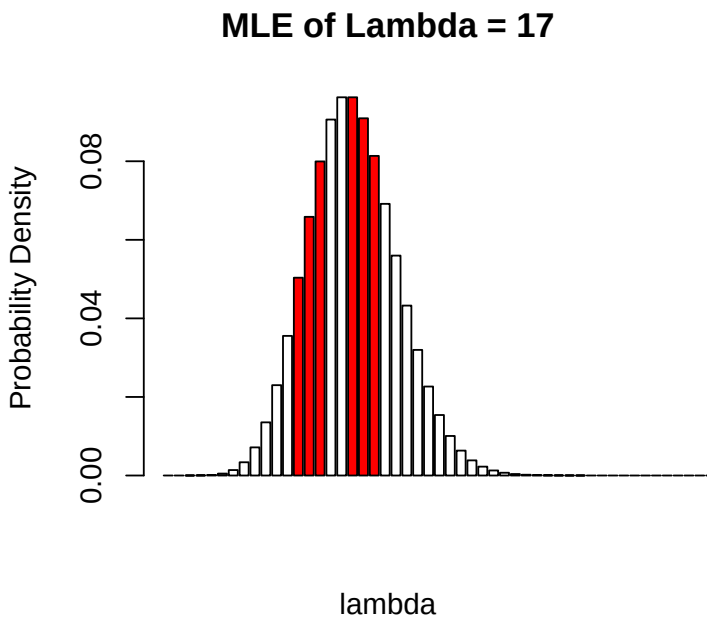
Fitting & Evaluating Likelihood Models

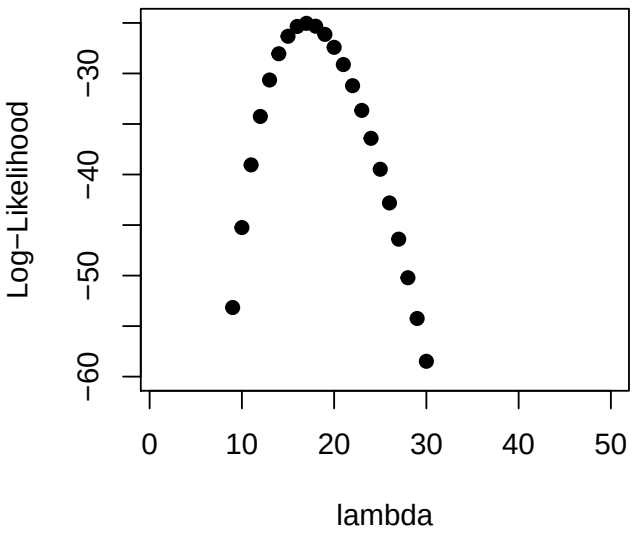
We Have the Maximum Likelihood Estimate of Lambda



p(a and b) = p(a)p(b)

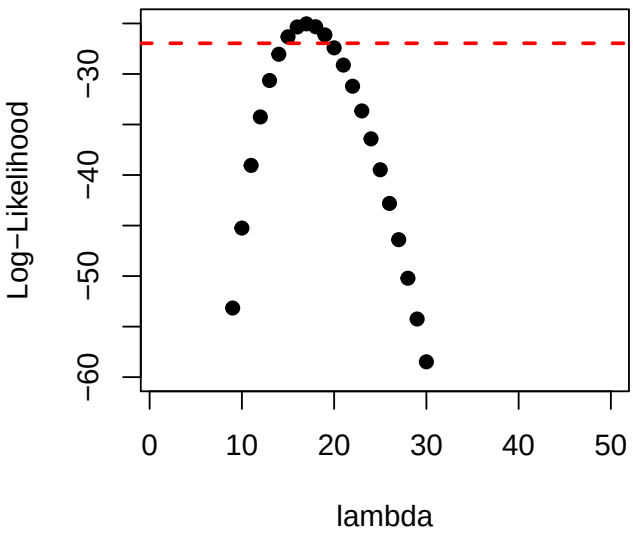
$$p(D|\theta) = \prod_{i=1}^{n} p(d_i|\theta)$$

