Nonparametric Analysis of US Dairy Production and Consumption

GAM model

Teo Bucci* Filippo Cipriani† Gabriele Corbo‡ Andrea Puricelli§

2023-02-17

Contents

1 Load libraries and data

```
library(pbapply)
library(mgcv)
library(conformalInference)
library(ggplot2)
library(progress)
library(parallel)
```

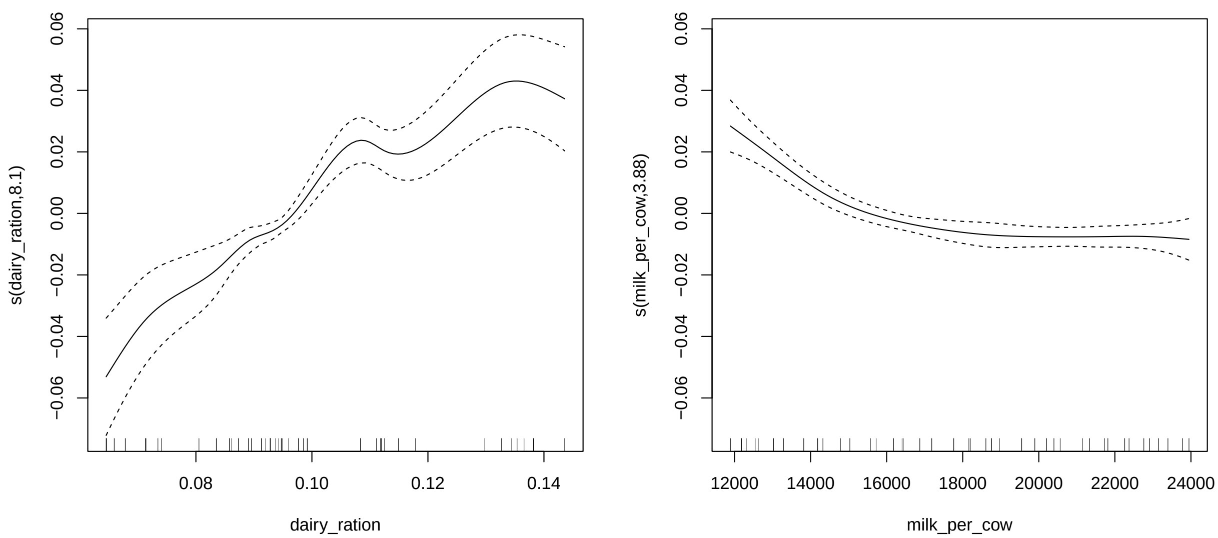
```
data_path = file.path('data_updated_2021')
output_path = file.path('output')
data_infl <-read.table(
        file.path(data_path, 'production_facts_inflated.csv'),
        header = T,
        sep = ';'
    )
```

*teo.bucci@mail.polimi.it
†filippo.cipriani@mail.polimi.it
‡gabriele.corbo@mail.polimi.it
§andrea3.puricelli@mail.polimi.it

2 Model

```
model_gam = gam(
    avg_price_milk ~ s(dairy_ration, bs = 'cr')
    + milk_cow_cost_per_animal + milk_volume_to_buy_cow_in_lbs
    + milk_feed_price_ratio + s(milk_per_cow, bs = 'cr'),
    data = data_infl
)
```

