Final Project

Dalton Flynn

4/27/16

The goal of this project is to determine the 90% confidence interval for the dimensionless diffusivity parameter K from the set of data given.

Contents

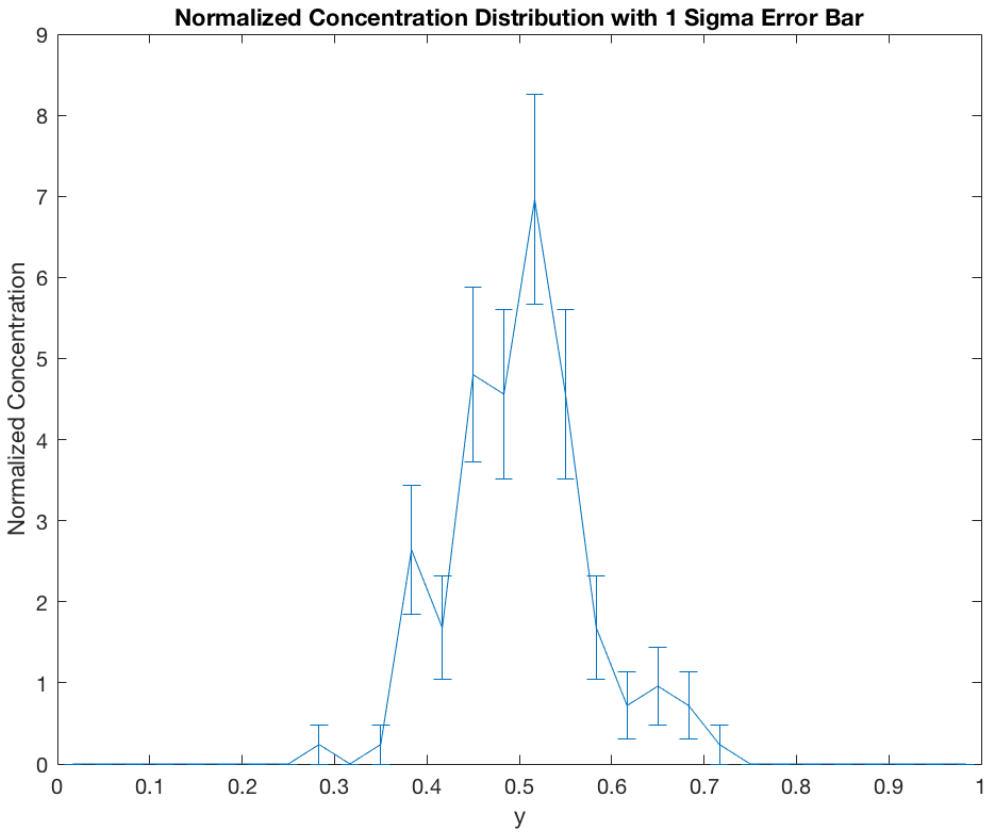
Part 1: Experimental Data

In this section the data is imported from an Excel spreadsheet using xlsread. Then, the data is centered about y = 0.5. Next, the data is split up into nbins of constant binsize from 0 to 1. The data in each bin is normalized by the total number of drops and the bin size, and the normalized concentration distribution and Poisson error is plotted up against the midpoint of the range in y of each bin.

```
clear;
clc;

y = xlsread('ydata.xlsx');

y = sort(y + (.5 - mean(y))); % ensures mean y value is 0.5 and
% sorts the data in increasing order

nbins = 30 % number of bins to place data points into
binsize = 1/nbins; % length of each bin as part of radius of length 1
ndrops = length(y); % number of data points

binconc = zeros(1,nbins); % initialize vector of bin concentrations
error = zeros(1,nbins); % intialize vector of bin errors
ndropsbin = zeros(1,nbins); % intitialize vector of number of drops in each
% bin

% place data into bins
for i = 1:nbins
    lowend = (i-1) * binsize; % low end parameter for ith bin
    highend = i * binsize; % high end parameter for ith bin
    keep = find(y <= highend & y>lowend); % find data within bin range
    ndropsbin(i) = length(keep);
    normalize = ndropsbin(i) / binsize / ndrops; % normalize
    binconc(i) = normalize;
    error(1,i) = sqrt(ndropsbin(i)) / binsize / ndrops;
    % from error propogation and knowledge of Poisson distribution
    % sigma = sqrt(N)
```

```
end

binedges = [0:binsize:1]; % to plot against
binMids = zeros(1,nbins); % initialize
% find midpoint in the range of each bin to plot up against concentration
for i = 1:nbins
    binMids(i) = (binedges(i)+binedges(i+1))/2;
end

% Plot
figure(1)
errorbar(binMids,binconc,error);
title('Normalized Concentration Distribution with 1 Sigma Error Bar')
xlabel('y')
ylabel('Normalized Concentration')
hold on
```

```
nbins =

    30
```