What is the Variation Around our Estimate



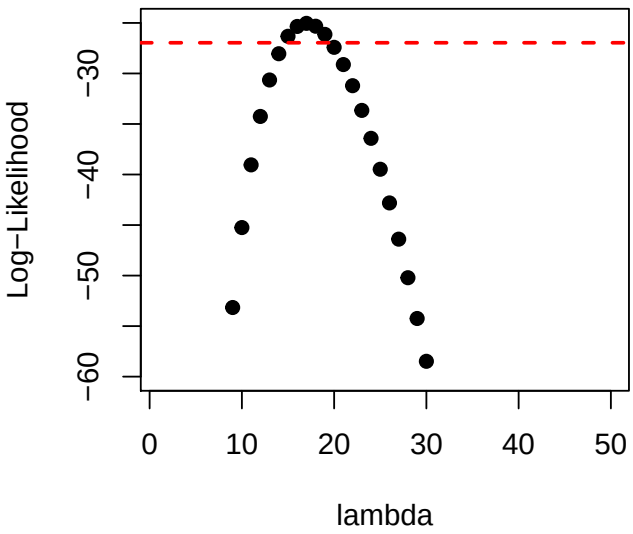
1. Log-Likelihood appxomiately χ^2 distirbuted
2. 95% CI holds all values within half of the .05 tail of $\chi^2_{df=1}$
3. (≈ 1.92)

Profile Likelihood CIs



1. Log-Likelihood appxomiately χ^2 distirbuted
2. 95% CI holds all values within half of the .05 tail of $\chi^2_{df=1}$
3. (≈ 1.92)

Profile Likelihood CIs




```
lConf <- max(ll) - 1.92
lConf

# [1] -26.96

#Get which values of lambda  that have LL >=
confVals <- lambdaVals[which(ll >= lConf)]

confVals[c(1, length(confVals))]

# [1] 15 19
```

How do we Compare Alternate Hypotheses?

$$G = 2ln(\frac{L_A}{L_0})$$

where L0 is from the more constrained hypothesis.
G is χ^2 distributed with DF = Difference in # Parameters

$$G = 2(LogL_A - LogL_0)$$

How do we Compare Alternate Hypotheses?

```
#compare our estimated fit to the hypothesis that lambda = 12
G12 <- 2*(ll[17] - ll[12])
pchisq(G12, 1, lower.tail=F)

# [1] 1.768e-05

#compare our estimated fit to the hypothesis that lambda = 15
G15 <- 2*(ll[17] - ll[15])
pchisq(G15, 1, lower.tail=F)

# [1] 0.1099
```

Exercise: Likelihood and Wolf Inbreeding!

- Load the Wolf Data
- Model Wolf Inbreeding Coefficient as a normal distribution with SD=0.1
- What is the ML estimate of the coefficient
- What is the 95% CI?
- Is the mean different from 0.3?

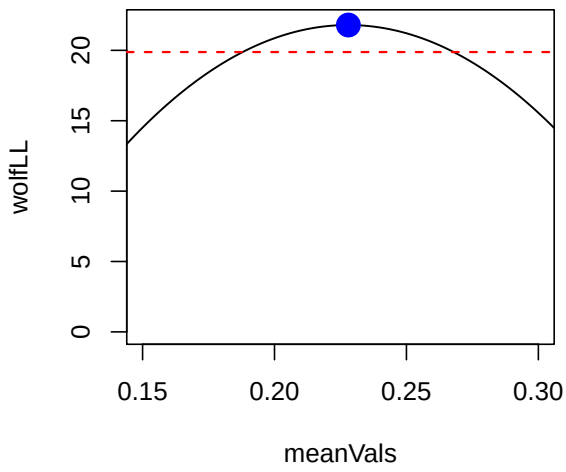
ML Estimate of Wolf Inbreeding Mean

```r
#load the wolf data
wolves <- read.csv("./data/16e2InbreedingWolves.csv")

meanVals <- seq(0, 0.5, .001)

wolfLL <- sapply(meanVals, function(x)
                sum(dnorm(wolves$inbreeding.coefficient,
                          mean=x, sd=0.1, log=TRUE)))
```

ML Estimate of Wolf Inbreeding Mean

```r
maxIDX <- which(wolfLL== max(wolfLL))

#the LL estimate of the mean
meanVals[maxIDX]

# [1] 0.228

#comparison to mean estimate
mean(wolves$inbreeding.coefficient)

# [1] 0.2279
```

