . . . Estimation B

Construct a confidence interval



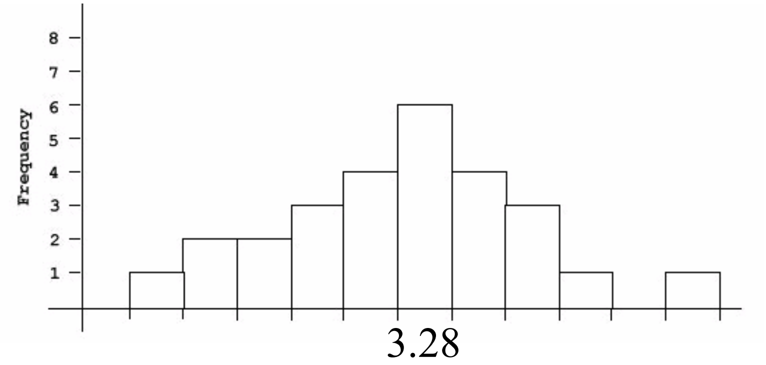
Pengyu Hong

Effect of generating bootstrap dataset from the distribution D is similar to the effect of obtaining dataset D={x_1, x_2, …, x_N} from the original distribution D’

Bootstrap Methods

Randomly draw datasets with replacement from training data

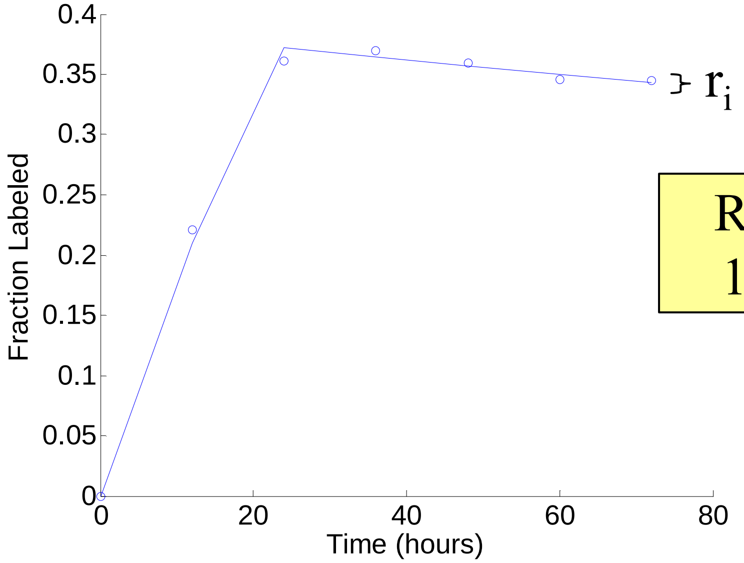
- D = [3.0, 2.8, 3.7, 3.4, 3.5] → average = 3.28
- Bootstrap samples D_N could be:
  - [2.8, 3.4, 3.7, 3.4, 3.5] → 3.36
  - [3.5, 3.0, 3.4, 2.8, 3.7] → 3.28
  - [3.5, 3.5, 3.4, 3.0, 2.8] → 3.24
  - ...



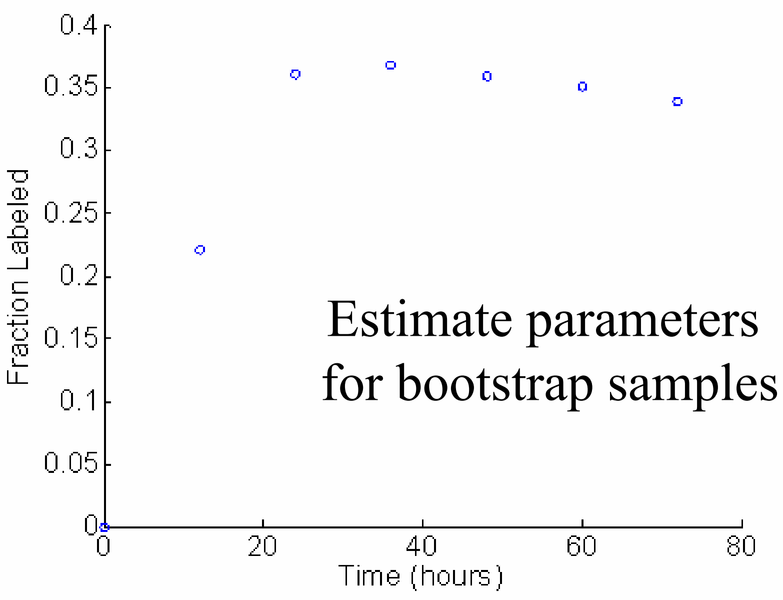
If sample is good approximation of population, bootstrap method will provide good approximation of sampling distribution of original statistic.