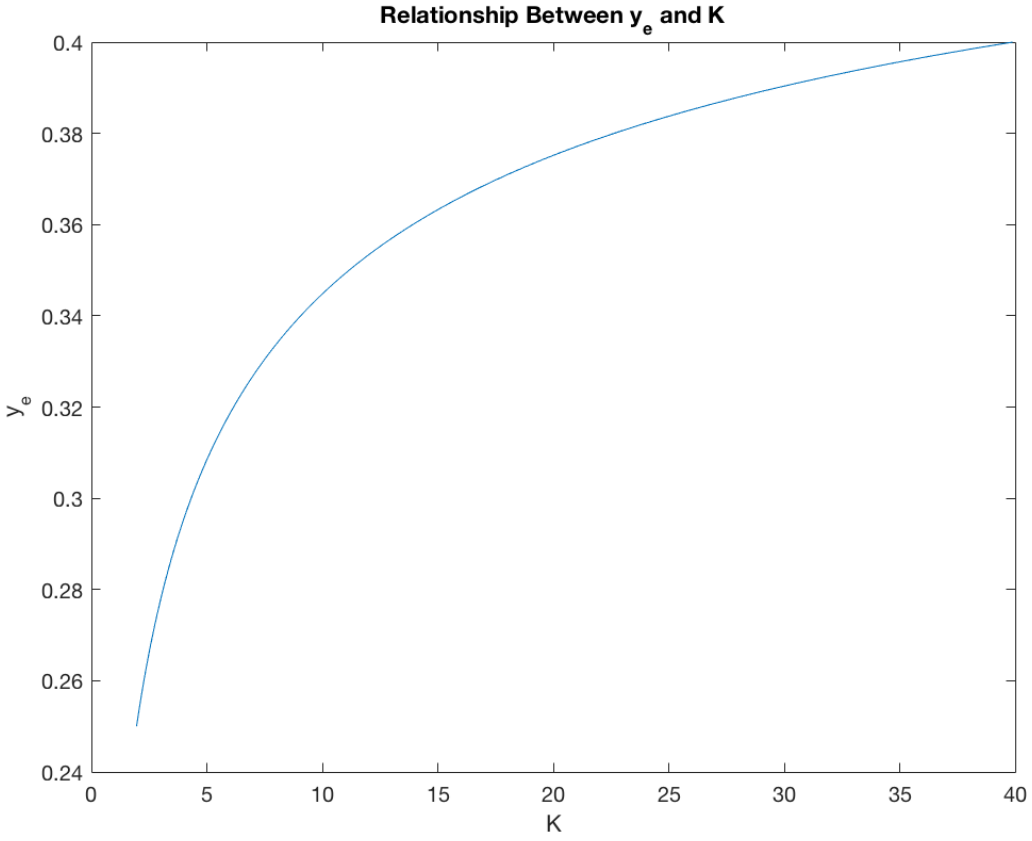


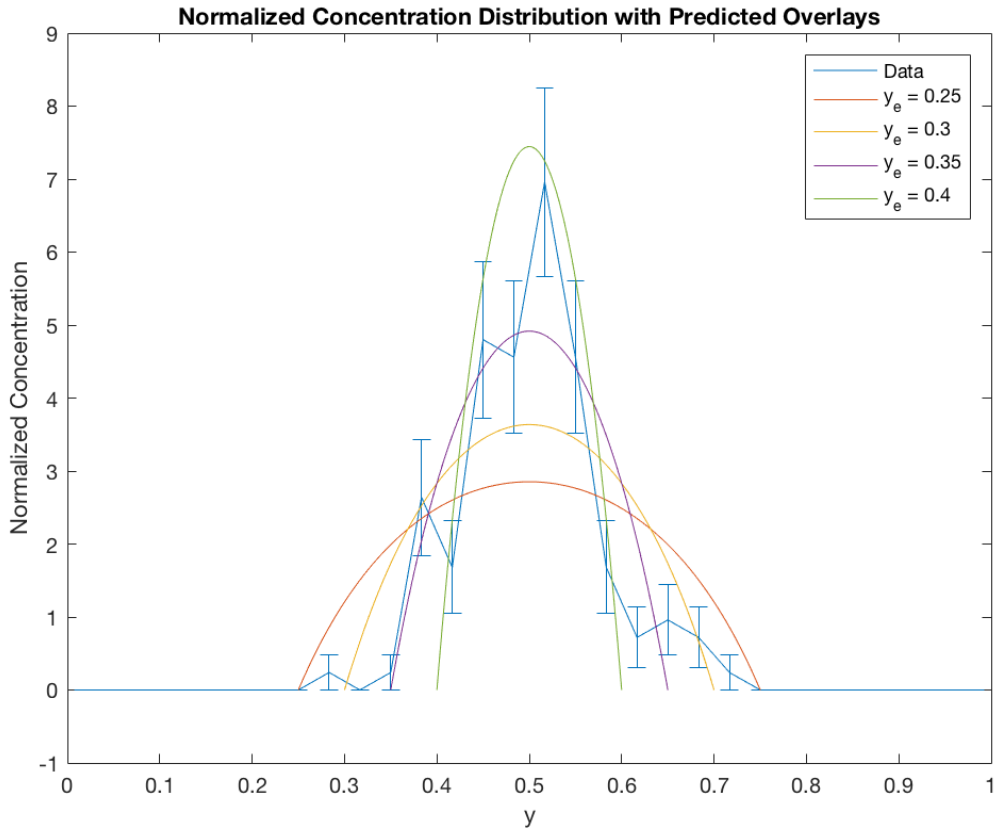
Part 2: Exploration of the Theoretical Distribution

By integrating d(phi)/dy and using an initial condition phi(ye) = 0 analytically, then integrating phi * dy = 1 for y = ye to y = 1-ye numerically, I get a relationship between K and ye. The first integration is done on paper and the relationship is plotted up here ye vs. K for values of ye which range from 0.25 to 0.4.

```
yevec = linspace(0.25,0.4,100); % vector to plot against K
for i = 1:length(yevec)
    K(i) = getK(yevec(i)); % calculates K by function getK, which is attached
end

figure(2)
plot(K,yevec)
xlabel('K')
ylabel('y_e')
title('Relationship Between y_e and K')

n = 100; % points to evaluate
phi = zeros(4,n); % initialize for 4 ye values
yval = zeros(4,n); % initialize for 4 ye values
j = 0; % counter to hold place in matrices yval and phi
for ye = [0.25 0.30 0.35 0.4]
    j = j + 1;
    yval(j,:) = linspace(ye,1-ye,n);
    phi(j,:) = getphi(yval(j,:),ye); % calculate phi using function getphi,
    % which is attached
end

figure(1)
plot(yval(1,:),phi(1,:),yval(2,:),phi(2,:),yval(3,:),phi(3,:),yval(4,:),phi(4,:))
title('Normalized Concentration Distribution with Predicted Overlays')
legend('Data', 'y_e = 0.25', 'y_e = 0.3', 'y_e = 0.35', 'y_e = 0.4')
hold off

iye = 0.36; % from eyeballing the overlay of predicted concentration
% distributions on the distribution given by the data
iK = getK(iye);

disp('Eyeballing the concentration gives me a best fit value of ');
disp([num2str(iye), ' for ye and therefore ', num2str(iK), ' for K.']);
```

```
Eyeballing the concentration gives me a best fit value of
0.36 for ye and therefore 13.9146 for K.
```