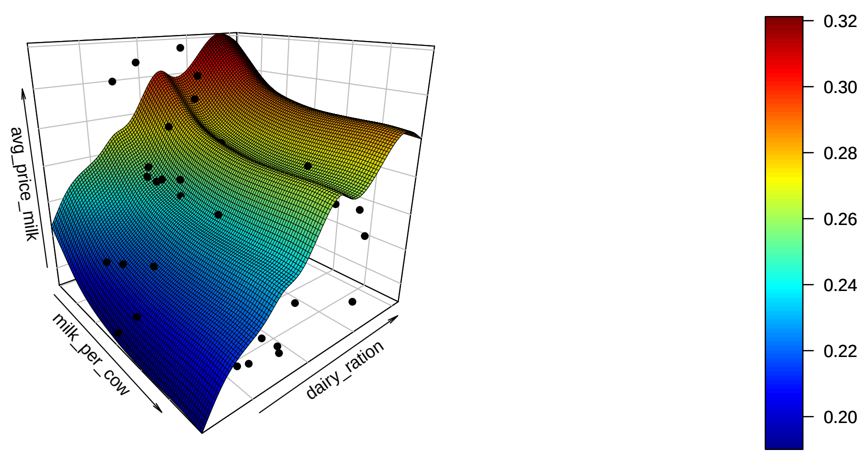
```
par(mfrow = c(1,2))
plot(model_gam)
```




```
milk_per_cow.grid=seq(range(data_infl$milk_per_cow)[1],
                      range(data_infl$milk_per_cow)[2],length.out = 100)
dairy_ration.grid=seq(range(data_infl$dairy_ration)[1],
                      range(data_infl$dairy_ration)[2],length.out = 100)
grid = expand.grid(
    milk_per_cow = milk_per_cow.grid,
    dairy_ration = dairy_ration.grid,
    milk_feed_price_ratio = mean(data_infl$milk_feed_price_ratio),
    milk_cow_cost_per_animal = mean(data_infl$milk_cow_cost_per_animal),
    milk_volume_to_buy_cow_in_lbs = mean(data_infl$milk_volume_to_buy_cow_in_lbs)
)
pred_gam = predict(model_gam, newdata = grid)
```

```
plot3D::persp3D(
  x=milk_per_cow.grid,
  y=dairy_ration.grid,
  z=matrix(pred_gam, nrow=length(milk_per_cow.grid), ncol=length(dairy_ration.grid)),
  col.palette = heat.colors,
```

```
  #xlim = range(data_infl$milk_per_cow),
  xlab = 'milk_per_cow',
  ylab = 'dairy_ration',
  zlab = 'avg_price_milk',
  box = TRUE,
  #contour = TRUE,
  border='black',
  lwd=0.1,
  shade=0.1,
  bty="b2", # https://rdrr.io/cran/plot3D/man/perspbox.html
  phi = 20, theta = 50
)

with(
  data_infl,
  plot3D::points3D(
    milk_per_cow,
    dairy_ration,
    avg_price_milk,
    col = 'black',
    size = 1,
    pch=16,
    add=TRUE
  )
)
```



3 Coefficients

```r
tab = summary(model_gam)
format(as.data.frame(tab$p.coeff), scientific = FALSE)
```

```
##                                      tab$p.coeff
## (Intercept)                       0.115131628796
## milk_cow_cost_per_animal          0.000039400099
## milk_volume_to_buy_cow_in_lbs -0.000008403757
## milk_feed_price_ratio             0.047613147536
```

```r
as.data.frame(tab$s.table)
```

```
##                       edf   Ref.df         F      p-value
## s(dairy_ration) 8.096756 8.674118  6.970188 5.701373e-05
## s(milk_per_cow) 3.877111 4.724961 13.326653 4.405487e-06
```

