Bilag 6

R-Koder til brug for benchmarking af drikkevandselskaber

August 2018

Konkurrence- og Forbrugerstyrelsen

Bilag 6 – R-Koder til brug for benchmarking af drikkevandsselskaber

Konkurrence- og Forbrugerstyrelsen
Forsyningssekretariatet
Carl Jacobsens Vej 35
2500 Valby
Tlf.: +45 41 71 50 00
E-mail: kfst@kfst.dk

Bilag 6 til Totaløkonomisk benchmarking for drikkevandsselskaber – R-Koder til brug for benchmarking af drikkevandsselskaber er udarbejdet af
Forsyningssekretariatet.

August 2018

Indhold

Kapitel 1
Korrigerede netvolumenmål

1.1 R-koder til brug for beregningen de to korrektioner af netvolumenmålene

Dette kapitel viser koderne, som bliver brugt til at lave de korrigerede netvolumenmål. Koderne laver regressionerne, der ligger til grund for beregningen af de alderskorrigerede og tæthedskorrigerede netvolumenmål. I *Bilag 2 – Beregning af de korrigerede netvolumenmål* fremgår resultaterne af regressionerne sammen med en forklaring på metoden.

1.2 Alderskorrigeret netvolumenmål

```
1.    ## Indlæsning af datasæt
2.    Data <- "Data fil indeholdende alder, FADO, finansielle +
3.    omkostninger og OPEX- og CAPEX Netvolumenmål"
4.
5.    Data_OPEX <- Data #Indlæser datasæt til brug i OPEX
6.    Data_CAPEX <- Data #Indlæser datasæt til brug i CAPEX
7.
8.
9.    ## I koden bruges følgende forkortelser for variable
10.
11.   # FADO = "Faktiske driftsomkostninger"
12.   # OPEXNet = "OPEX Netvolumenmål"
13.   # CAPEXNet = "CAPEX Netvolumenmål"
14.   # OPEX_Pr_Net = "Udregnes som FADO over OPEX Netvolumen"
15.   # CAPEX_Pr_Net = "Udregnes som anlægsomkostninger over CAPEX
16.   # Netvolumen"
17.   # Alder = "Aldersmålet"
18.
19.
20.
21.   #=====================================================#
22.   #### Alderskorrigeret OPEX netvolmuenmål          ####
23.   #=====================================================#
24.
25.   OPEX_alder <- lm(Data_OPEX$OPEX_pr_net ~ Data_OPEX$Alder)
26.
27.   # Finder mulige outliers med Cook' Distance
28.   j <- max(cooks.distance(OPEX_alder))
29.
30.   # Her laves en itterativ outlier analyse, hvor de selskaber
31.   # med en høj cooks distance værdi analyseres for at
32.   # vurdere om det er en outlier
33.
34.   # Vandfællesskabet Nordvest fjernes som outlier
35.   Data_OPEX <- Data_OPEX[-c(which(Data_OPEX$ID == "V202")),]
36.
```

```
# Endelig model regression for alderskorrigeret CAPEX +
# netvolmenmål

OPEX_alder <- lm(Data_OPEX$OPEX_pr_net ~ Data_OPEX$Alder)
summary(OPEX_alder)



#=====================================================#
#### Alderskorrigeret CAPEX netvolmuenmål         ####
#=====================================================#

CAPEXPEX_alder <- lm(Data_CAPEX$cAPEX_pr_net ~
Data_caPEX$Alder)

# Finder mulige outliers med Cook' Distance
j <- max(cooks.distance(cAPEX_alder))

# Her laves en itterativ outlier analyse, hvor de selskaber
# med en høj cooks distance værdi analyseres for at
# vurdere om det er en outlier

# Vandfællesskabet Nordvest fjernes som outlier
Data_cAPEX <- Data_CAPEX[-c(which(Data_CAPEX$ID
== "V202")),]


# Endelig model regression for alderskorrigeret CAPEX
# netvolmenmål

CAPEX_alder <- lm(Data_CAPEX$CAPEX_pr_net ~
Data_CAPEX$Alder)
summary(CAPEX_alder)
```