ML Estimate of Wolf Inbreeding Mean



abline(h=x) for horizontal lines, use v for vertical lines

ML Estimate of 95% CI Of Inbreeding

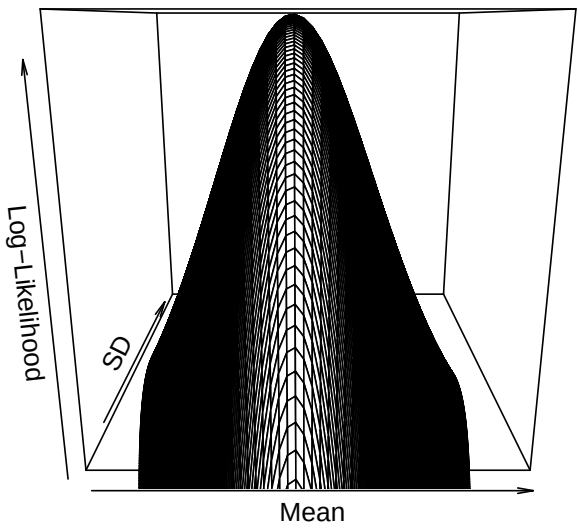
```
ciIDX <- which(abs(wolfLL-wolfLL[maxIDX]) <= 1.92)

#lowerCI
meanVals[ciIDX[1]]

# [1] 0.188

#upperCI
meanVals[ciIDX[length(ciIDX)]]

# [1] 0.267
```

Testing a Hypothesis of Means

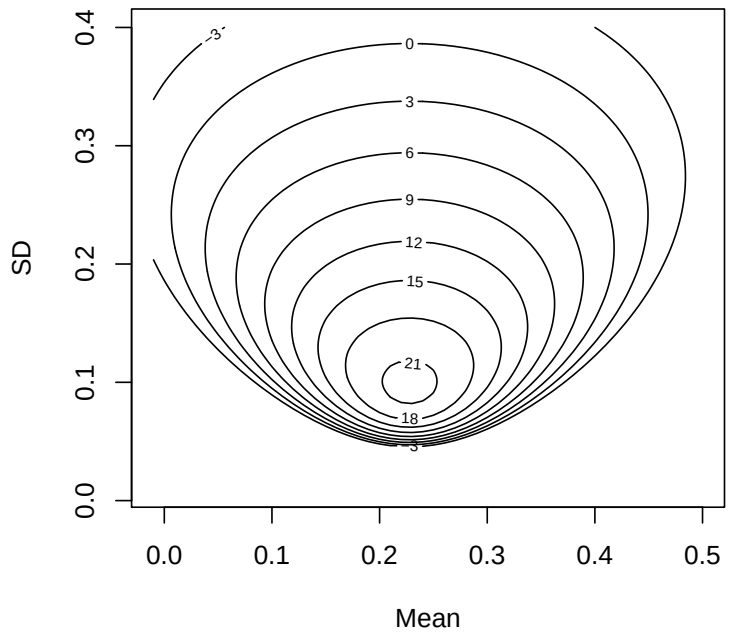
```
#Compare to Mean = 0.3
idx3 <- which(meanVals==0.3)

#calculate the Chisq LR
G <- 2*(wolfLL[maxIDX] - wolfLL[idx3])

#put it to the test
pchisq(G, 1, lower.tail=F)

# [1] 0.0004135
```

What if you have multiple parmeters?

