FAS6337C - Lab 7

Marcel Gietzmann-Sanders

Estimation of Stock-Recruit Relationships

You have obtained some density estimates for Walleye Sander vitreus from Lake Escanaba, Wisconsin. Age-0 and adult Walleye population estimates were obtained using intensive mark-recapture experiments. Data for spawners and recruits exist from 1958-1991.

The objective of the lab is to:

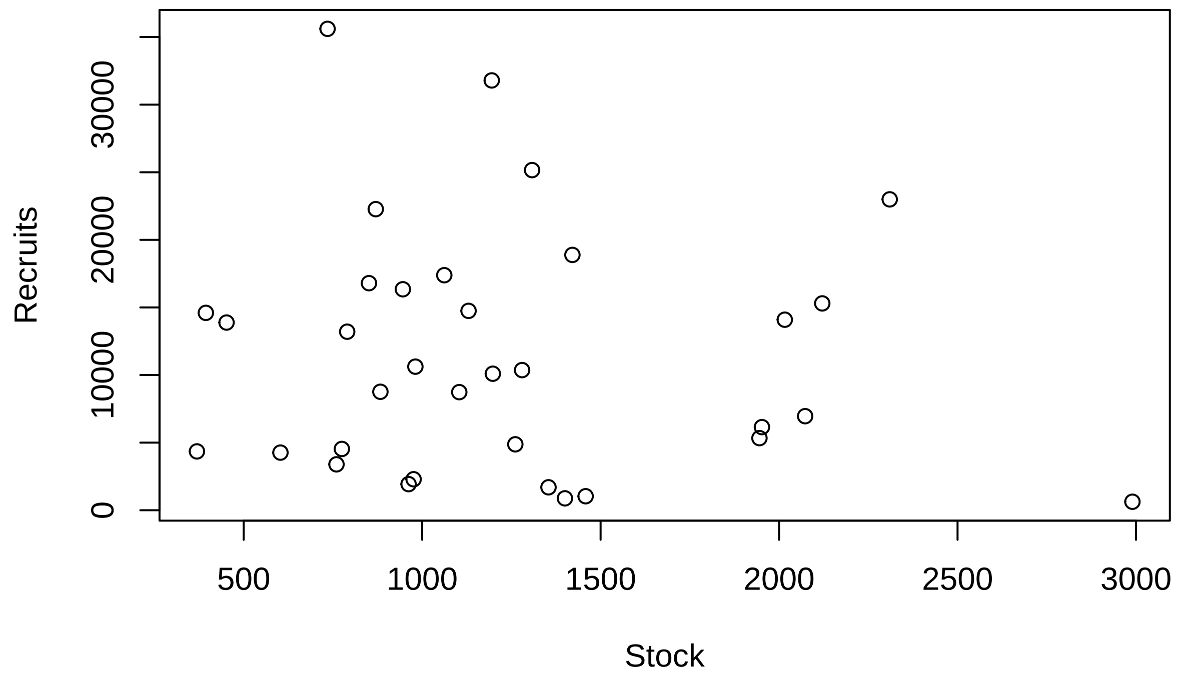
- Fit the most appropriate stock-recruitment equation and assess the overall fit of the selected model.

```r
setwd("/workspaces/schooling/population_dynamics/lab_7/")
data <- read.table("data/walleye_data.txt", header=T, sep="")
head(data)
```

```
##   Year     R    S
## 1 1958  4532  775
## 2 1959 22996 2310
## 3 1960   628 2990
## 4 1961   879 1400
## 5 1962 14747 1130
## 6 1963 13205  790
```

1. Plot recruits (R) on stock (S).

```r
plot(
    data$S,
    data$R,
    xlab="Stock",
    ylab="Recruits"
)
```



Does there appear to be a relationship?

If I didn't know we were looking for one in this lab, I'd never think there was a relationship here. Perhaps there's more variance in the middle versus the ends but that just feels natural with any kind of "clustered" data.

Is it linear or nonlinear?

Given the change in variance with stock I'd say it's nonlinear.