Bootstrapping Parameter Confidence Intervals

1) Fit model to data to obtain parameter estimates
2) Draw a bootstrap sample of the residuals (Fixed-X Bootstrapping)
3) Create bootstrap sample of observations by adding randomly sampled residual to predicted value of each observation



Repeat 1000x



Estimate parameters for bootstrap samples

Bootstrapping observations also possible – asymptotically equivalent

Bootstrapping Parameter Confidence Intervals

Three commonly used methods: 1. Normal Theory Intervals, 2. Percentile Intervals, 3. Bias Corrected Percentile Intervals

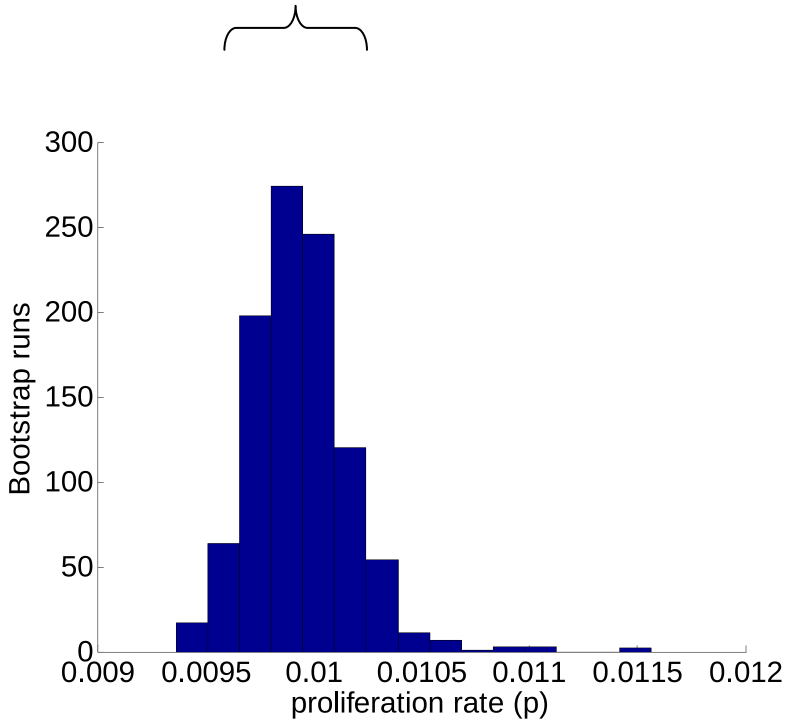
Percentile Intervals

Calculate the parameter for each bootstrap sample and select α (e.g., 0.05)

LCL = α /2th percentile.

UCL = (1-α/2)th percentile.

Use MATLAB's prctile function:
= prctile(bootstrap estimates, 0.025)


Contains 95% of the estimates

Parameter estimates for synthetic data

Estimate of s = 0.0017 [0.0009,0.0030]
Estimate of p = 0.0099 [0.0095,0.0100]

