Running Linear and Generalized Additive Models

Generating data

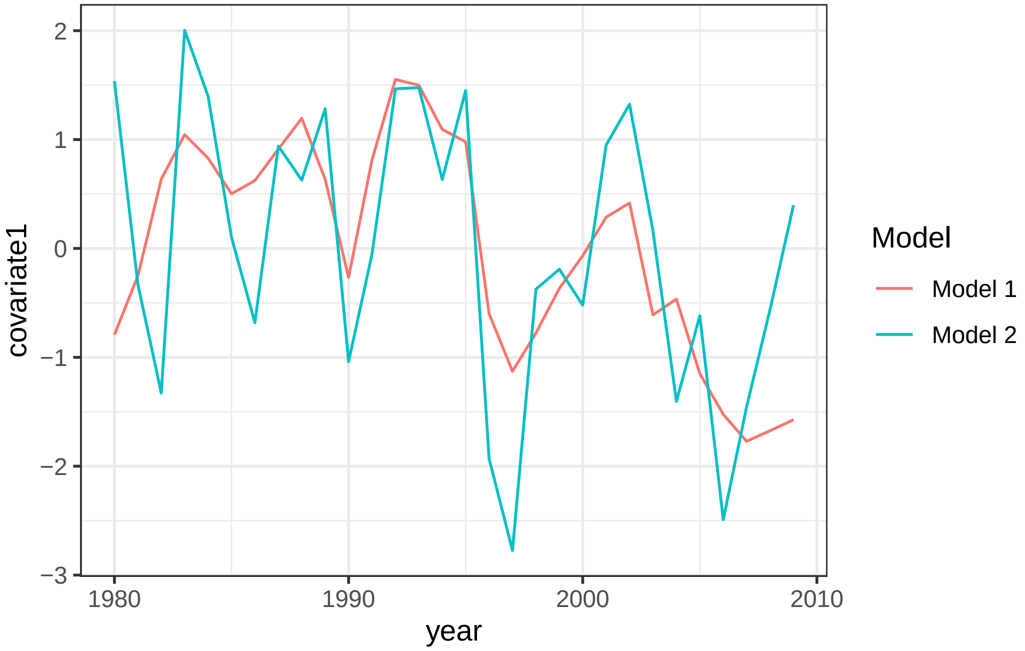
There are two ways we can fit these models:

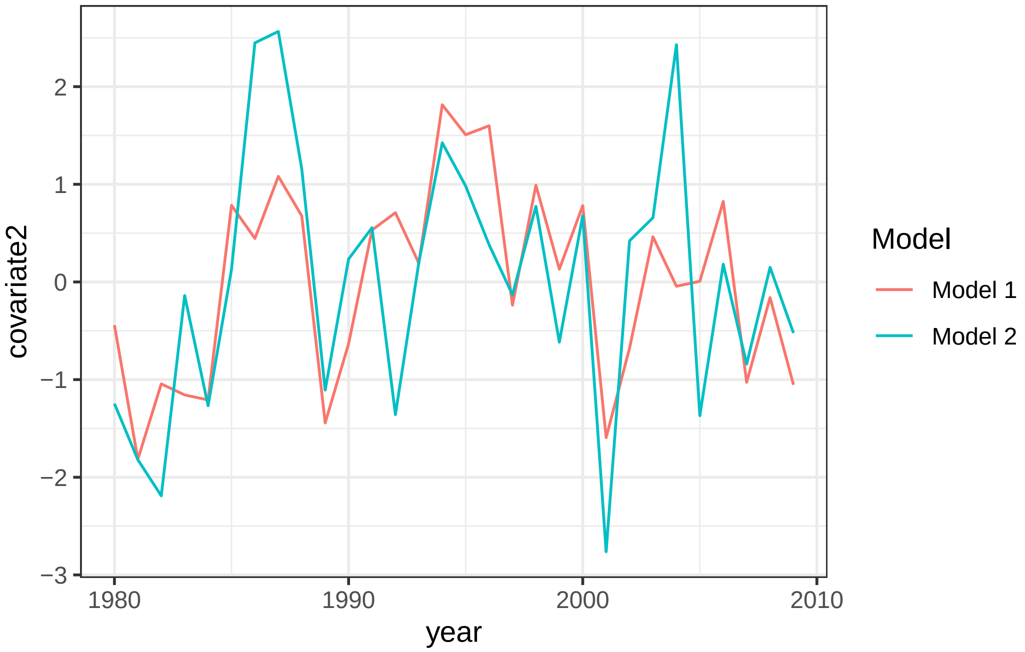
1) we can fit 2 separate models one for ROMs and one for GLORYs and compare r-squared and 2) we can fit 1 model and use a interaction term or model selection process. Doing #1 is easier and faster while #2 is better suited for a model selection technique.