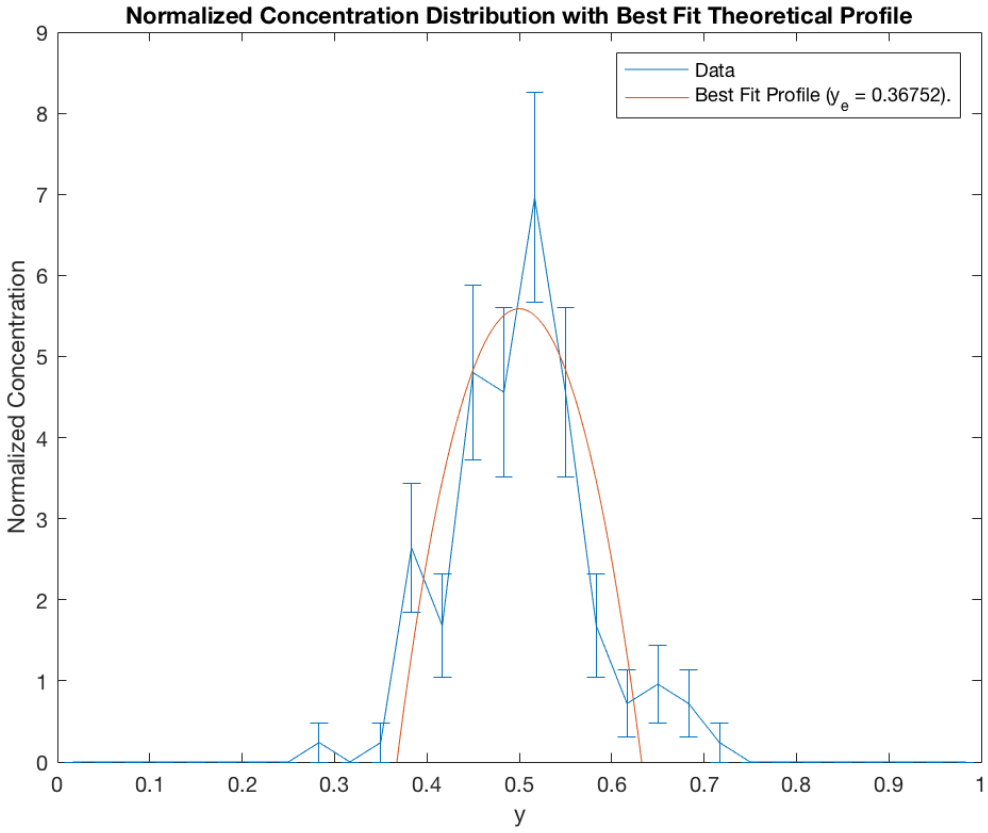


Part 3: Calculation of K from the Experimental Results

In this section, I use non-linear regression to determine the value of K which minimizes the residual between predicted and actual concentration distributions. I find the ye which minimizes the residual, then calculate the corresponding K. I then produced a new figure with the data (with errorbars) and the best fit theoretical concentration profile.

```
% to pass into minimization routine
global concpass nbinpass
concpass = binconc;
nbinpass = nbins;

guess = iye; % guess using the best fit from eyeballing in part 2
bfye = fminsearch('dphi',guess); % minimizes the function dphi, which is attached
bfK = getK(bfye);

disp(['Nonlinear regression gives me a best fit value of ',num2str(bfye)]);
disp(['for ye and therefore ', num2str(bfK), ' for K.']);

bfyval = linspace(bfye,1-bfye,n); % to plot new best fit phi
bfphi = getphi(bfyval,bfye); % calculate phi using function getphi

figure(3)
errorbar(binMids,binconc,error);
hold on
plot(bfyval,bfphi)
hold off
title('Normalized Concentration Distribution with Best Fit Theoretical Profile')
xlabel('y')
ylabel('Normalized Concentration')
legend('Data', ['Best Fit Profile (y_e = ', num2str(bfye), ').'])
```

```
Nonlinear regression gives me a best fit value of 0.36752
for ye and therefore 16.5792 for K.
```