Estimating Mean and SD: Likelihood Surface



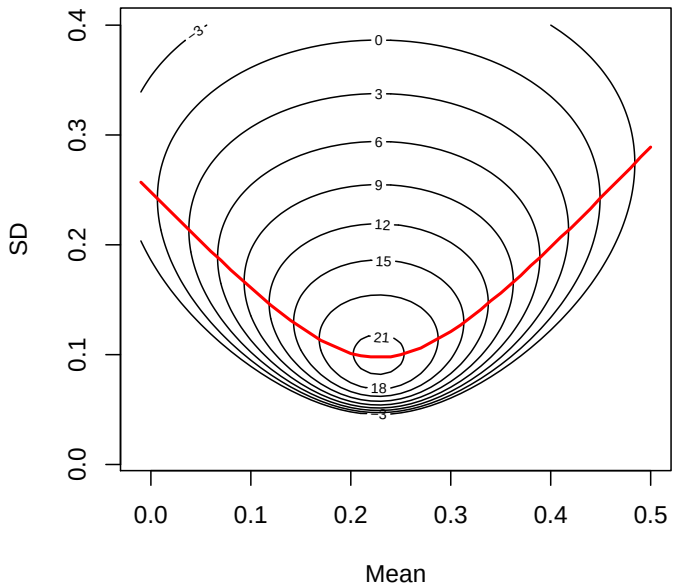
Contour Plot of Likelihood Surface



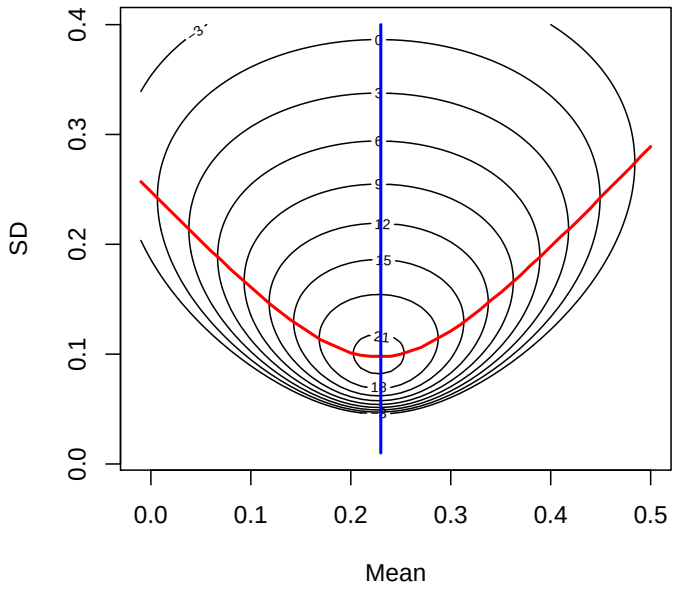
New Issues with Multiple Parameters

1. What Log-Likelihood Values Are Used for 95% CI?
2. Brute-Force Becomes Slow
3. Algorithmic Solutions Necessary
4. Specification Unwieldy

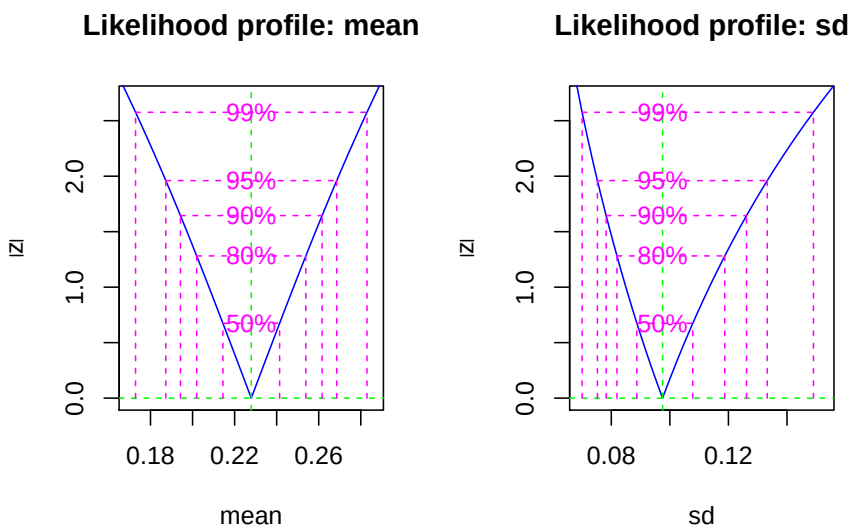
Likelihood Profile of One Coefficient Along ML Estimates of the other



Or we Use the Curve for a Single Parameter if Independet of the Other



Likelihood Profiles to get CIs



How do we Search Likelihood Space?

Optimizing to find a Minimum Value

- optim
- nlm
- nlminb
- mle2 (wrapper for all of the above)

Did You Say Minimum?

YES!

We optimize using -sum(LL Function)

Deviance = -2 * LL

How do we Search Likelihood Space?