2. Non-linear Additive Error Structure Model:

Write an Excel/R equation to predict the number of recruits (R) for a given stock size (Eq. 1).
$R = aSe^{-bS}$

```
nonlinear_ricker <- function(params, S) {
    a <- params[1]
    b <- params[2]
    return(a * S * exp(-b * S))
}
```

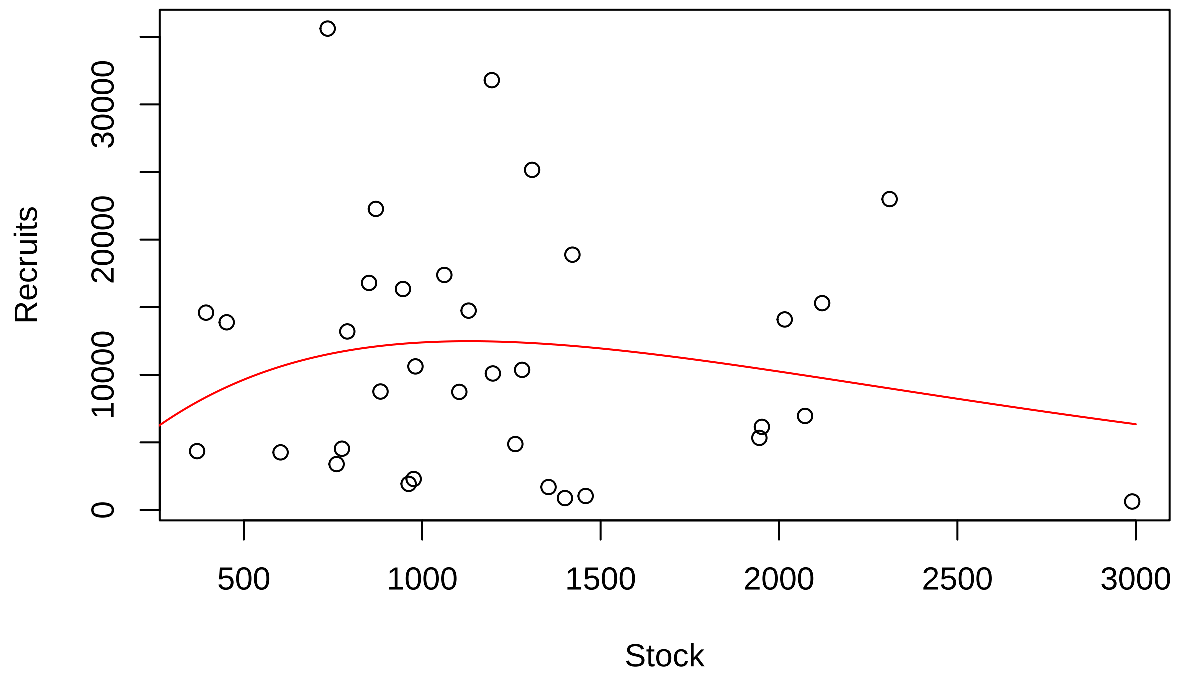
Using a maximum likelihood framework with a normal error distribution (additive error structure), fit the Ricker stock-recruit relationship to the data.

```
normal_NLL <- function(params, pred, obs) {
    sigma <- tail(params, 1)
    return(-1 * sum(dnorm(obs, pred, sigma, log=T)))
}
```

```
func <- function(params) {
    pred <- nonlinear_ricker(params, data$S)
    return(normal_NLL(params, pred, data$R))
}

params <- c(30, 0.001, 8000)
for (i in 1:10) {
    fit <- optim(params, func)
    params <- fit$par
}
fit
```

```
## $par
## [1] 2.999706e+01 8.839359e-04 8.799996e+03
##
## $value
## [1] 356.6238
##
## $counts
## function gradient
##      134       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

```
S <- seq(0, 3000, 1)
full_span <- data.frame(S)
full_span$nonlinear_additive <- nonlinear_ricker(params, full_span$S)
data$nonlinear_additive <- nonlinear_ricker(params, data$S)
plot(
    data$S,
    data$R,
    xlab="Stock",
    ylab="Recruits"
)
lines(
    full_span$S,
    full_span$nonlinear_additive,
    col="red"
)
```



What are the estimates of a and b?