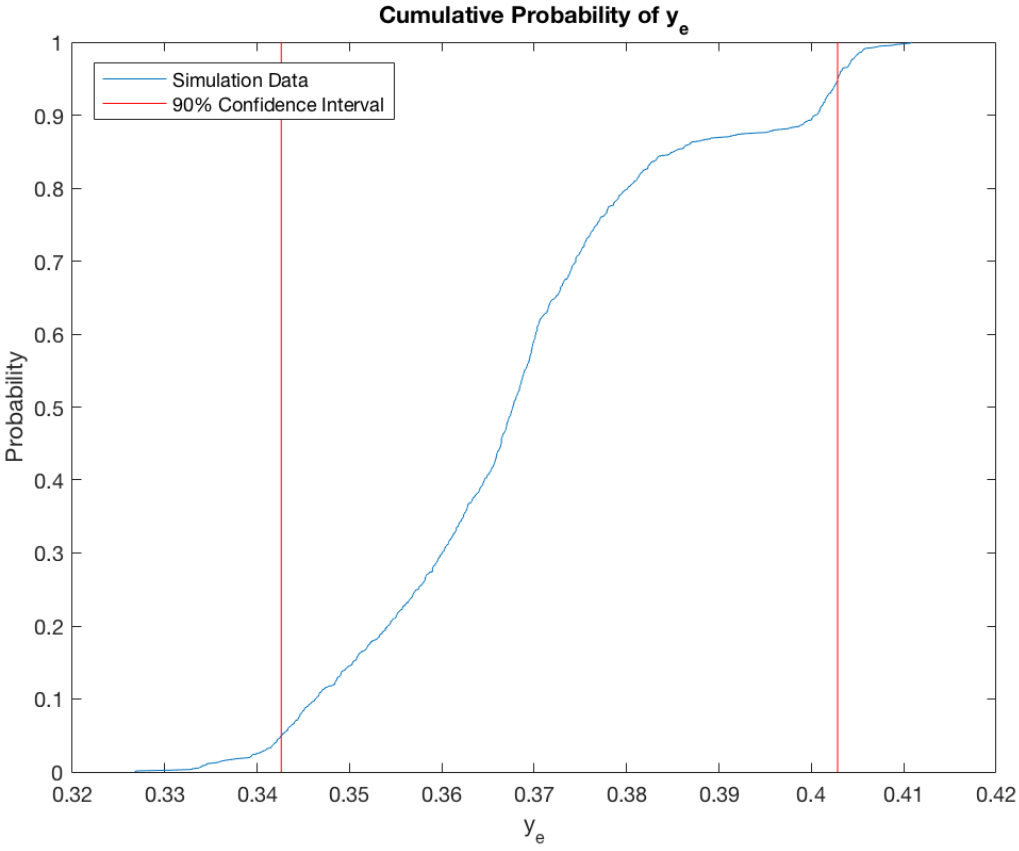
Part 4: Calculation of the Uncertainty in ye and K

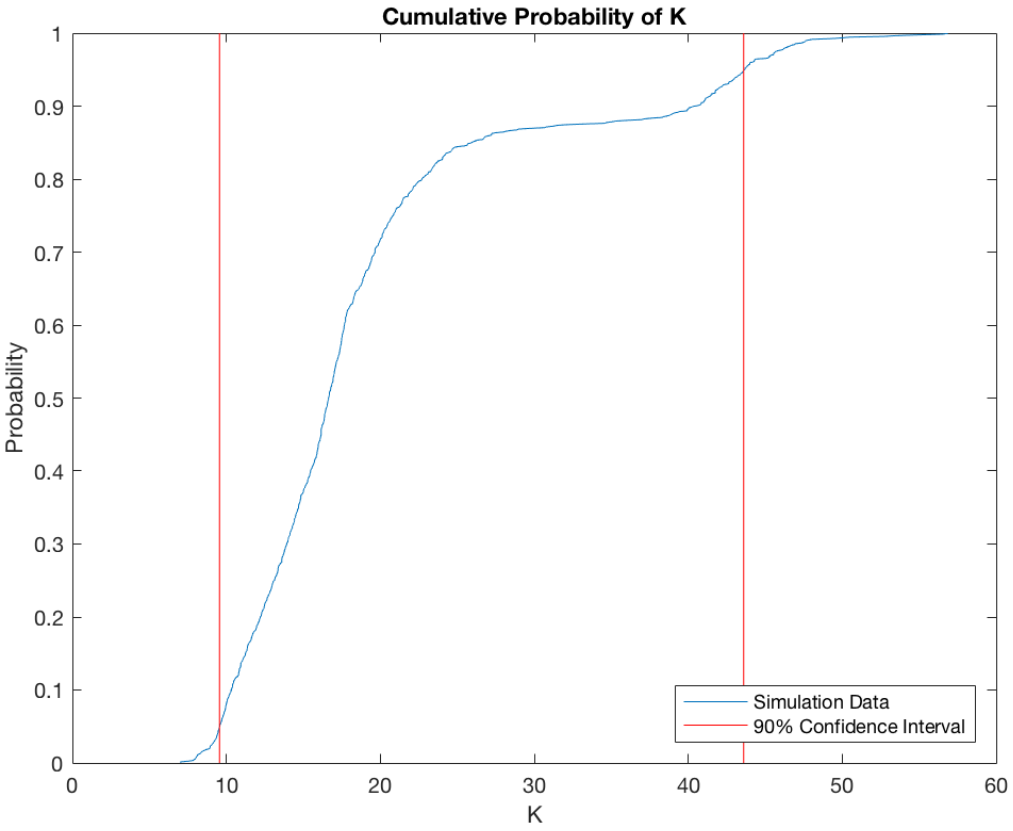
In this section the 90% confidence intervals in ye and K are determined. This is done by a Monte Carlo simulation which simulates and modifies the number of particles in each bin according to the Poisson distribution and recalculates ye and K. This simulation generates a distribution of the fitted values of ye and K, so that the cumulative probability of each can be plotted and the 90% confidence interval indicated on the graph.

```
psummcsq=zeros(2,2); % Initialize array to sum products of MC parameters
psummc=zeros(1,2); % Initialize vector to sum MC parameters
nmc=1000; % number of iterations to run through
yemc=zeros(1,nmc); % initialize vector of ye values to be calculated in the
% MC simulation
Kmc=zeros(1,nmc); % initialize vector of L values to be calculated in the
% MC simulation

for i=1:nmc
    ndropsimbin=random('poisson',ndropsbin); % create a distribution of
    % bins of 'data' points based on the poisson distribution
    ndropsim = sum(ndropsimbin); % total number of drops in simulation
    concpass = ndropsimbin / ndropsim / binsize; % normalize concentration
    parammc(1)=fminsearch('dphi',bfye); % recalculate ye as parameter 1
    yemc(i) = parammc(1); % add to yemc vector
    parammc(2)=getK(parammc(1)); % calculate K as parameter 2
    Kmc(i) = parammc(2); % add to Kmc vector
    psummc=psummc+parammc; % add to sum
    psummcsq=psummcsq+parammc'*parammc; % add to sum of products
end
% Now I calculate the mean and matrix of covariance of these values:
meanparammc=psummc/nmc;
```

```
% The covariance:
varparammc=(psummcsq-nmc*meanparammc'*meanparammc)/(nmc-1);
sigYe = sqrt(varparammc(1,1)); % standard deviation in ye
sigK = sqrt(varparammc(2,2)); % standard deviation in K

% 90% Confidence Interval of ye
x = sort(yemc); % order the yemc
n = length(yemc);
cumprobye=([1:n]-.5)/n; % calculate cumulative probablity
conflowye = x(50); % 50th point is at roughly 5%
confhighye = x(950); % 950th point is at roughly 95%
confintye = [conflowye,confhighye];
clowyeplot = [conflowye, conflowye];
chighyeplot = [confhighye, confhighye]; % for plotting conf int line
disp(['The 90% confidence interval for ye is ', num2str(confintye(1)),...
    ' to ', num2str(confintye(2)),' when nbins = ',num2str(nbins), '.'])

% 90% Confidence Interval of K
x2 = sort(Kmc); % order the Kmc
n = length(Kmc);
cumprobK=([1:n]-.5)/n; % calculate cumulative probability
conflowK = x2(50); % 50th point is at roughly 5%
confhighK = x2(950); % 950th point is at roughly 95%
confintK = [conflowK,confhighK];
clowKplot = [conflowK, conflowK];
chighKplot = [confhighK, confhighK]; % for plotting conf int line
disp(['The 90% confidence interval for K is ', num2str(confintK(1)),...
    ' to ', num2str(confintK(2)),' when nbins = ',num2str(nbins), '.'])

cintplot = [0, 1]; % to plot conf int vertical lines

% plot cumulative probability of ye
figure(4)
plot(x,cumprobye,clowyeplot,cintplot,'r',chighyeplot,cintplot,'r')
title('Cumulative Probability of y_e')
xlabel('y_e')
ylabel('Probability')
legend('Simulation Data','90% Confidence Interval','Location','NorthWest')

% plot cumulative probability of K
figure(5)
plot(x2,cumprobK,clowKplot,cintplot,'r',chighKplot,cintplot,'r')
title('Cumulative Probability of K')
xlabel('K')
ylabel('Probability')
legend('Simulation Data','90% Confidence Interval','Location','SouthEast')
```

```
The 90% confidence interval for ye is 0.34262 to 0.40284 when nbins = 30.
The 90% confidence interval for K is 9.5584 to 43.5985 when nbins = 30.
```





