Monte Carlo Simulations

Contents

Appendix A: Monte Carlo Simulations

In this appendix, I use Monte Carlo simulations to motivate and verify the use of Generalized Additive Models in order to estimate the effect of price on production. For the sake of the reproducibility, the full r-code and results are imbedded and explained in the document.

First, we load some r packages that we will need:

```
library(ggplot2) #for plotting
library(reshape2) #for some data manipulation
library(mgcv) #for GAM modeling
```

```
## Loading required package: nlme
## This is mgcv 1.8-6. For overview type 'help("mgcv-package")'.
```

Functions

Below is the code for a function that generates a starting year for an oil field conditional on the size of the oil field. I could use the actual field sizes and starting years from the Norwegian Continental Shelf, but generating them randomly inserts a generality to the Monte Carlo results.

The function generates a starting year for field production from a half-normal distribution with a mean at 1970, but constrained to be between 1970 to 1990. This is meant to reflect the common industry pattern that most significant finds are found early in a regions activity.

```
gen_year<-function(sizexz){
    #half-normal distribution with mean at 1970
    #constrained to be between 1970 and 1990
    range<-FALSE
    while(range==FALSE){
        year<-trunc(rnorm(1,mean=(1970), sd=10))
        ifelse(year>=1970 & year<=2000, range<-TRUE, range<-FALSE)
        }

return(year)
}
```

The following function takes a variable field with three components: size, starting year, and id. It then generates a time grid of production proportional to the square root of the size. It then generates a smooth cumulative time path of production based on the logistic function:

$cumProd = \frac{size}{1+exp(\frac{-prodTime_t}{3})}$

The cumulative production series is then first-differenced to created the production profile of the field, and a data frame with the production shape and other field characteristics are returned.

```
gen_production_shape<-function(field){
    #Given size of field, generates production shape based on
    #
    #field<-c(20, 1990, 1)
    #
    size<-as.numeric(field[1])
    start<-as.numeric(field[2])
    name<-field[3]

    t<-trunc(sqrt(size)) + 3
    time<- -t:(t)
    year<-start:(start+2*t-1)

    #use a cumulative logistic function to represent shape of function
    cum_production <- 1/(1+exp(-(time)/3))*size

    #take difference to create production per year
    prod_shape <- diff(cum_production)
    prod_years <- 1:length(prod_shape)
    production<-data.frame(year, prod_shape, name, size, prod_years)
    return(production)
}
```

The next function takes input of a data frame with the smooth production profile and oil prices, as well as the "true" effect of oil price on production beta. The oil price effect is then added to the production series as well as uncertainty in the form of a draws from a log-normal distribution. The final production data can then be described as:

$log(production) = f`(time) + beta * log(price) + epsilon$

where f is the logistic cumulative distribution function.

```
add_uncertainty<-function(beta, fields){
    fields$prod<-fields$prod_shape*
        exp(beta*(fields$prices))*
            rlnorm(length(fields$prices), meanlog=0, sdlog=.05)
    return(fields)
}
```

The below function takes a regression formula and a data frame of field production data and runs a GAM regression, returning the coefficient on *prices* variable.

```
run_gam<-function(formula, fields){
    #runs GAM model and returns coefficient on price

    gam_sim<-gam(formula,
        family=gaussian(link=log), weights=size, data=fields,
        na.action='na.omit')
    coefficients(gam_sim)
    return(coefficients(gam_sim)["prices"])
}
```

The below function takes a formula and a data frame of field production data and runs a GLM model, returning the estimated coefficient on the *prices* variable.

```
run_glm<-function(formula, fields){
    glm_sim<-lm(formula, data=fields)
    return(coefficients(glm_sim)["prices"])
}
```