```
print(paste("a =", params[1]))
```

```
## [1] "a = 29.9970616863395"
```

```
print(paste("b =", params[2]))
```

```
## [1] "b = 0.000883935857074229"
```

3. Linear Multiplicative Error Structure Model:

Write an Excel/R equation to predict the log-transformed recruit-stock ratio $\ln\frac{R}{S} = intercept + slope \bullet S$

```
linear_multiplicative <- function(params, S) {
    intercept <- params[1]
    slope <- params[2]
    return(intercept + slope * S)
}
```

Using linear regression and a maximum likelihood framework with a normal error distribution, fit the Ricker stock-recruit relationship:

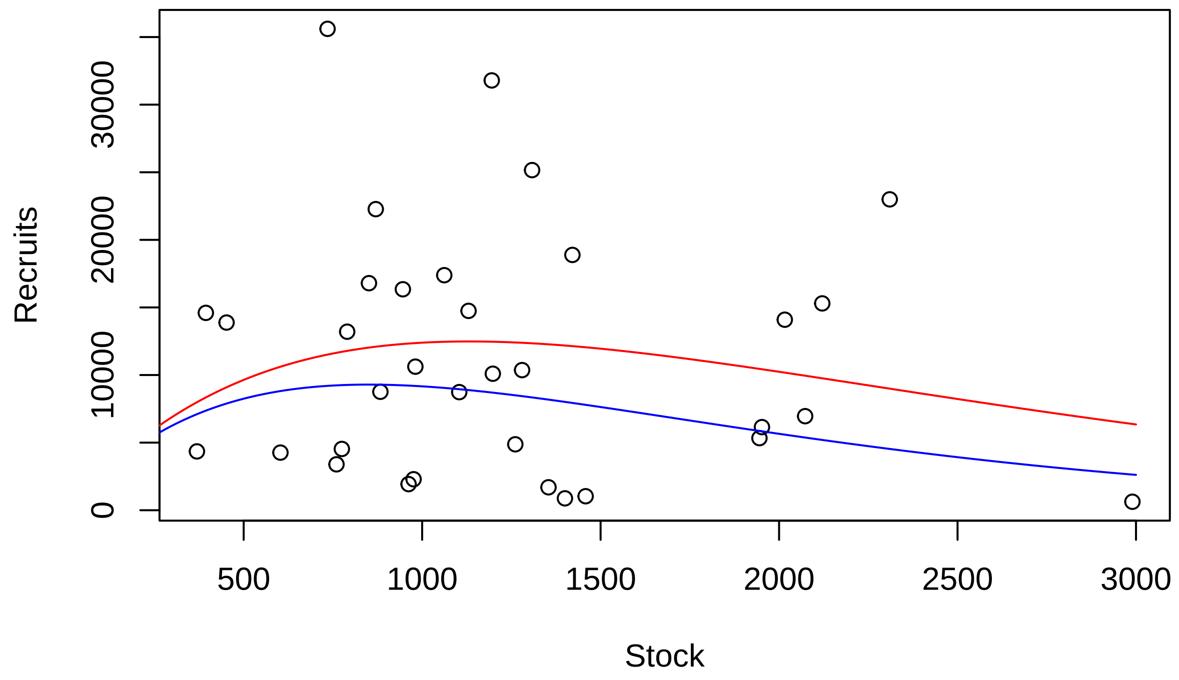
```
func <- function(params) {
    pred <- linear_multiplicative(params, data$S)
    obs <- log(data$R / data$S)
    return(normal_NLL(params, pred, obs))
}

params <- c(30, -0.001, 1)
for (i in 1:10) {
    fit <- optim(params, func)
    params <- fit$par
}
```

```
## Warning in dnorm(obs, pred, sigma, log = T): NaNs produced

## Warning in dnorm(obs, pred, sigma, log = T): NaNs produced

## Warning in dnorm(obs, pred, sigma, log = T): NaNs produced
```

```
fit
```

```
## $par
## [1]  3.391546827 -0.001176288  0.996917470
##
## $value
## [1] 48.14017
##
## $counts
## function gradient
##       90       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

```
full_span$linear_multiplicative <- exp(linear_multiplicative(params, full_span$S)) * full_span$S
data$linear_multiplicative <- exp(linear_multiplicative(params, data$S)) * data$S
plot(
    data$S,
    data$R,
    xlab="Stock",
    ylab="Recruits"
)
lines(
    full_span$S,
    full_span$nonlinear_additive,
    col="red"
)
lines(
    full_span$S,
    full_span$linear_multiplicative,
    col="blue"
)
```