Part 5: Conclusions

Was the drop distribution well described by the model, or were there systematic deviations that might have affected your fitted values?

The drop distribution was not very well described by the model for nbins= 30. This is due to the jaggedness of the normalized concentration data. The theoretical model predicts a smooth parabola, which can't be fit easily to a jagged curve. However, the fit improves for nbins = 29, 31, and 33 and generally for odd number bins because a bin is centered at y = 0.5 with the maximum concentration. Centering the maximum concentration aids in developing a smoother shape of the normalized concentration data. Additionally, the tails of the data are more populated than the model predicts - the actual concentration distribution shows some low concentration on the edges where the model predicts 0 concentration.

Were ye and K well described by a normal distribution?

As can be seen in the graphs below, for nbins = 30, small values of ye are well approximated by the normal distribution but large values are not. No values of K are well approximated by the normal distribution. However, for nbins = 29, 31, 33 (and again generally for an odd number of bins), both ye and K are well approximated by the normal distribution. This can be seen in the figures below for nbins = 33.

Is there any systematic variation in the behavior?

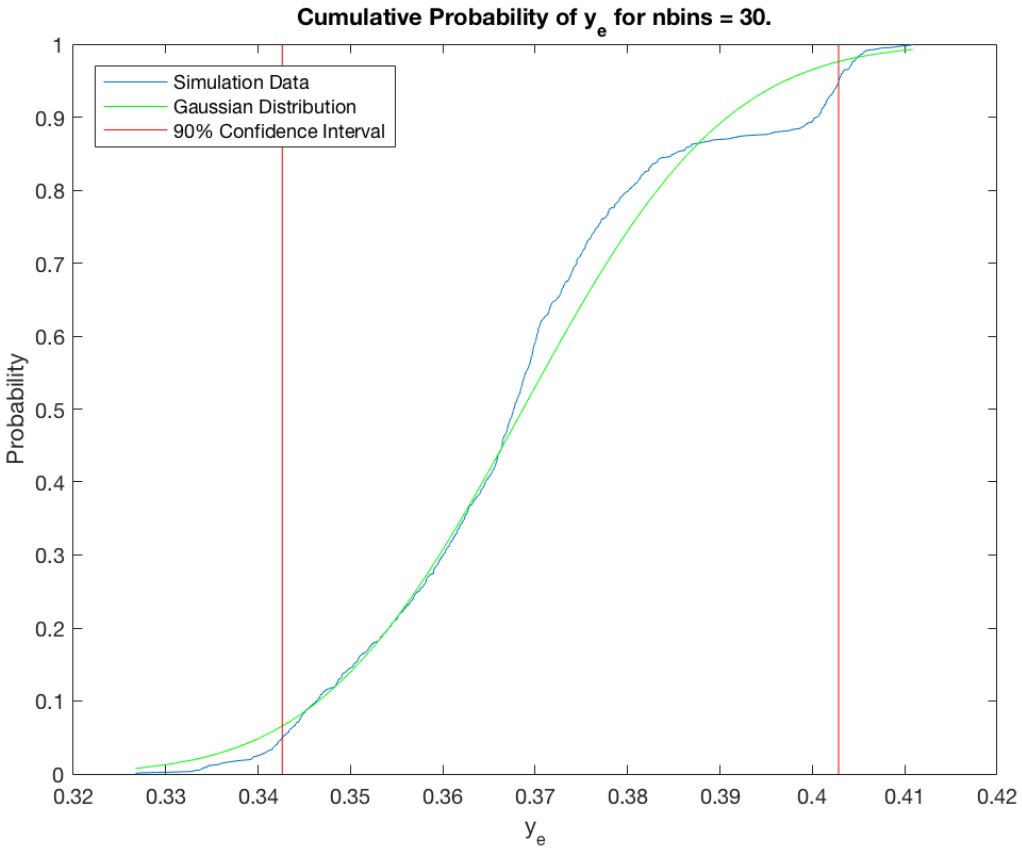
As described above, odd numbers around 30 for nbins systematically better approximate the actual data than even numbers. Above nbins = 80 or so, the actual distribution has increasingly large error and the distribution becomes jagged, making it more difficult to fit the smooth model to. For nbins = 1000, the error bars become huge. You can see this in the last figure below.