May not have correct coverage when sampling distribution skewed

Immune System **Adapts** to Pathogenic Challenge

Secondary responses are quantitatively and qualitatively different

Faster kinetics, greater magnitude

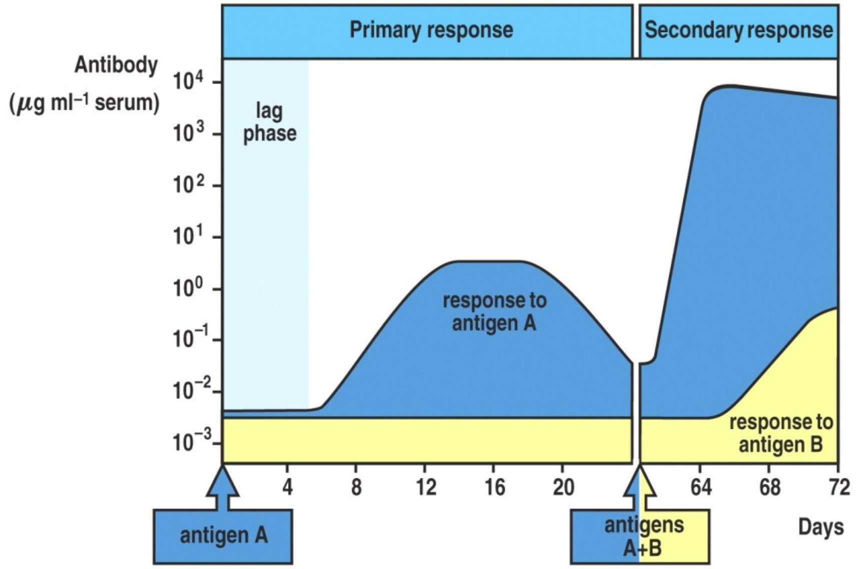


Figure 1-20 Immunobiology, 6/e. (© Garland Science 2005)

Increased affinity

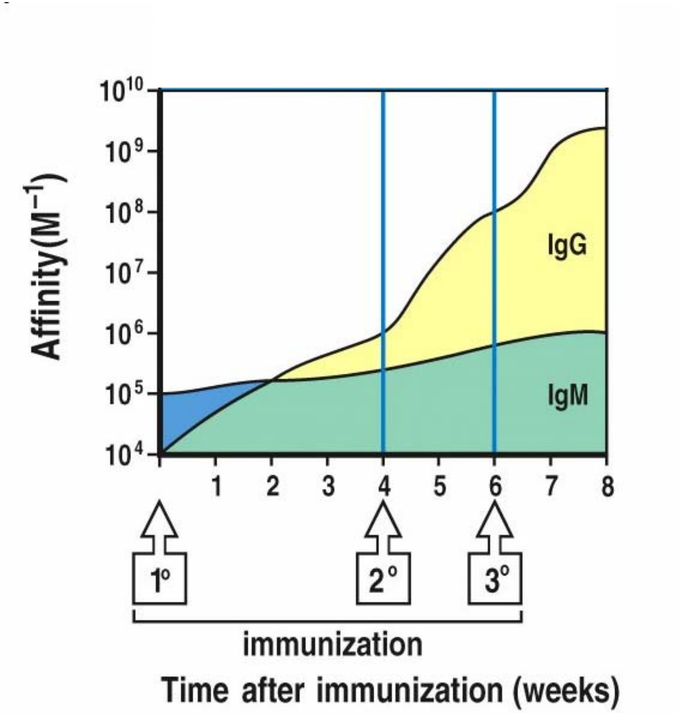
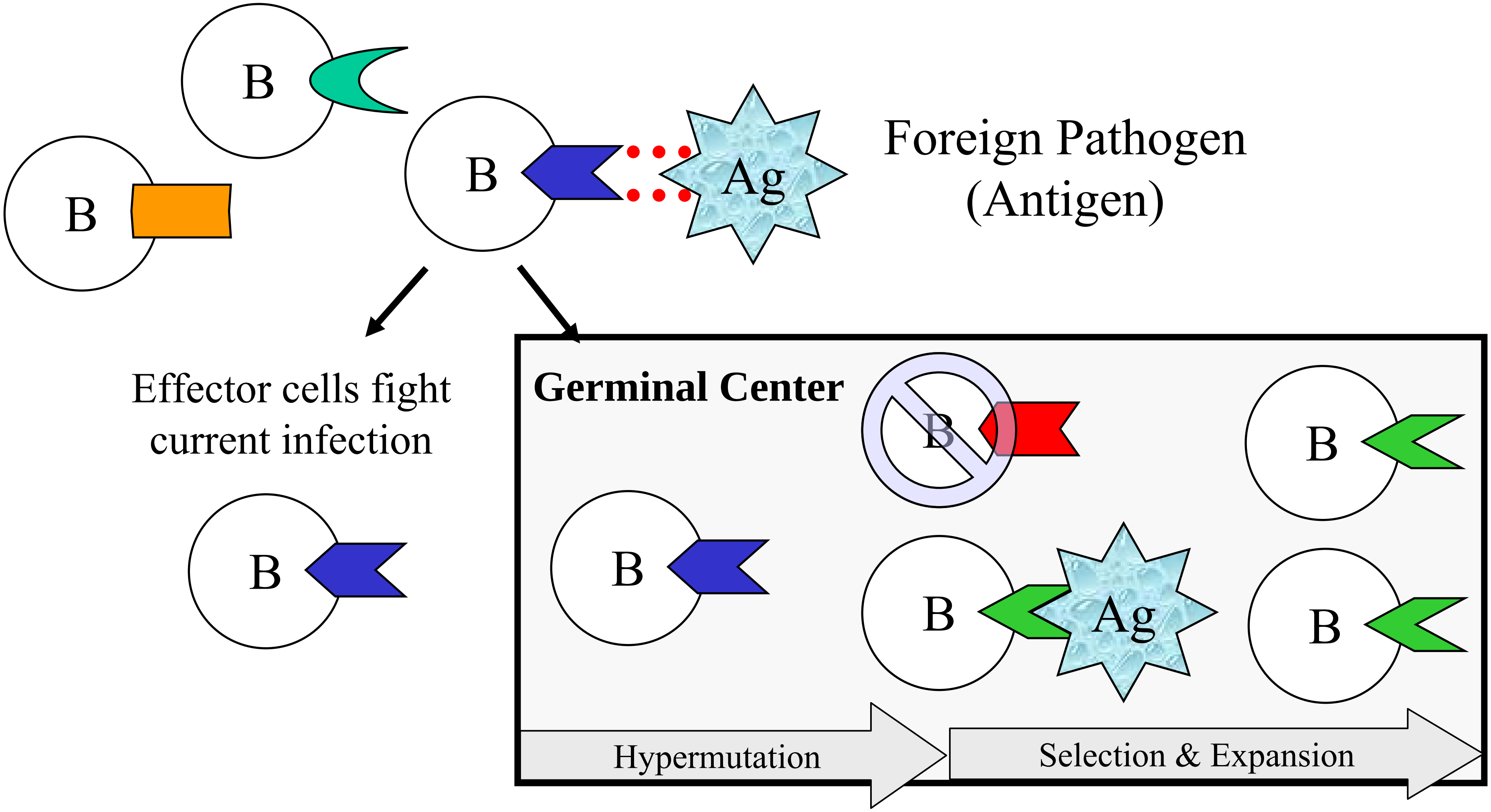