Convert the slope and intercept, output the parameters a and b.

```
print(paste("a =", exp(params[1])))
```

```
## [1] "a = 29.7118758616848"
```

```
print(paste("b =", -params[2]))
```

```
## [1] "b = 0.00117628807739535"
```

Are they the same parameters as the additive error structure model; if not, how did they change?

They are not the same, a dropped slightly and b rose. The overall effect as can be seen from the plot is the predictions shrank.

4. Linear Multiplicative Error Structure Model version 2

Write an Excel/R equation to predict the log-transformed recruits.

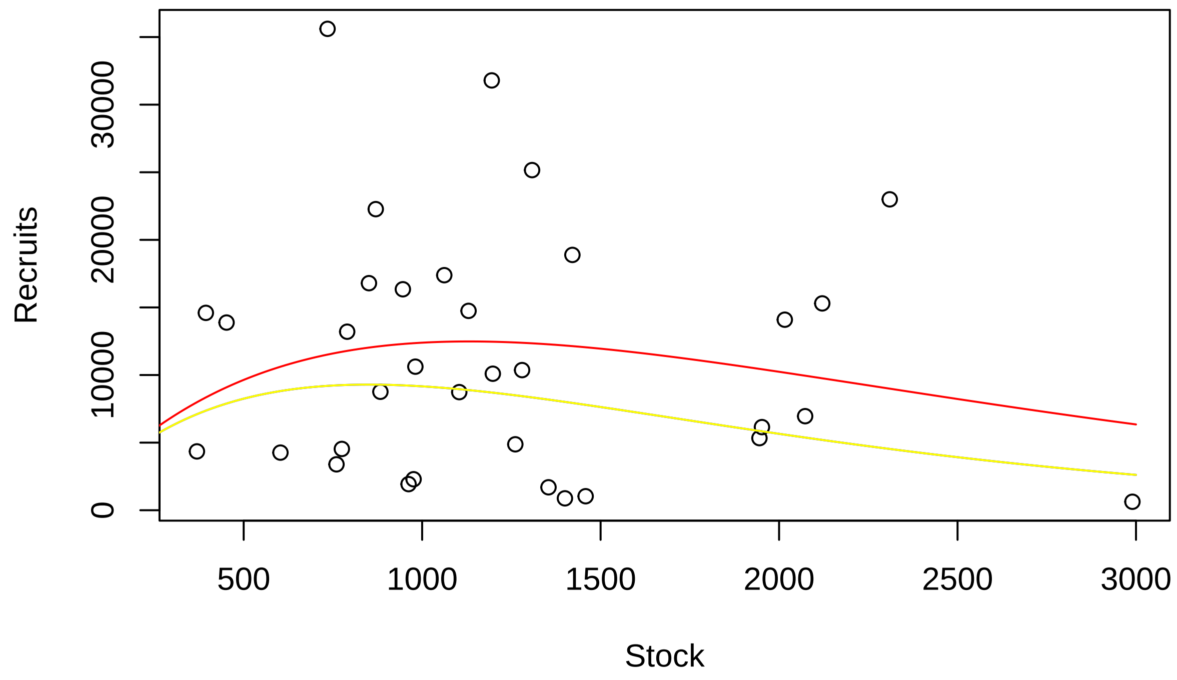
We'll just be reusing the nonlinear_ricker and taking the logarithm before feeding it to NLL.