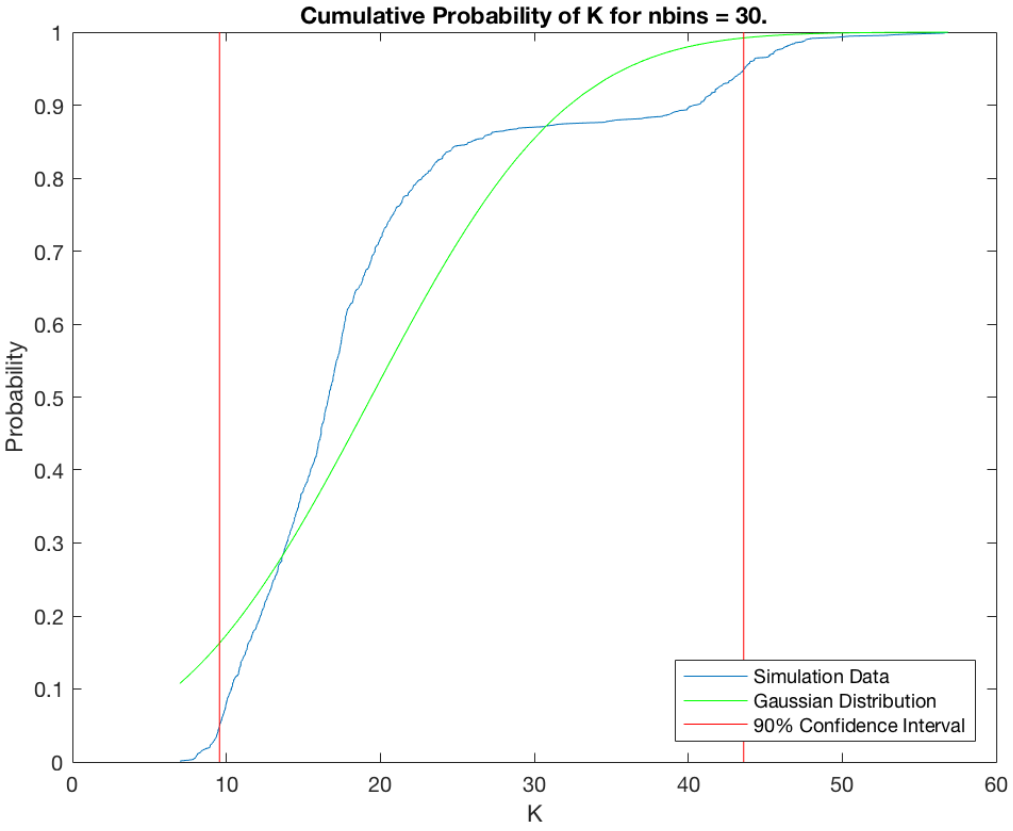
Based on all your calculations, what is your "best guess" as to the correct value of K and its confidence interval?

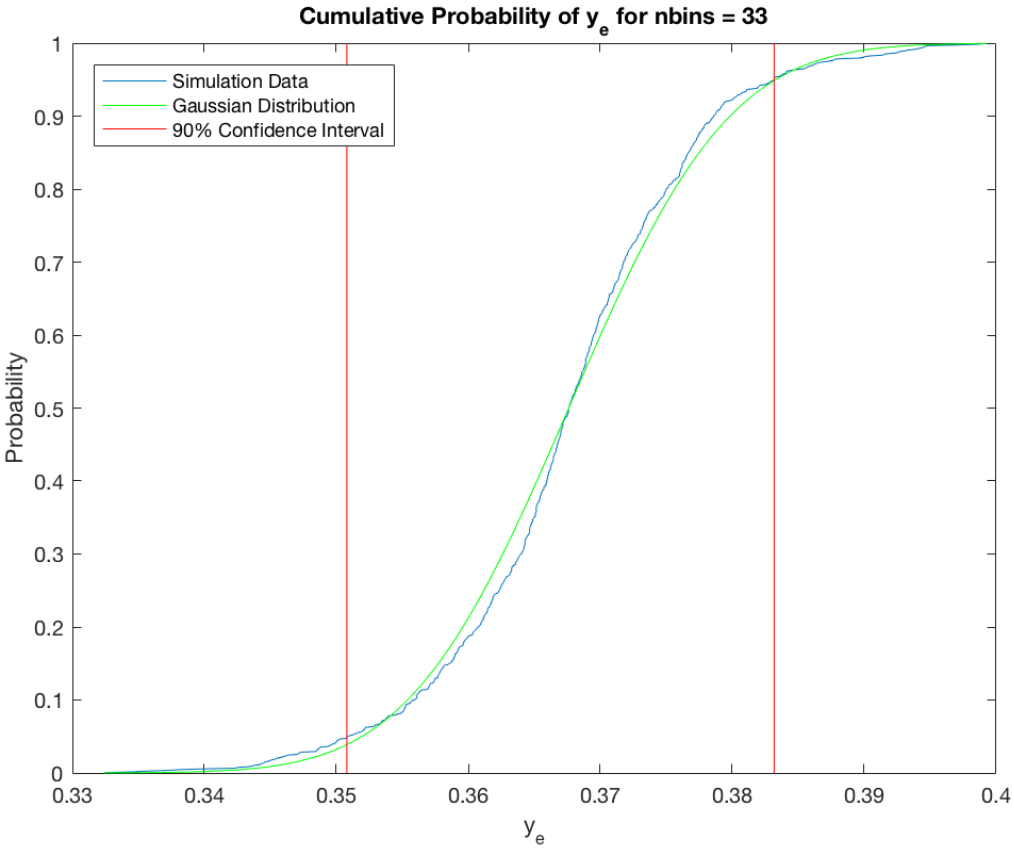
My best guess for K is the Monte Carlo mean K for nbins = 33, which is about k = 17.25. The 90% confidence interval is roughly [11.2141, 25.0384]. I chose this number of bins because the model fits the data nicely (see the figure below) and the confidence interval is tight. I use the Monte Carlo mean value because the Monte Carlo simulation simulates 1000 data sets and I assume the Poisson distribution very accurately depicts the data.