Figure 10-31 Immunobiology, 6/e. (© Garland Science 2005)

Affinity Maturation is Fundamental to Adaptive Immunity

Germinal Centers are Site of Affinity Maturation



Foreign Pathogen (Antigen)

Effector cells fight current infection

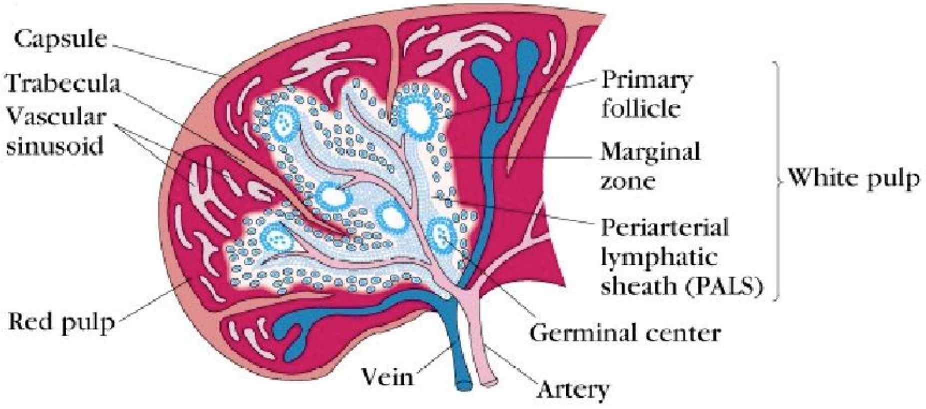
Germinal Center

Hypermutation

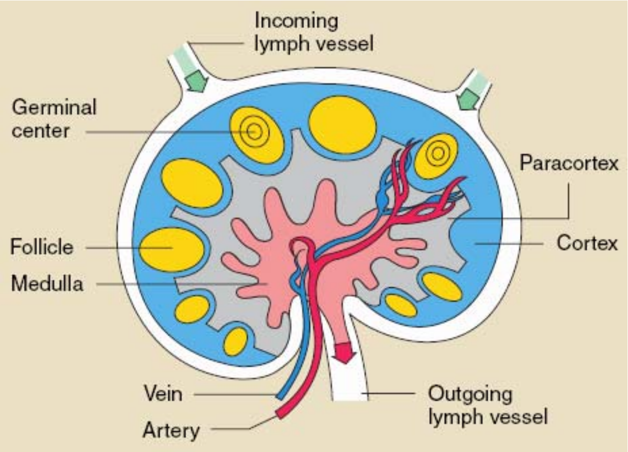
Selection & Expansion

Affinity maturation accomplished through somatic hypermutation of B cell receptor, followed by expansion of rare higher-affinity mutants