Using a maximum likelihood framework, fit the Ricker stock-recruit relationship with a normal error distribution

```
func <- function(params) {
    pred <- nonlinear_ricker(params, data$S)
    obs <- data$R
    return(normal_NLL(params, log(pred), log(obs)))
}

params <- c(30, -0.001, 1)
for (i in 1:10) {
    fit <- optim(params, func)
    params <- fit$par
}
fit
```

```
## $par
## [1] 29.716561340  0.001176316  0.996932778
##
## $value
## [1] 48.14017
##
## $counts
## function gradient
##       97       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

```
full_span$linear_multiplicative_v2 <- nonlinear_ricker(params, full_span$S)
data$linear_multiplicative_v2 <- nonlinear_ricker(params, data$S)
plot(
    data$S,
    data$R,
    xlab="Stock",
    ylab="Recruits"
)
lines(
    full_span$S,
    full_span$nonlinear_additive,
    col="red"
)
lines(
    full_span$S,
    full_span$linear_multiplicative,
    col="blue"
)
lines(
    full_span$S,
    full_span$linear_multiplicative_v2,
    col="yellow"
)
```



How do the parameter estimates compare to linear regression fit

They are the same.

5. Nonlinear Multiplicative Error Structure Model

Write an Excel/R equation to predict recruits

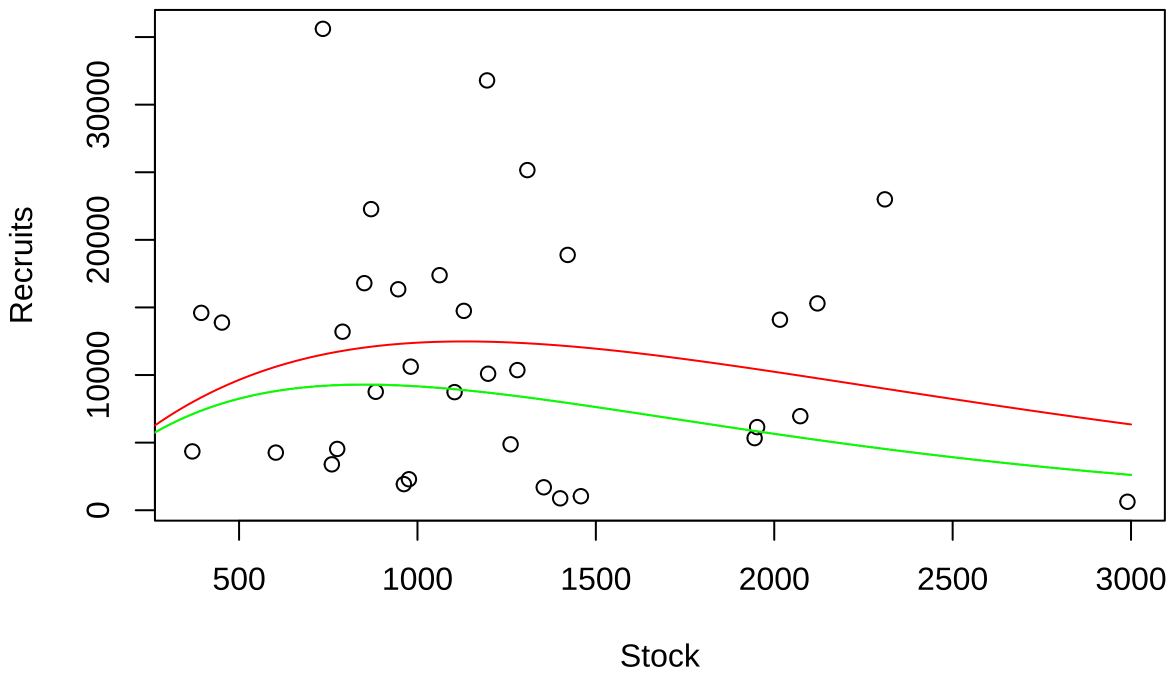
We will reuse nonlinear ricker.

Using a maximum likelihood framework, fit the Ricker stock-recruit relationship with a log-normal error distribution

```r
log_NLL <- function(params, pred, obs) {
    sigma <- tail(params, 1)
    return(-1 * sum(dlnorm(obs, log(pred), sigma, log=T)))
}

func <- function(params) {
    pred <- nonlinear_ricker(params, data$S)
    obs <- data$R
    return(log_NLL(params, pred, obs))
}

params <- c(30, -0.001, 1)
for (i in 1:10) {
    fit <- optim(params, func)
    params <- fit$par
}
fit
```