4 Bootstrap interval on response

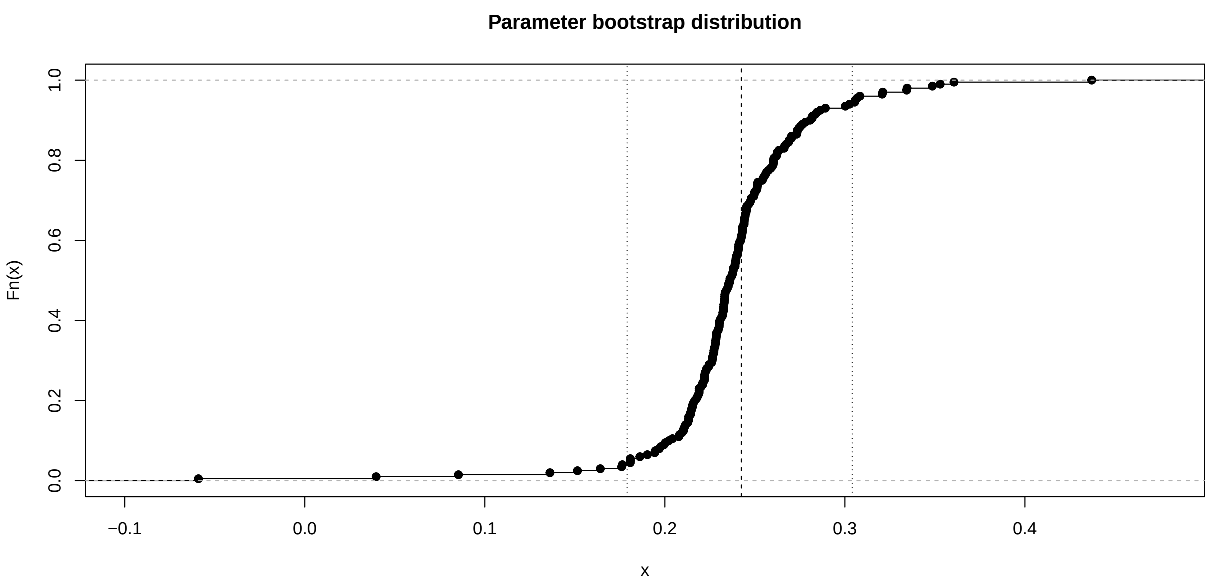
Taking into consideration the values of December, January and February of the covariates, we perform three bootstrap intervals on the prediction of the milk price, one for each month.

```r
milk_cow = c(1526.43,1531.21,1436.44)
dairy_rat = c(0.12308,0.12732,0.11571)
milk_feed = c(2.467,2.311,2.161)
milk_per_cow.med <- median(data_infl$milk_per_cow)
milk_volume_to_buy_cow_in_lbs.med <- median(data_infl$milk_volume_to_buy_cow_in_lbs)
```

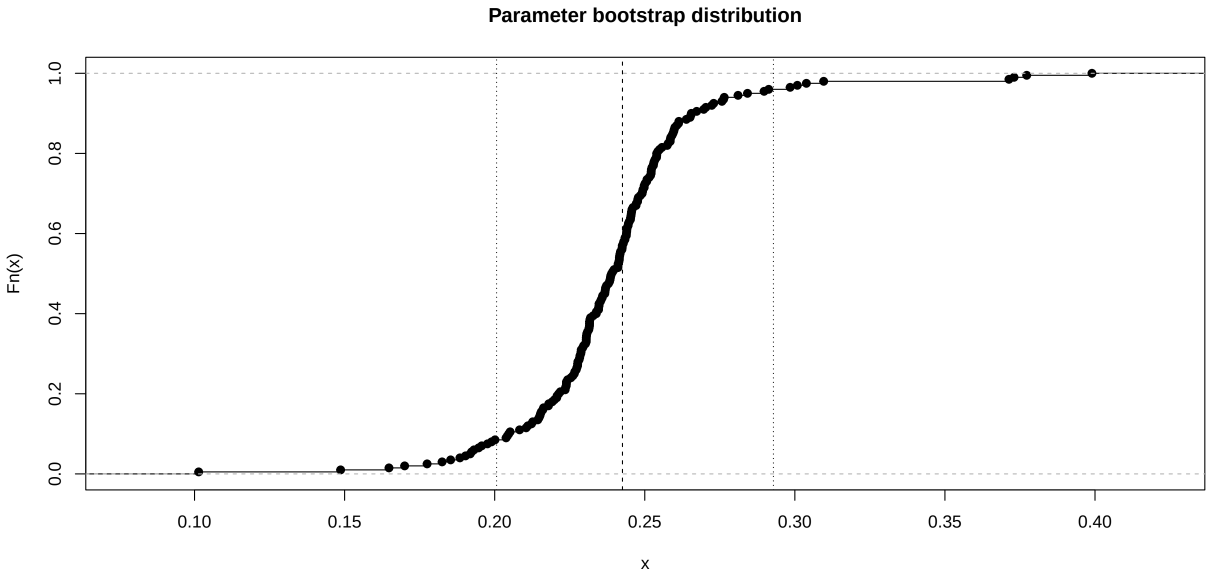
```r
CI <- matrix(0,3,3)
set.seed(1)
for(i in 1:3){
  newdata <-data.frame(milk_per_cow=milk_per_cow.med,
                       dairy_ration=dairy_rat[i],
                       milk_feed_price_ratio=milk_feed[i],
                       milk_cow_cost_per_animal=milk_cow[i],
                       milk_volume_to_buy_cow_in_lbs=milk_volume_to_buy_cow_in_lbs.med)
  B = 200
  fitted.obs <- predict(model_gam)
  res.obs <- data_infl$avg_price_milk - fitted.obs
  pred.obs = predict(model_gam, newdata = newdata)
  T.boot <- numeric(B)
  library(progress)
  pb <- progress_bar$new(
    format = "  processing [:bar] :percent eta: :eta",
    total = B, clear = FALSE)
  for (b in 1:B) {

    perm <- sample(1:nrow(data_infl), replace = T)
    dataset.boot = data_infl[perm,]

    model_gam_reduced.boot =
      mgcv::gam(avg_price_milk ~ s(dairy_ration, bs = 'cr')
```

```
                + milk_cow_cost_per_animal
                + milk_volume_to_buy_cow_in_lbs
                + milk_feed_price_ratio
                + s(milk_per_cow, bs = 'cr'), data = dataset.boot)

    T.boot[b] <- predict(model_gam_reduced.boot, newdata = newdata)
    pb$tick()
  }
  inter <- diagnostic_bootstrap(distro = T.boot, obs = pred.obs)
  CI[i,] <- inter
}
```