The below function encapsulates the two core operations involved in each step of the Monte Carlo experiment. The effect of price and uncertainty are added to each field via the "add_uncertainty()" function. Then either a GAM or GLM regression is run on the data.

```
mc_run<-function(formula, use_gam=TRUE, fields, beta){
    #adds uncertainty then runs regression
    #
    fields_unc<-add_uncertainty(fields=fields, beta=beta)
    if(use_gam){
        return(run_gam(formula, fields_unc))
    }
    else{
        return(run_glm(formula, fields_unc))
    }
}
```

Generate fields

```
field_size<-round(exp(rnorm(40,mean=4, sd=1.5)), digits=1)
init_year<-trunc(sapply(field_size, gen_year))
fields<-cbind(field_size, init_year,1:length(init_year))
```

Then, we can load the oil price data

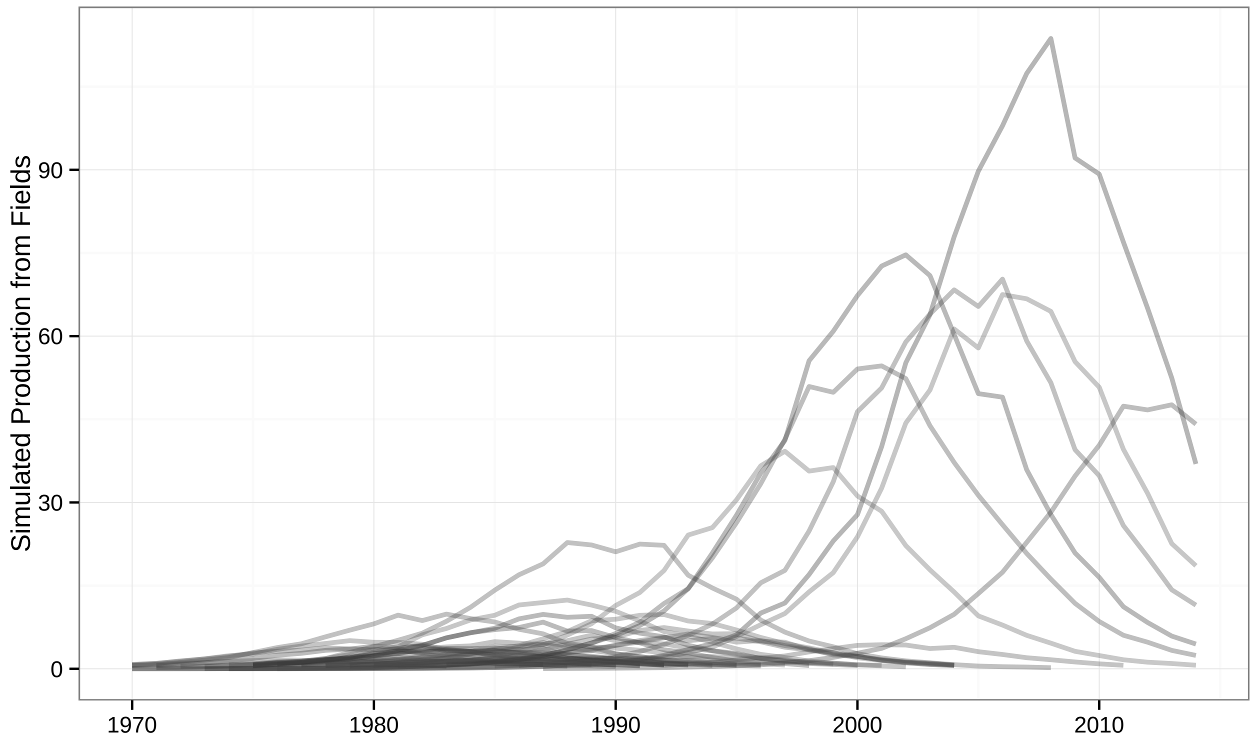
```
oil_price<-read.csv("/Users/johannesmauritzen/research/oil_prices/data/oil_price.csv")
prices<-oil_price[c("year", "oil_price_real")]
names(prices)<-c("year", "prices")
#change units to $10 to aid interpretation
prices$prices<-prices$prices/10
```