To start, lets simulate some example data. Simulating data is a really great tool for testing models because you can simulate it with known or true relationships (based on values you assign).

```
set.seed(99)
year<- rep(seq(1980, 2009),2) # a variable for year
modelnames<- rep(c("Model 1", "Model 2"), each=30) # a variable for models,
#equivalent to ROMS/GLORYs

cov1mod1<- arima.sim(list(order = c(1,0,0), ar = 0.7), n = 30)%>% #random time series
  scale() # standardize it!
cov1mod2 <- cov1mod1+rnorm(n= 30, mean = .5, sd=0.3)%>%
  scale()#second time series that
#covaries with the first

set.seed(150)
cov2mod1<- arima.sim(list(order = c(1,0,0), ar = 0.5), n = 30)%>% #random time series
  scale() # standardize it!
cov2mod2 <- cov2mod1+rnorm(n= 30, mean =1, sd=1)%>%
  scale()
  #second time series that
#covaries with the first
```

```
dataset <- data.frame(year=year,covariate1=c(cov1mod1,cov1mod2),covariate2=c(cov2mod1,cov2mo
```





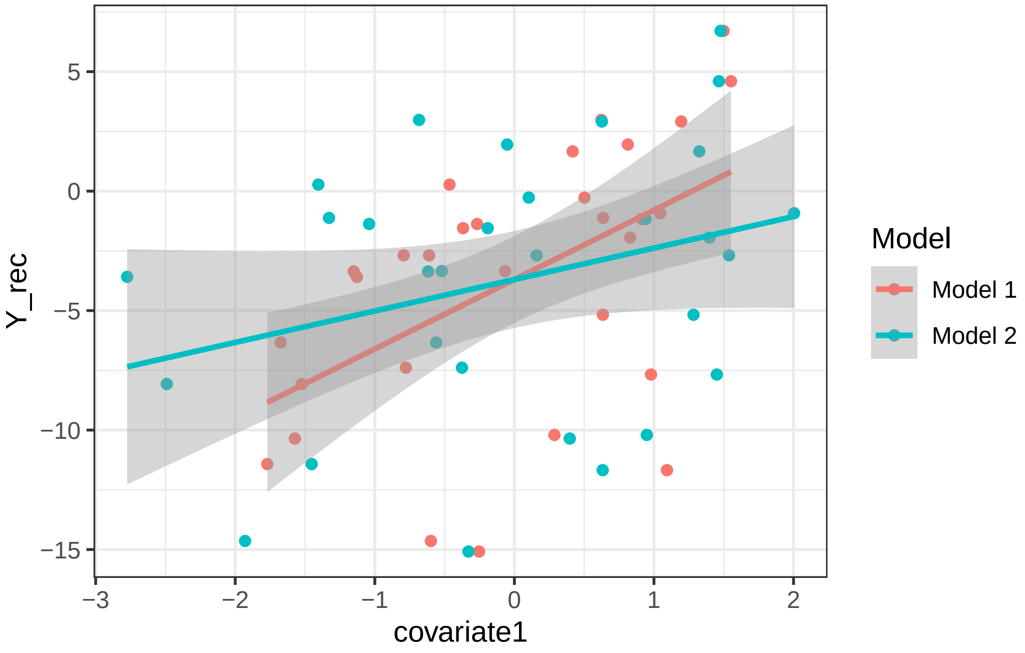
Now lets simulate the recruitment data, lets pretend recruitment has a linear relationship with covariate 2 and a linear relationship with covariate 1. Let pretend model 1 is a better predictor of Y rec

```r
linear1 <- 4
linear2 <- 0.6
quad2<- 5
intercept <- 1
error<- rnorm(30, 0.1,0.5)

#this is just a simple qudratic equation with 2 covraiates plus random error
Y_rec <- linear1*cov1mod1-linear2*cov2mod1-quad2*cov2mod1^2+error+intercept

data_full<-dataset%>%
  mutate(Y_rec=rep(Y_rec, 2))
```

Lets look at the relationship between covariate 1 and recruitment.

```r
ggplot(data=data_full, aes(x=covariate1, y=Y_rec, group=Model, col=Model))+
  geom_point()+
  geom_smooth(method='lm')+
  theme_bw()
```