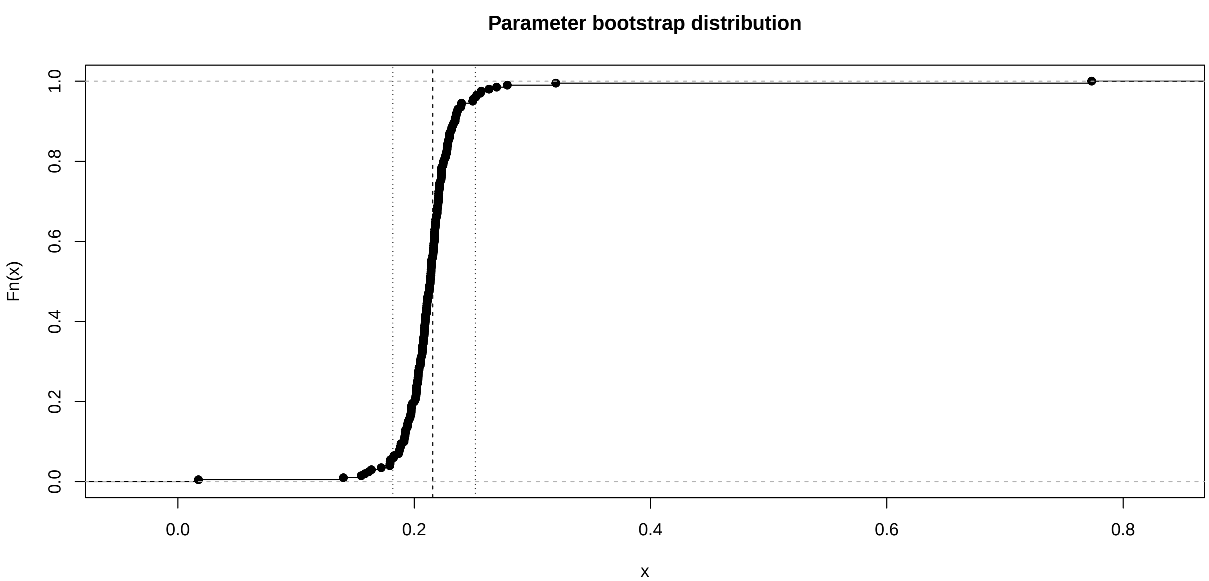
```
## [1] "Standard deviation:  0.0453288970527418"
## [1] "Bias:  -0.00444676719374543"
##       lwr       lvl       upr
## 0.1790048 0.2424174 0.3040295
```