```
## $par
## [1] 29.717523482  0.001176523  0.996814408
##
## $value
## [1] 352.3424
##
## $counts
## function gradient
##       85       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

```r
full_span$nonlinear_multiplicative <- nonlinear_ricker(params, full_span$S)
data$nonlinear_multiplicative <- nonlinear_ricker(params, data$S)
plot(
    data$S,
    data$R,
    xlab="Stock",
    ylab="Recruits"
)
lines(
    full_span$S,
    full_span$nonlinear_additive,
    col="red"
)
lines(
    full_span$S,
    full_span$linear_multiplicative,
    col="blue"
)
lines(
    full_span$S,
    full_span$linear_multiplicative_v2,
    col="yellow"
)
lines(
    full_span$S,
    full_span$nonlinear_multiplicative,
    col="green"
)
```

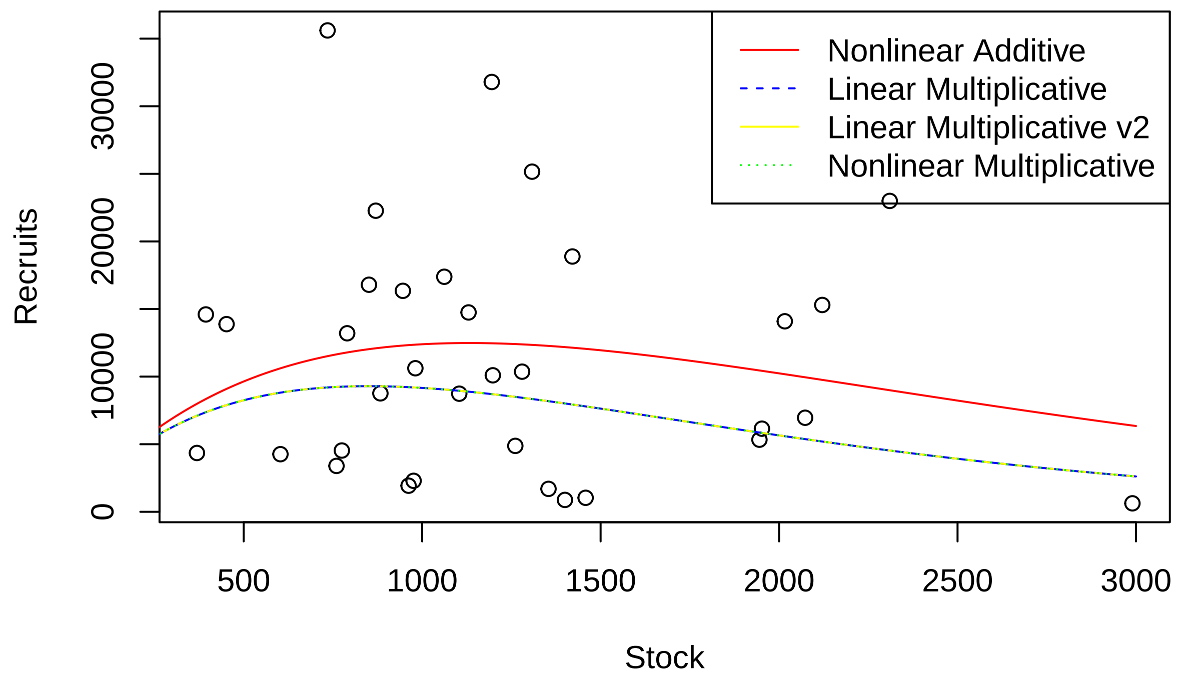


Does it give the same parameter estimates as the linear regression?

Same estimates again.