There are many Algorithms

- Newtown-Raphson (algorithmicly implemented in `nlm` and `BFGS` method) uses derivatives
  - good for smooth surfaces & good start values
- Brent's Method - for single parameter fits
- Nelder-Mead Simplex (`optim`'s default)
  - good for rougher surfaces, but slower
- Simulated Annealing (`SANN`) uses Metropolis Algorithm search
  - global solution, but slow

Warning: If your algorithm fails to converge, you cannot evaluate your model or coefficients

Fitting Multiple Parameters with MLE2 from bbmle

```
#first write a function that you want to minimize
wolfLL <- function(m, s) -sum(dnorm(inbreeding.coefficient,
                                    mean=m, sd=s,
                                    log=TRUE))

#now feed the function to an optimizer
wolfLL_Fit <- mle2(wolfLL, data=wolves, start=list(m=0.2, s=0.2))
```

Watch for Warnings

```
#first write a function that you want to minimize
wolfLL <- function(m, s) -sum(dnorm(inbreeding.coefficient,
                                    mean=m, sd=s,
                                    log=TRUE))

#now feed the function to an optimizer
wolfLL_Fit <- mle2(wolfLL, data=wolves, start=list(m=0.2, s=0.2))

# Warning:  NaNs produced
# Warning:  NaNs produced
# Warning:  NaNs produced
# Warning:  NaNs produced
# Warning:  NaNs produced
# Warning:  NaNs produced
# Warning:  NaNs produced
# Warning:  NaNs produced
# Warning:  NaNs produced
# Warning:  NaNs produced
# Warning:  NaNs produced
```

Fix Functions to Behave Sensibly

```
#first write a function that you want to minimize
wolfLL <- function(m, s){

  #if an impossible value for a param
  #return NaN
  if(s <= 0) return(NaN)

 -sum(dnorm(inbreeding.coefficient,
                                    mean=m, sd=s,
                                    log=TRUE))
}

#now feed the function to an optimizer
wolfLL_Fit <- mle2(wolfLL, data=wolves, start=list(m=0.2, s=0.2))
```

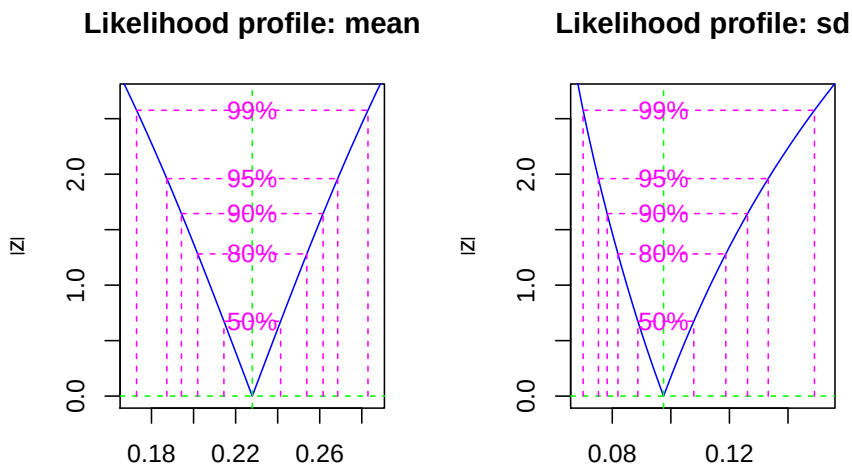
Coefficient Estimates and SE from MLE2

```
summary(wolfLL_Fit)

# Maximum likelihood estimation
#
# Call:
# mle2(minuslogl = wolfLL, start = list(m = 0.2, s = 0.2), data = wolves)
#
# Coefficients:
#   Estimate Std. Error z value   Pr(z)
# m   0.2279     0.0199   11.45 < 2e-16
# s   0.0975     0.0141    6.93 4.2e-12
#
# -2 log L: -43.63
```

Coefficient tests based on Wald Confidence Intervals - assume quadratic relationship at peak

Easy Profiling

```
mleWolvesNames <- mle2(inbreeding.coefficient ~
                         dnorm(mean=mean, sd=sd),
                       data=wolves, start=list(mean=0.1,sd=0.1))

plot(profile(mleWolvesNames))
```