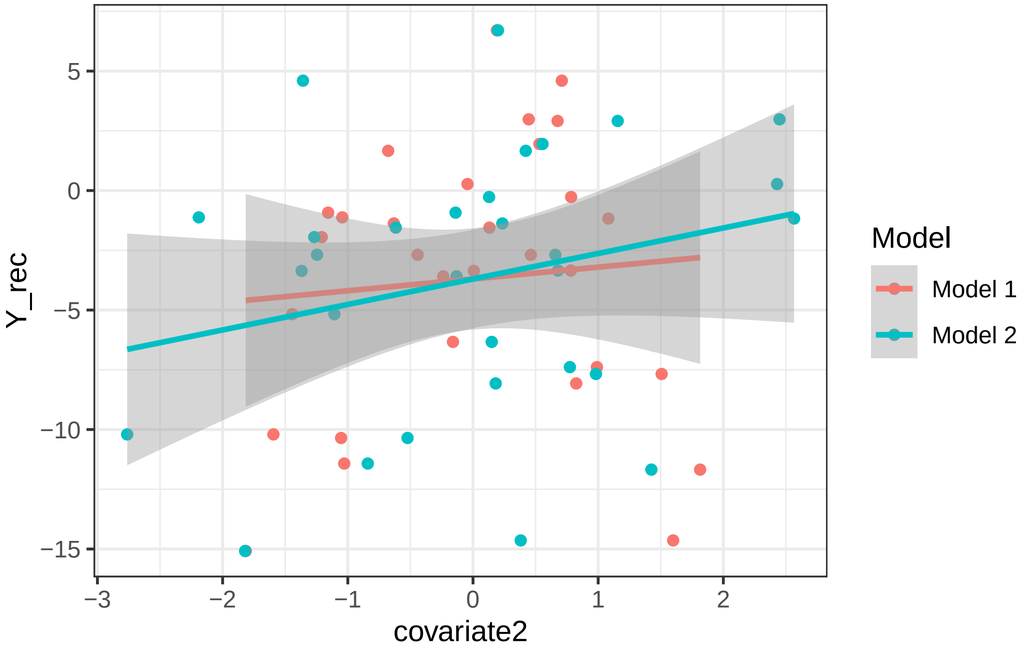
```
`geom_smooth()` using formula = 'y ~ x'
```



It looks pretty good! The covariate data look fairly linearly related, but it is impossible to tell which model does a better job just by looking.

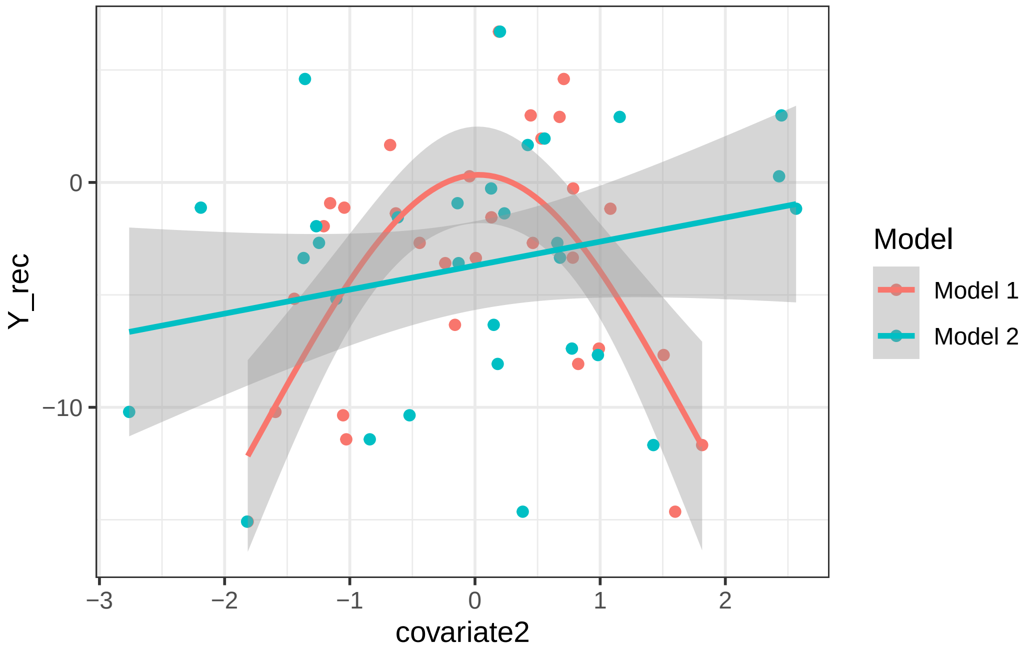
```
ggplot(data=data_full, aes(x=covariate2, y=Y_rec, group=Model, col=Model))+
  geom_point()+
  geom_smooth(method='lm')+
  theme_bw()
```

```
`geom_smooth()` using formula = 'y ~ x'
```



Hmm. this looks like a mess. These do not look like great linear relationships. What if we try looking at a more flexible model that can handle more relationship shapes! *This formulation of the geom_smooth() call is what I reccommend you use on your data*

```
ggplot(data=data_full, aes(x=covariate2, y=Y_rec, group=Model, col=Model))+
  geom_point()+
  geom_smooth(method = "gam", formula = y ~ s(x, k = 3))+
  theme_bw()
```



Much better! But it appears model 1 and 2 explain the data very differently and have very different relationships. For these plots we were just using the GAM and LM function built into ggplot. That is a great first pass that would be useful with the data you are using! BUT we also want actual information about the fits and the parameters that we can only get from actually fitting the model…so lets try fitting a some linear and generalized additive models first.

GAMs

GAM stand for a generalized additive model. It is useful and used frequently in ecology because it allows for more flexible "functional relationships" aka the relationship between your driver and predictor, than linear models. For linear models you need to use quadratic or cubic terms to fit relationships that do not follow a line, and as a result they are "rigid".

```
1 + 1
```

```
[1] 2
```

You can add options to executable code like this

```
[1] 4
```

The `echo: false` option disables the printing of code (only output is displayed).