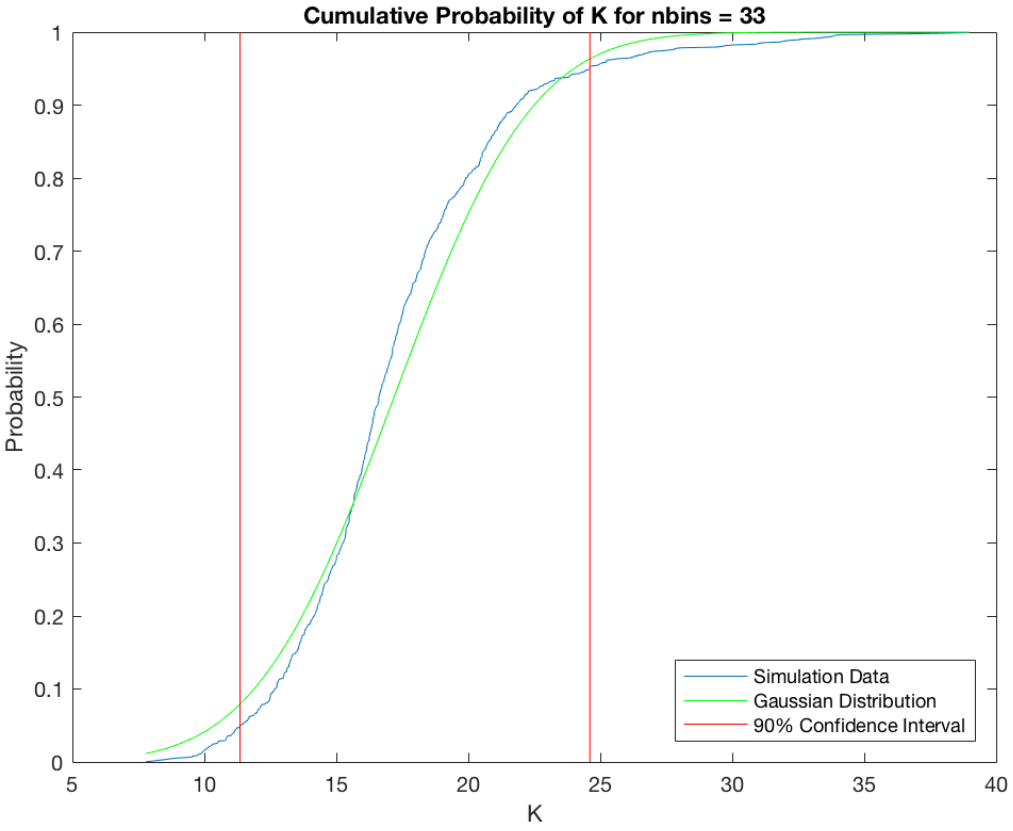
```
% Gaussian ye
Pgauss = 0.5 + erf((x-meanparammc(1))/sigYe/sqrt(2))/2;
figure(6)
plot(x,cumprobye,x,Pgauss,'g',clowyeplot,cintplot,'r',chighyeplot,cintplot,'r')
title(['Cumulative Probability of y_e for nbins = ', num2str(nbins),'.'])
xlabel('y_e')
ylabel('Probability')
legend('Simulation Data','Gaussian Distribution','90% Confidence Interval',...
    'Location','NorthWest')

% Gaussian K
PgaussK = 0.5 + erf((x2-meanparammc(2))/sigK/sqrt(2))/2;
figure(7)
plot(x2,cumprobK,x2,PgaussK,'g',clowKplot,cintplot,'r',chighKplot,cintplot,'r')
title(['Cumulative Probability of K for nbins = ', num2str(nbins),'.'])
xlabel('K')
ylabel('Probability')
legend('Simulation Data','Gaussian Distribution','90% Confidence Interval',...
    'Location','SouthEast')

% the following figures were saved after running this code with different
% numbers of bins and are loaded in here for comparison
openfig('yecumprob33.fig','new');
```