Confidence Intervals

```
confint(wolfLL_Fit)

#     2.5 % 97.5 %
# m 0.18729 0.2685
# s 0.07529 0.1333

# Wald CIs assume quadratic relationship at peak
confint(wolfLL_Fit, method="quad")

#     2.5 % 97.5 %
# m 0.18891 0.2669
# s 0.06992 0.1251
```

Builtin Probability Distributions

```
mleWolves <- mle2(inbreeding.coefficient ~ dnorm(mean=m, sd=s),
                  data=wolves, start=list(m=0.1,s=0.1))
```

Or...MLE2 has Many Probability Functions Builtin

```
mleWolvesPups <- mle2(pups ~ dpois(lambda=lambda),
                      data=wolves, start=list(lambda=3))
```

Linear Regression with ML

```
wolfRegLL <- function(slope, int, sdResid){

  #in case of non-possible SD value, NaN
  if(sdResid <= 0) return(NaN)

  #fitted values as means
  fittedPups <- slope * inbreeding.coefficient + int

  -sum(dnorm(pups, mean = fittedPups, sd = sdResid, log=T))
}

mleWolfLM <- mle2(wolfRegLL, data=wolves,
                  start=list(slope=-10, int=6, sdResid =1.2))
```

We Can Flexibly Compare Models using LRT

```
mleWolfNull <- mle2(wolfRegLL, data=wolves,
                    start=list(int=6, sdResid =1.2),
                    fixed=list(slope=0))

anova(mleWolfLM, mleWolfNull)

# Likelihood Ratio Tests
# Model 1: mleWolfLM, [wolfRegLL]: slope+int+sdResid
# Model 2: mleWolfNull, [wolfRegLL]: int+sdResid
#   Tot Df Deviance Chisq Df Pr(>Chisq)
# 1      3     86.2
# 2      2     97.3  11.1  1    0.00088
```

To fit Ho for a linear model fit, fix or drop the predictor variable alltogether.

Exercise: Pufferfish & Predator Responses

- Load the pufferfish data
- Write a MLE regression for predators resemblance
- This model will have three parameters
- Evluate it's CIs and Wald Tests
- Compare it to your Ho
- Compare it to `lm` results

Regression as Function

```
#load the pufferfish data
puffer <- read.csv("./data/16q11PufferfishMimicry Caley & Schluter 2003.csv"

pufferFun <- function(slope, intercept, residSD){
  #in case of non-possible SD value, NaN
  if(residSD <= 0) return(NaN)

  predatorsFit <- intercept + slope*resemblance

  -sum(dnorm(predators, predatorsFit, sd=residSD, log=T))
}

puffer_mle <- mle2(pufferFun,
                   start=list(slope=1, intercept=1, residSD=2),
                   data=puffer)
```

Regression with Distribution

```
puffer_mle2 <- mle2(predators ~ dnorm(slope*resemblance +
                                      intercept, sd=residSD),
                    start=list(slope=1, intercept=1, residSD=2),
                    data=puffer)
```

Confidence Intervals

```
confint(puffer_mle)

#             2.5 % 97.5 %
# slope       1.874  4.105
# intercept -1.017  4.866
# residSD     2.188  4.092
```

Null Hypothesis Test

```
puffer_mle_null <- mle2(predators ~ dnorm(intercept, sd=residSD),
                        start=list(intercept=9, residSD=4),
                        data=puffer)

anova(puffer_mle, puffer_mle_null)

# Likelihood Ratio Tests
# Model 1: puffer_mle, [pufferFun]: slope+intercept+residSD
# Model 2: puffer_mle_null,
#            predators~dnorm(intercept,sd=residSD)
#   Tot Df Deviance Chisq Df Pr(>Chisq)
# 1      3     99.3
# 2      2    117.8  18.5  1    1.7e-05
```

Comparison to LM

```
coef(summary(puffer_mle))

#           Estimate Std. Error z value     Pr(z)
# slope        2.990     0.5421   5.515 3.493e-08
# intercept    1.925     1.4291   1.347 1.781e-01
# residSD      2.897     0.4580   6.325 2.540e-10

coef(summary(lm(predators ~ resemblance, data=puffer)))

#             Estimate Std. Error t value  Pr(>|t|)
# (Intercept)    1.925     1.5064   1.278 2.176e-01
# resemblance    2.989     0.5714   5.232 5.638e-05
```