1.3 Tæthedskorrigeret netvolumenmål

```
# Indlæsning af datasæt
Data <- "Data fil indeholdende alder, FADO,
finansielleomkostninger og OPEX- og CAPEX Netvolumenmål"

Data_OPEX <- Data #Indlæser datasæt til brug i OPEX
Data_CAPEX <- Data #Indlæser datasæt til brug i CAPEX


# I koden bruges følgende forkortelser

FADO = "Faktiske driftsomkostninger"
OPEXNet = "OPEX Netvolumenmål"
CAPEXNet = "CAPEX Netvolumenmål"
OPEX_Pr_Net = "Udregnes som FADO over OPEX Netvolumen"
CAPEX_Pr_Net = "Udregnes som anlægsomkostninger over CAPEX
Netvolumen"
Tæt = "Tæthedsmålet"
```

```
87.
88.
89.   #=====================================================#
90.   #### Tæthedskorrigeret OPEX netvolmuenmål         ####
91.   #=====================================================#
92.
93.   OPEX_Tæt <- lm(Data_OPEX$OPEX_pr_net ~ Data_OPEX$Tæt)
94.
95.   #Finder mulige outliers med Cook' Distance
96.   j <- max(cooks.distance(OPEX_Tæt))
97.
98.   # Her laves en itterativ outlier analyse, hvor de selskaber
99.   # med en høj cooks distance værdi analyseres for at
100.  # vurdere om det er en outlier
101.
102.  # Ingen outliers
103.  Data_OPEX <- Data_OPEX
104.
105.
106.  # Endelig model regression for alderskorrigeret CAPEX
107.  # netvolmenmål
108.
109.  OPEX_Tæt <- lm(Data_OPEX$OPEX_pr_net ~ Data_OPEX$Tæt)
110.  summary(OPEX_Tæt)
111.
112.
113.  #=====================================================#
114.  #### Tæthedskorrigeret CAPEX netvolmuenmål        ####
115.  #=====================================================#
116.
117.  CAPEXPEX_Tæt <- lm(Data_CAPEX$cAPEX_pr_net ~ Data_caPEX$Tæt)
118.
119.  # Finder mulige outliers med Cook' Distance
120.  j <- max(cooks.distance(cAPEX_Tæt))
121.
122.  # Her laves en itterativ outlier analyse, hvor de selskaber
123.  # med en høj cooks distance værdi analyseres for at
124.  # vurdere om det er en outlier
125.
126.  # Vandfællesskabet Nordvestsjælland fjernes som outlier
127.  Data_CAPEX <- Data_CAPEX[-c(which(Data_CAPEX$ID ==
128.  "V202")),]
129.
130.
131.  # Endelig model regression for alderskorrigeret CAPEX
132.  netvolmenmål
133.  CAPEX_Tæt <- lm(Data_CAPEX$CAPEX_pr_net ~ Data_CAPEX$Tæt)
134.  summary(CAPEX_Tæt)
```

Kapitel 2
Bencmarking

2.1 R-koder til brug for DEA og SFA modellen

Dette kapitel viser R-koder for DEA og SFA benchmarking. Koderne bruges til at fastsætte fronterne og beregne efficiensscorer for selskaberne. I *Bilag 3 – Fronterne i DEA og SFA* beskrives fremgangsmetoden for fastsættelse af fronterne.