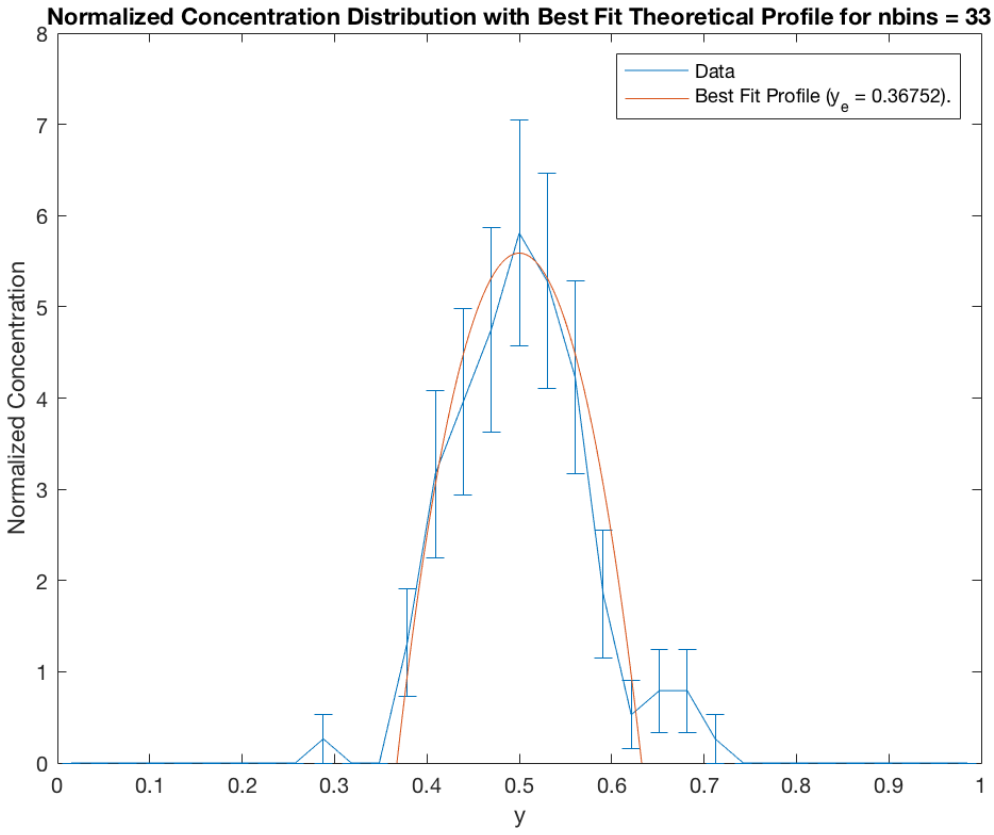
```
openfig('kcumprob33.fig','new');

openfig('bfprofile33.fig','new');

openfig('bfprofile1000','new');
```

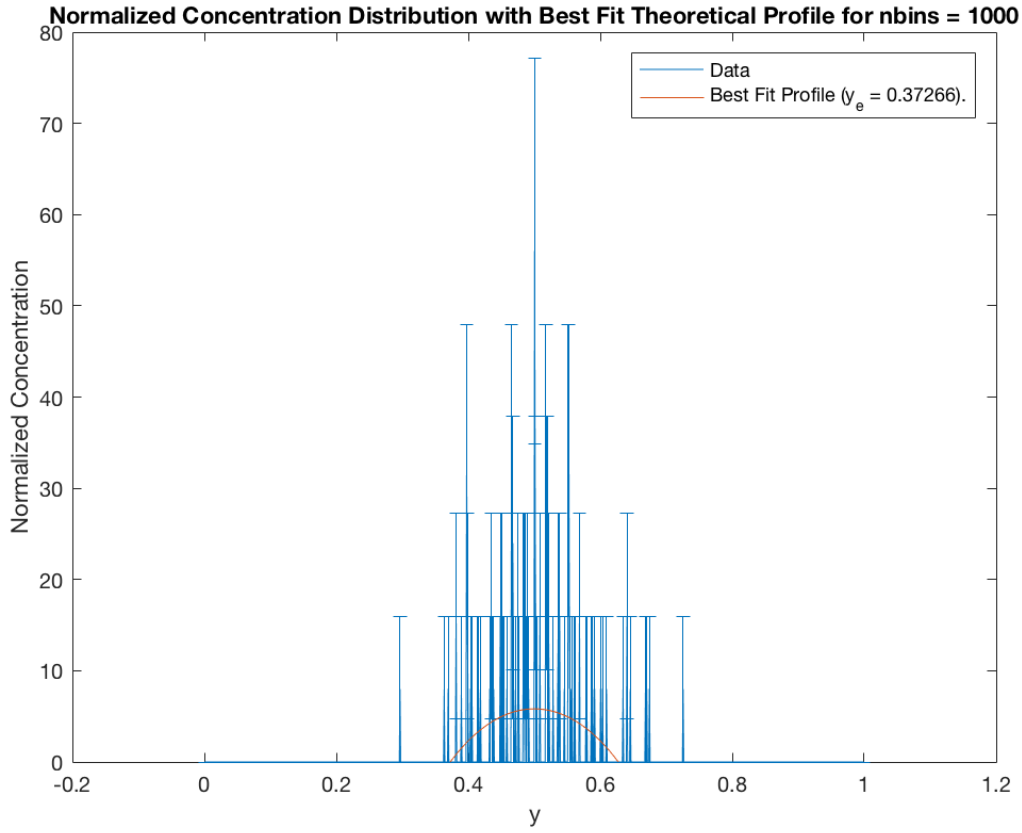



Cumulative Probability of K for nbins = 30.
Probability
K
Simulation Data
Gaussian Distribution
90% Confidence Interval
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
10
20
30
40
50
60


Cumulative Probability of y_e for nbins = 33
Probability
y_e
Simulation Data
Gaussian Distribution
90% Confidence Interval
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.33
0.34
0.35
0.36
0.37
0.38
0.39
0.4


Cumulative Probability of K for nbins = 33
Probability
K
Simulation Data
Gaussian Distribution
90% Confidence Interval


Normalized Concentration Distribution with Best Fit Theoretical Profile for nbins = 33
Normalized Concentration
y
Data
Best Fit Profile (y_e = 0.36752).



Published with MATLAB® R2016a