Now we create the simulated smoothed production profiles of the fields

```
sim_fields<-apply(fields, 1, gen_production_shape)
sim_fields<-Reduce(rbind, sim_fields)
sim_fields<-merge(sim_fields, prices, by="year")
sim_fields<-sim_fields[order(sim_fields$name, sim_fields$year),]
```

Below we do a single run of generating production uncertainty, while setting the effect of prices to zero to visualize what the data looks like. It looks reasonable.

```
sim_fields_unc<-add_uncertainty(fields=sim_fields, beta=0)
ggplot(sim_fields_unc)+
 geom_line(aes(x=year, y=prod, color=factor(name)),alpha=.3, size=1) +
 guides(color=FALSE) +
 scale_color_grey(start=0, end=.3) +
 theme_bw() +
 labs(x="", y="Simulated Production from Fields")
```



GLM Monte Carlo Experiment

Lets start with a monte-carlo regression using standard GLM, using a 4th-degree polynomial in order to try to control for the production profile of the fields.

```
formula_glm <- formula(I(log(prod)) ~ prod_years + I(prod_years^2) + I(prod_years^3) +
    I(prod_years^4) + size + prices + year + I(year^2) + I(year^3))
```

We then do a 100 runs of the Monte Carlo experiment, setting the "true" effect of price, $beta = 0$ and $beta = 0.05$. The latter being interpreted as a 10 dollar increase in the oil price leading to 5% more production.

```
glm_mc_beta0<-replicate(100, mc_run(beta=0, formula=formula_glm, fields=sim_fields, use_gam=FALSE))
glm_mc_beta05<-replicate(100, mc_run(beta=0.05, formula=formula_glm, fields=sim_fields, use_gam=FALSE))
```

We transform the data in order to make it easier to visualize

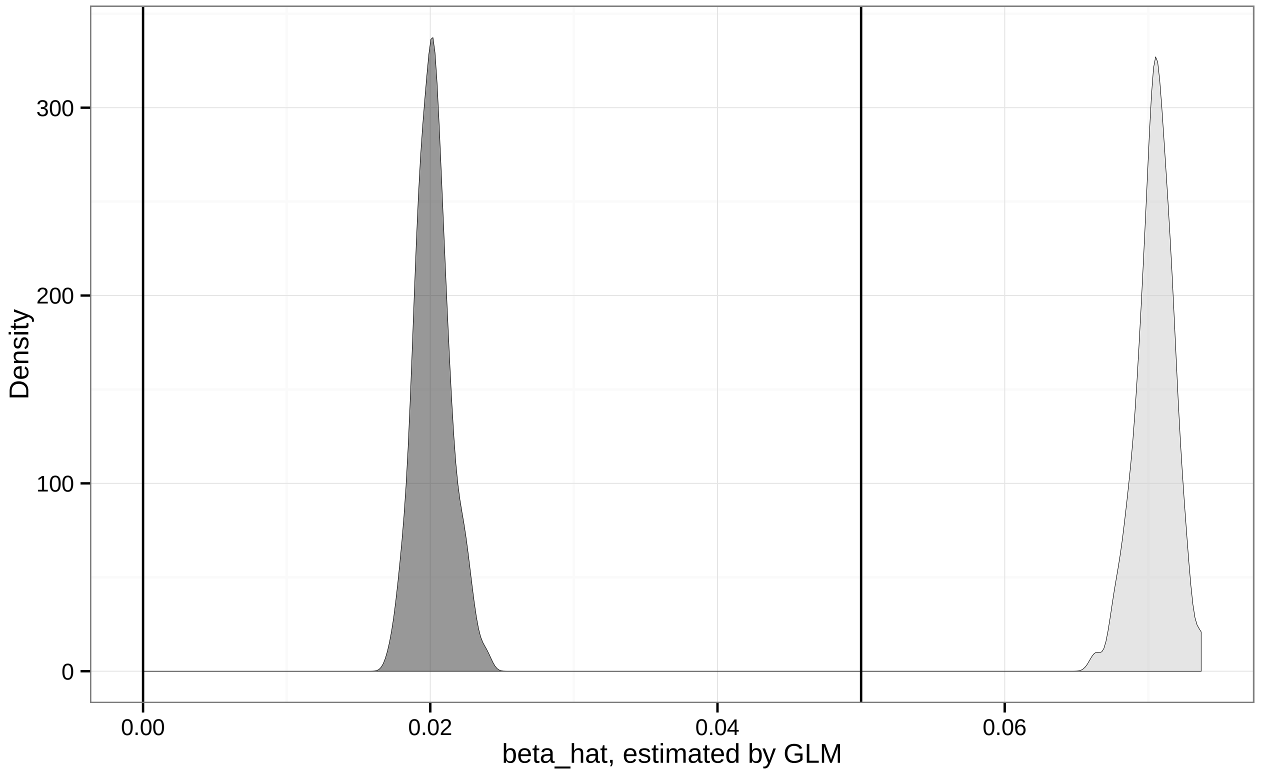
```
glm_mc_data<-data.frame(beta0=glm_mc_beta0, beta05=glm_mc_beta05)
glm_mc_data<-melt(glm_mc_data)
```

```
## No id variables; using all as measure variables
```

```
colnames(glm_mc_data)<-c("beta", "coefficient")
levels(glm_mc_data$beta)<-c("0", "0.05")
```