2.2 DEA

```
135.  # Indlæser pakke "Benchmarking", for at kunne bruge "sfa"
136.  # og "dea"
137.  library(Benchmarking)
138.
139.  # Indlæsning af datasæt, indeholdene alle Netvolumenmål og
140.  # FATO
141.  Data <- "data lagres her"
142.
143.  #### Datasæt til Fronten
144.
145.  # Kalundborg Overfladevand, København og Sjælsø Vand fjernes
146.  # fra fronten
147.
148.  Front_Data <- Data
149.  Front_Data <- Front_Data[-c(which(Front_Data$ID ==
150.  "V111")),]
151.  Front_Data <- Front_Data[-c(which(Front_Data$ID ==
152.  "V202")),]
153.  Front_Data <- Front_Data[-c(which(Front_Data$ID ==
154.  "V091")),]
155.
156.
157.  # Definerer variable til brug for at sætte fronten
158.
159.  Input_Front_DEA <- Front_Data$FATO
160.  Output_Front_DEA <- cbind(Front_Data$OPEX,
161.  Front_Data$OPEX_Tæt,
162.  Front_Data$CAPEX, Front_Data$CAPEX_Alder,
163.  Front_Data$CAPEX_Tæt)
164.
165.
166.  # Laver variable til brug for beregning af efficiensscore
167.  # hvor der er taget højde for særlige forhold
168.
169.  Input <- Data$FATOsf
170.  Output <- cbind(Data$OPEX, Data$OPEX_Tæt, Data$CAPEX,
171.  Data$CAPEX_Alder,
172.  Data$CAPEX_Tæt)
173.
```

```
174.
175.  #Beregning af DEA modellen
176.
177.  A <- dea(Input, Output, RTS="crs", FAST = TRUE,
178.  ORIENTATION = "in",
179.  XREF = Input_Front_DEA, YREF = Output_Front_DEA)
180.  summary(A)
```