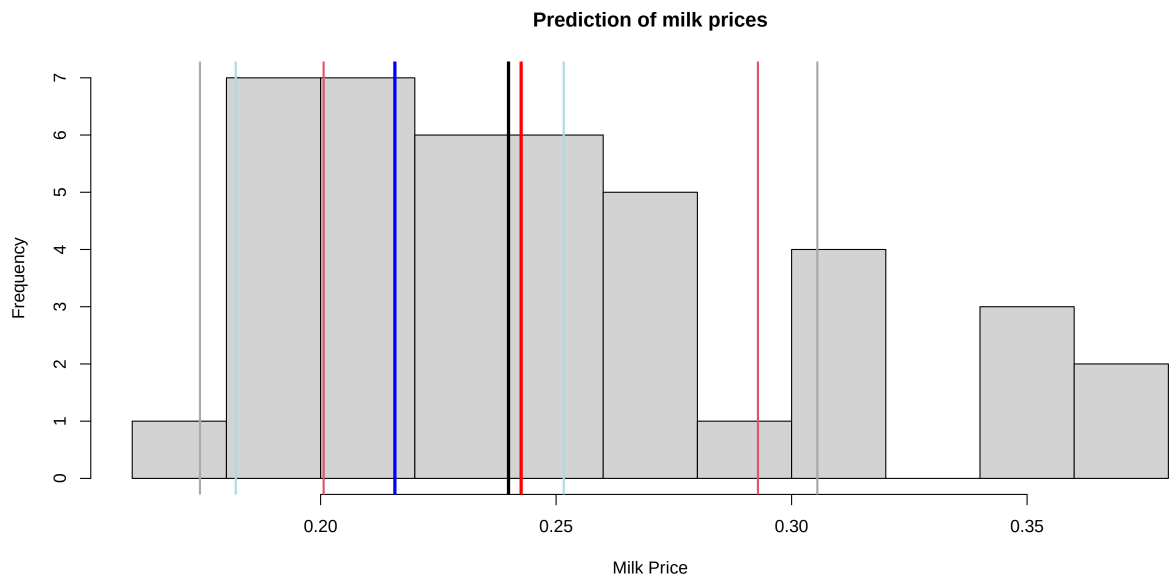

```
## [1] "Standard deviation:  0.0329405377967974"
## [1] "Bias:  -0.00299887213194591"
##       lwr       lvl       upr
## 0.2006431 0.2425718 0.2928557
```



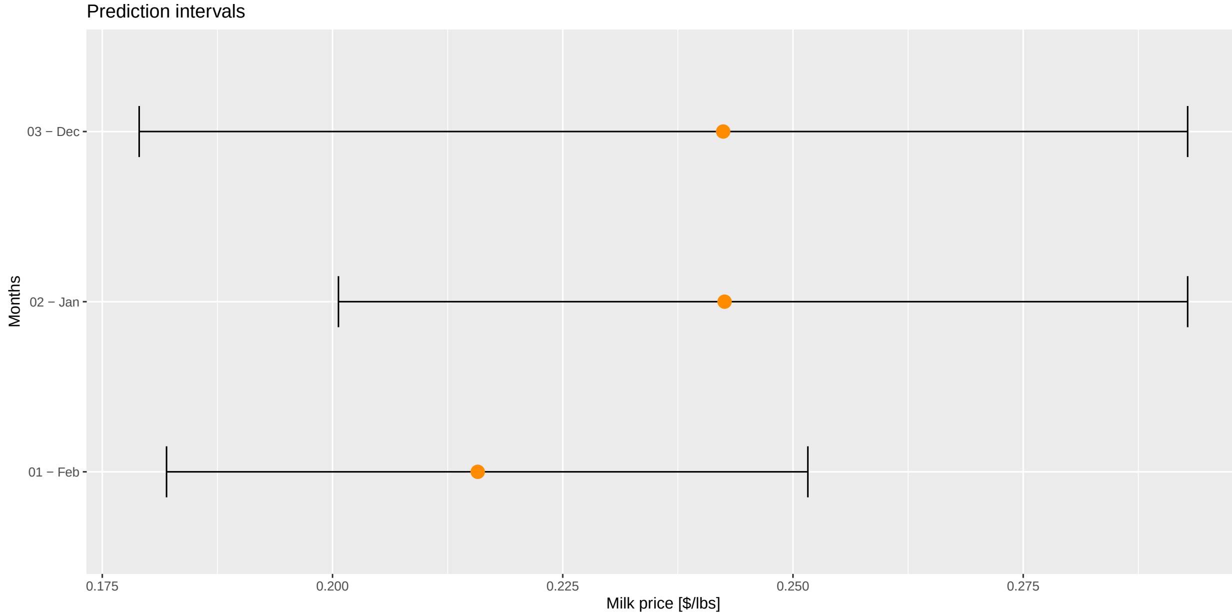

```
## [1] "Standard deviation:  0.0466987883336541"
## [1] "Bias:  -0.000829885741253006"
##       lwr       lvl       upr
## 0.1819760 0.2157682 0.2516137
```