Below we plot the results in the form of empirical density functions of the results. The vertical black lines represent the true values of beta. Clearly, the GLM estimation has a significant negative bias.

```
ggplot(glm_mc_data, aes(x=coefficient, fill=beta)) +
geom_density(alpha=.5, size=0) +
geom_vline(aes(xintercept=c(0, 0.05))) +
scale_fill_grey() +
theme_bw() +
guides(fill=FALSE) +
xlab("beta_hat, estimated by GLM") +
ylab("Density")
```



GAM Monte Carlo Experiment

Below we set the formula as a slightly simplified version of our preffered GAM model. I have removed the cost index as this was not taken into account when generating the data. For simplicity I also do not take into account lagged price terms as the main point of this monte-carlo experiment is to show how the production profile can be effectively controlled for.

```
formula_1<- formula(prod~s(prod_years, size) + prices + year + I(year^2))
```

We then run the Monte Carlo experiment 100 times, again setting $\beta = 0$ and $\beta = 0.05$.

```
gam_mc_beta0<-replicate(100, mc_run(beta=0, formula=formula_1, fields=sim_fields, use_gam=TRUE))
```

```
## Warning: Step size truncated due to divergence

## Warning: Step size truncated due to divergence

## Warning: Step size truncated due to divergence
```

```
gam_mc_beta05<-replicate(100, mc_run(beta=.05, formula=formula_1, fields=sim_fields, use_gam=TRUE))
```

```
## Warning: Step size truncated due to divergence

## Warning: Step size truncated due to divergence

## Warning: Step size truncated due to divergence

## Warning: Step size truncated due to divergence

## Warning: Step size truncated due to divergence

## Warning: Step size truncated due to divergence

## Warning: Step size truncated due to divergence
```

Converting the data to a format for plotting

```
gam_mc_data<-data.frame(beta0=gam_mc_beta0, beta05=gam_mc_beta05)
gam_mc_data<-melt(gam_mc_data)
```