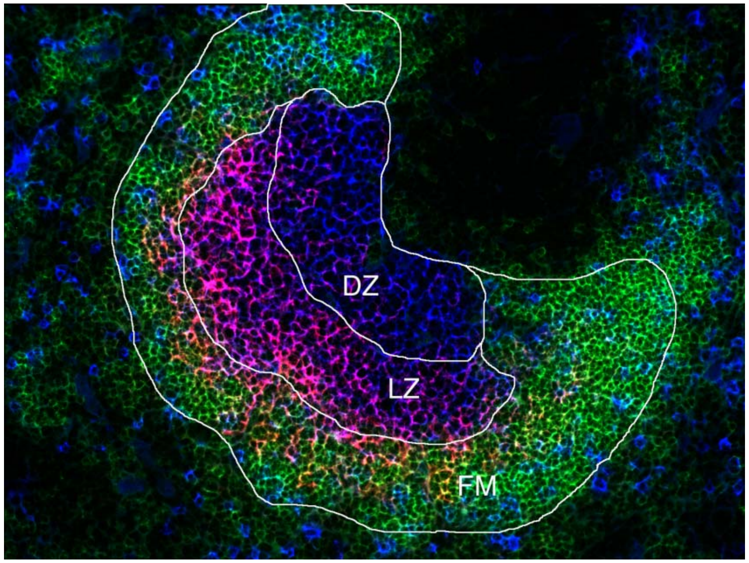
Germinal Centers in Spleen & Lymph Nodes



http://mcb.berkeley.edu/courses/mcb150/Lect10/Lect10.pdf

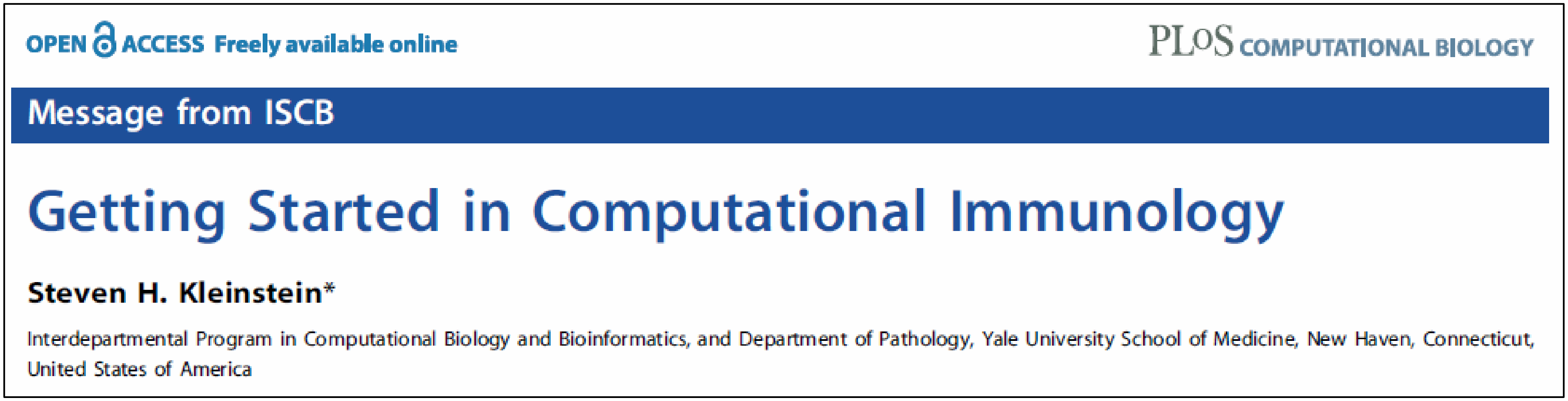


Germinal Center



Site of somatic hypermutation, and production of long-lived memory and plasma cells

For more information…



OPEN ACCESS Freely available online

PLoS COMPUTATIONAL BIOLOGY

Message from ISCB

Getting Started in Computational Immunology

Steven H. Kleinstein*

Interdepartmental Program in Computational Biology and Bioinformatics, and Department of Pathology, Yale University School of Medicine, New Haven, Connecticut, United States of America

Feel free to email me with any questions!
steven.kleinstein@yale.edu