6. Compare all fits of the Ricker by predicting the expected recruitment for a range of adults from 0 to 3000 adults, then plotting each of these predictions in one plot (make sure to label the predictions).

```r
plot(
    data$S,
    data$R,
    xlab="Stock",
    ylab="Recruits"
)
lines(
    full_span$S,
    full_span$nonlinear_additive,
    col="red",
    lty=1
)
lines(
    full_span$S,
    full_span$linear_multiplicative,
    col="blue",
    lty=1
)
lines(
    full_span$S,
    full_span$linear_multiplicative_v2,
    col="yellow",
    lty=2
)
lines(
    full_span$S,
    full_span$nonlinear_multiplicative,
    col="green",
    lty=3
)
legend(
    "topright",
    legend=c(
        "Nonlinear Additive",
        "Linear Multiplicative",
        "Linear Multiplicative v2",
        "Nonlinear Multiplicative"
    ),
    col=c("red", "blue", "yellow", "green"),
    lty=c(1, 2, 1, 3)
)
```



7. Nonlinear Multiplicative Error Structure Model—Beverton-Holt

Write an Excel/R equation to predict recruits using the Beverton-Holt stock-recruit relationship

```r
beverton_holt <- function(params, S) {
    a <- params[1]
    b <- params[2]
    return(a * S / (1 + b * S))
}
```

Using a maximum likelihood framework, fit the Beverton-Holt stock-recruit relationship with a log-normal error distribution

```r
func <- function(params) {
    pred <- beverton_holt(params, data$S)
    obs <- data$R
    return(log_NLL(params, pred, obs))
}

params <- c(30000, 4, 1)
for (i in 1:10) {
    fit <- optim(params, func)
    params <- fit$par
}
```

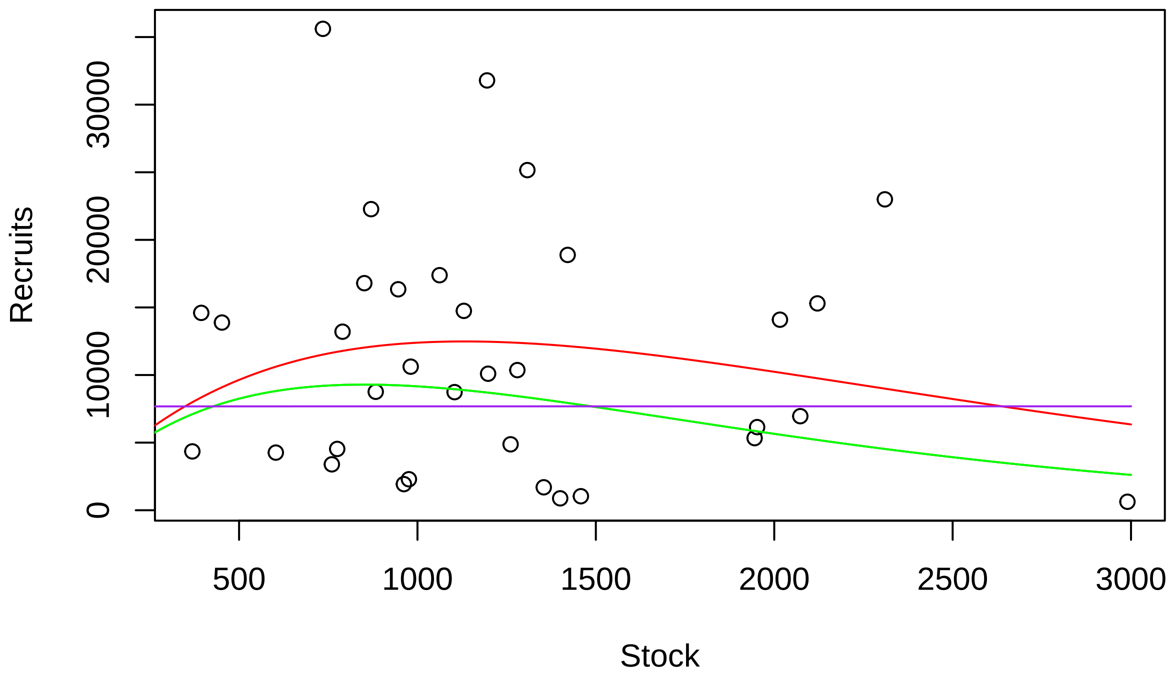
```
## Warning in log(pred): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced

## Warning in log(pred): NaNs produced
## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced

## Warning in dlnorm(obs, log(pred), sigma, log = T): NaNs produced
```

```
fit
```

```
## $par
## [1] 31237.976033     4.063805     1.029735
##
## $value
## [1] 353.4396
##
## $counts
## function gradient
##      118       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

Output the predicted values and plot them against the raw data.