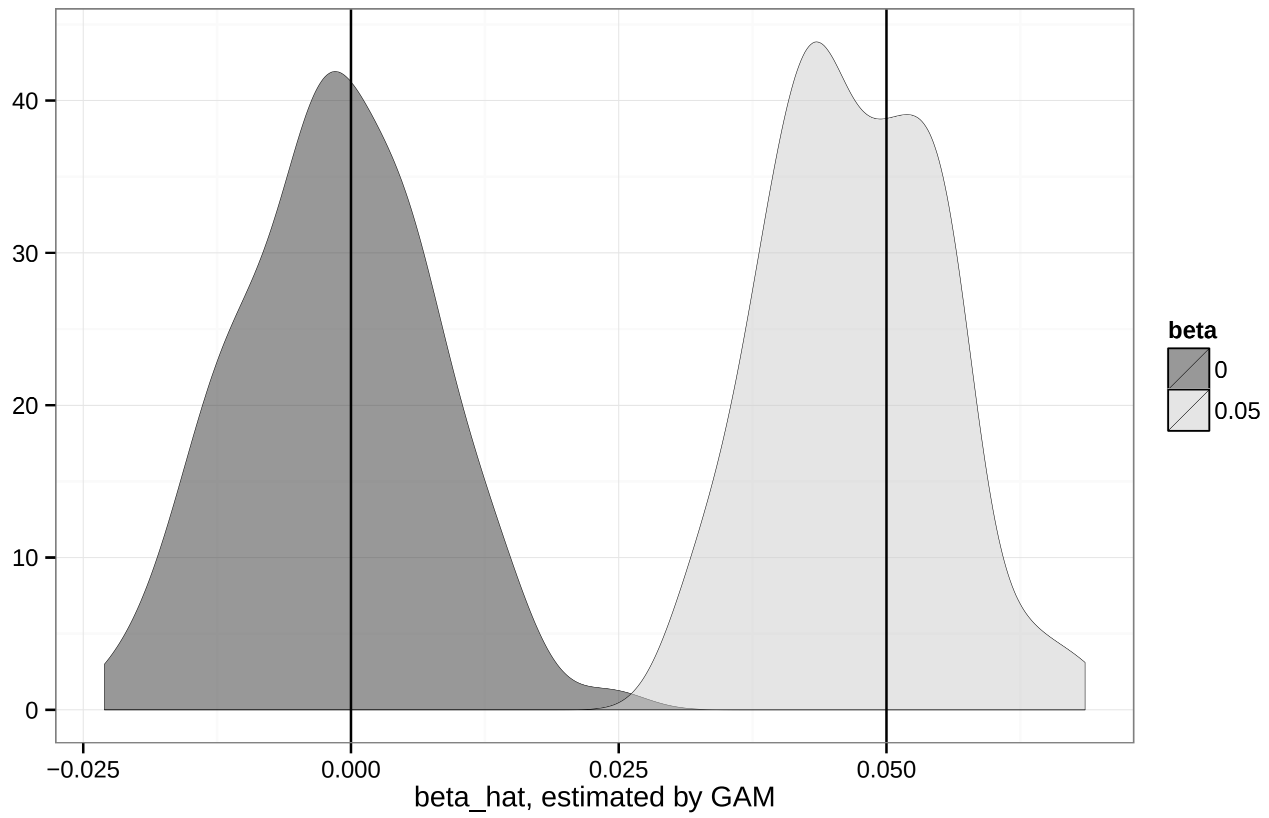
```
## No id variables; using all as measure variables
```

```
colnames(gam_mc_data)<-c("beta", "coefficient")
levels(gam_mc_data$beta)<-c("0", "0.05")
levels(gam_mc_data)
```

```
## NULL
```

Plotting the results, we see that the GAM model does a substantially better job of getting close to the "true" value.

```
ggplot(gam_mc_data, aes(x=coefficient, fill=beta)) +
geom_density(alpha=.5, size=0) +
geom_vline(aes(xintercept=c(0, 0.05))) +
xlab("beta_hat, estimated by GAM") +
ylab("") +
scale_fill_grey() +
theme_bw()
```


40
30
20
10
0
−0.025
0.000
0.025
0.050
beta_hat, estimated by GAM
beta
0
0.05