and we compare them:




```
L = c(0.1790048 ,0.2006431 ,0.1819760 )
U = c(0.2928557 ,0.2928557 ,0.2516137 )
y = c(0.2424174 ,0.2425718 ,0.2157682 )
x=c("03 - Dec","02 - Jan","01 - Feb")
df = data.frame(x=x, y =y)

ggplot(df, aes(x = x, y = y)) +
    geom_errorbar(aes(ymax = U, ymin = L), width = 0.3) +
    geom_point(size = 4, col = "darkorange") +
    coord_flip() +
    labs(x = "Months",
         y = "Milk price [$/lbs]",
         title = "Prediction intervals")
```



5 Conformal Prediction

Using the `conformal.pred` function, it's possible to give a prediction and a conformal prediction interval on the price of the milk, considering fixed all variables except `milk_per_cow`.

The other 4 covariates are fixed to specified values.

```
newdata <- c(milk_per_cow=0,dairy_ration=0.097,milk_feed_price_ratio=2.01,
             milk_cow_cost_per_animal=2037,milk_volume_to_buy_cow_in_lbs=10000)
milk_per_cow.grid=seq(range(data_infl$milk_per_cow)[1],
                      range(data_infl$milk_per_cow)[2],length.out = 100)
```

```
wrapper_milk_per_cow=function(grid_point){
  newdata_t <- newdata
  newdata_t[1] <- grid_point
  alpha=0.1
  n_grid = 200
  c_preds = conformal.pred(
    cbind(
      data_infl$milk_per_cow,
      data_infl$dairy_ration,
      data_infl$milk_feed_price_ratio,
      data_infl$milk_cow_cost_per_animal,
      data_infl$milk_volume_to_buy_cow_in_lbs
    ),
    data_infl$avg_price_milk,
    newdata_t,
    alpha = alpha,
    verbose = T,
    train.fun = train_gam ,
    predict.fun = predict_gam,
    num.grid.pts = n_grid
```

```r
  )
  inter<-c("LOWER" = c_preds$lo,
           "PRED" = c_preds$pred,
           "UPPER" = c_preds$up)
  return(inter)
}
```

```r
n_cores <- detectCores()
cl = makeCluster(n_cores)
invisible(clusterEvalQ(cl, library(DepthProc)))
clusterExport(cl, varlist = list("milk_per_cow.grid","wrapper_milk_per_cow",
                                  "newdata","data_infl","gam","predict.gam",
                                  "conformal.pred", "train_gam", "predict_gam"))
set.seed(1)
inter=pbsapply(milk_per_cow.grid,wrapper_milk_per_cow, cl = cl)
stopCluster(cl)
```

```r
plot(milk_per_cow.grid,inter[2,],type='l',ylim=c(0.15,0.26))
points(milk_per_cow.grid,inter[1,],col=2,type='l')
points(milk_per_cow.grid,inter[3,],col=3,type='l')
```