```r
full_span$beverton_holt <- beverton_holt(params, full_span$S)
data$beverton_holt <- beverton_holt(params, data$S)
plot(
    data$S,
    data$R,
    xlab="Stock",
    ylab="Recruits"
)
lines(
    full_span$S,
    full_span$nonlinear_additive,
    col="red"
)
lines(
    full_span$S,
    full_span$linear_multiplicative,
    col="blue"
)
lines(
    full_span$S,
    full_span$linear_multiplicative_v2,
    col="yellow"
)
lines(
    full_span$S,
    full_span$nonlinear_multiplicative,
    col="green"
)
lines(
    full_span$S,
    full_span$beverton_holt,
    col="purple"
)
```



Which model appears to fit the data better?

Honestly a mean would probably do as well as the Beverton-Holt model. So I'm going to go with the Ricker is better. Technically the Ricker also has lower loss. But honestly it's not by much.

8. From the observed data:

Do you think there are indications of any of the problems commonly encountered estimating stock-recruit relationships (e.g. time-series bias, error-in-variables, etc.)? Why or why not?

- There is clearly time series bias in this data as the variance is not constant. Really high and really low values of the stock have less representation than stock values in the middle.
- Given the nature in which this data was collected I suspect we've got pretty low error in the variables themselves (given every single boat was checked)
- Likewise there's quick a lot of data here as this has 34 years represented so I would say we're not dealing with limited data (besides the time series bias mentioned above).

9. Suppose you (as a scientist) were approached by a manager who was concerned with the variability apparent in the relationship. Should the manager be concerned, and what further research might you suggest?

I would suggest that the manager shouldn't be overly concerned. While there is a lot of variability, the real concern here would be if it is that likely we could drive the stock down to a point where compensation wouldn't happen. But if you look at the collected data, at every single stock size represented (besides our $S = 3000$ point) the majority of the data shows recruitment being far higher than the stock size itself. Even at very low values of stock we see recruitment being high, so if issues are going to arise it's not going to be as a result of stock size but other factors.

Therefore I'd recommend the manager look into the life history of this fish and understand what other factors (perhaps environmental) might result in low recruitment. But overall I think these results suggest that at the levels this stock has been fished stock size is not a concern for recruitment levels.

10. Navigate to https://zsiders.shinyapps.io/Lab_6/ and answer the following:

Reduce the σ to 0.05 and leaving a at 20 for Ricker and Beverton-Holt SRR simulations then determine the b for each SRR that results in Ricker and Beverton-Holt predicting approximately the same stock-recruit relationship.

- 0.0004 for Ricker
- 0.001 for Beverton-Holt

Now increase the σ for both SRRs to 0.2, 0.5, and 1 and determine the b for each SRR that results in Ricker and Beverton-Holt predicting approximately the same stock-recruit relationship.

$\sigma = 0.2$ - 0.0006 for Ricker - 0.001 for Beverton-Holt

$\sigma = 0.5$ - 0.0012 for Ricker - 0.005 for Beverton-Holt

$\sigma = 1.0$ - 0.0018 for Ricker - 0.007 for Beverton-Holt

What is the role of error/noise (σ) in determining which SRR is appropriate for the data?

As the underlying data gets noiser there is a wider range of parameters (b specifically) for which the two models will predict the same stock-recruitment relationship. I.e. as the data gets noisier it becomes harder and harder to determine which model is more "correct".

Summary of All Models

Stock-Recruit Model	Equation	Error Structure	a	b	σ	Question
Ricker	$R = aSe^{-bS}$	$N(R, \sigma)$	29.997	0.0009	8800	2
Ricker	$\ln R/S = intercept + slope \bullet S$	$N(\ln R/S, \sigma)$	29.712	0.0012	0.997	3
Ricker	$\ln R = \ln aSe^{-bS}$	$N(\ln R)$	29.712	0.0012	0.997	4
Ricker	$R = aSe^{-bS}$	$LN(R, \sigma)$	29.712	0.0012	0.997	5
Beverton-Holt	$R = \frac{aS}{1+bS}$	$LN(R, \sigma)$	31238	4.06	1.03	7