2.3 SFA

```
181.  # Indlæsning af datasæt med alle Netvolumenmål og FATO
182.  Data <- "data lagres her"
183.
184.
185.  #### Kode til fastholdelse af front i SFA
186.  # For at kunne fastholde en front i SFA, skal følgende kode
187.  # køres inden SFA beregningerne påbegyndes
188.
189.  te.sfa_PB_all <- function (object,x_all,y_all){
190.    lambda <- object$lambda
191.    s2 <- object$sigma2
192.    coefficients <- as.matrix(object$coef,ncol=1)
193.    n_obs_all <- dim(x_all)[1]
194.    x_constant_all <- as.matrix(cbind(rep(1,n_obs_all),x_all))
195.    #colnames(x_constant_all)[1]<-"(intercepts)"
196.    #colnames(x_constant_all)<-NULL
197.    residuals_all <- y_all-   x_constant_all %*%
198.   t(coefficients)[1,]
199.    ustar <- -residuals_all * lambda^2/(1 + lambda^2)
200.    sstar <- lambda/(1 + lambda^2) * sqrt(s2)
201.    TE = pnorm(ustar/sstar - sstar)/pnorm(ustar/sstar) *
202.   exp(sstar^2/2 - ustar)
203.    colnames(TE) <- "te"
204.    return(array(TE))
205.  }
206.
207.
208.
209.  #================================================#
210.  #### SFA med ukorrigerede netvolumenmål        ####
211.  #================================================#
212.
213.  #Indlæser data
214.  Front_Ukorr <- Data
215.
216.
217.  # For at finde eventuelle outliers beregnes Cook's distance
218.
219.  outlier_Ukorr <- lm(log(Front_Ukorr$FATO) ~
220.  log(Front_Ukorr$OPEX) +
221.  log(Front_Ukorr$CAPEX))
222.  j <- max(cooks.distance(outlier_Ukorr))
223.
224.
```

```
225.  # Selskaber med høje Cook's distance analyseres for hvorvidt
226.  # de bør fjernes fra frontsættelsen
227.  # På baggrund af dette, fjerens de selskaber som ikke skal
228.  # være med i frontsættelsen
229.
230.  Front_Ukorr <- Data
231.  Front_Ukorr <- Front_Ukorr[-c(which(Front_Ukorr$ID ==
232.  "V111")),]
233.  Front_Ukorr <- Front_Ukorr[-c(which(Front_Ukorr$ID ==
234.  "V202")),]
235.  Front_Ukorr <- Front_Ukorr[-c(which(Front_Ukorr$ID ==
236.  "V164")),]
237.  Front_Ukorr <- Front_Ukorr[-c(which(Front_Ukorr$ID ==
238.  "V091")),]
239.
240.
241.
242.  # Laver input og output til fastsættelse af SFA-fronten med
243.  # de ukorrigerede
244.  # netvolumenmål
245.
246.  Input_Front_SFA_ukorr <- Front_Ukorr$FATO
247.  Output_Front_SFA_ukorr <- cbind(Front_Ukorr$OPEX,
248.  Front_Ukorr$CAPEX)
249.
250.
251.  # Laver SFA-fronten med de ukorrigerede netvolumenmål
252.
253.  O1 <- sfa(-log(Output_Front_SFA_ukorr),-
254.  log(Input_Front_SFA_ukorr))
255.
256.  # Laver input og output til beregning af efficiensscorer
257.  # med de ukorrigerede netvolumenmål, hvor der er taget højde
258.  # for særlige forhold
259.
260.  Input_ukorr <- Data$FATO
261.  Output_ukorr <- cbind(Data$OPEX, Data$CAPEX)
262.
263.
264.  # Beregner SFA-scorerne med de ukorrigerede netvolumenmål
265.
266.  B1 <- te.sfa_PB_all(O1,-log(Output_ukorr),-log(Input_ukorr))
267.  mean(B1)
268.
269.
270.
271.
272.  #=============================================#
273.  #### SFA med ALDERskorrigerede netvolumenmål ####
274.  #=============================================#
275.
276.  # Indlæser data
277.  Front_alder <- Data
278.
279.
280.  # For at finde eventuelle outliers beregnes
```

```
# Cook's distance mål

outlier_Alder <- lm(log(Front_alder$FATO) ~
log(Front_alder$OPEX) + log(Front_alder$CAPEX_Alder))
j <- max(cooks.distance(outlier_Alder))


# Selskaber med høje Cook's distance analyseres for hvorvidt
# de bør fjernes fra frontsættelsen
# På baggrund af dette, fjerens de selskaber som ikke skal
# være med i frontsættelsen

Front_alder <- Data
Front_alder <- Front_alder[-c(which(Front_alder$ID ==
"V111")),]
Front_alder <- Front_alder[-c(which(Front_alder$ID ==
"V202")),]
Front_alder <- Front_alder[-c(which(Front_alder$ID ==
"V164")),]
Front_alder <- Front_alder[-c(which(Front_alder$ID ==
"V091")),]


# Laver input og output til fastsættelse af SFA-fronten med
# de alderskorrigerede netvolumenmål

Input_Front_SFA_alder <- Front_alder$FATO
Output_Front_SFA_alder <- cbind(Front_alder$OPEX,
Front_alder$CAPEX_Alder)

# Laver SFA-fronten med de alderskorrigerede netvolumenmål

O2 <- sfa(-log(Output_Front_SFA_alder),-
log(Input_Front_SFA_alder))


# Laver input og output til beregning af efficiensscorer
# med de alderskorrigerede netvolumenmål, hvor der er taget
# højde for særlige forhold

Input <- Data$FATOsf
Output <- cbind(Data$OPEX, Data$CAPEX_Alder)


# Beregner SFA-scorerne med de ukorrigerede netvolumenmål

B2 <- te.sfa_PB_all(O2,-log(Output),-log(Input))



#=============================================#
#### SFA med Tæthedskorrigerede netvolumenmål ####
#=============================================#

```

```
337.  # Indlæser data
338.  Front_tæthed<-Data
339.
340.
341.  # For at finde eventuelle outliers, køres nedestående som
342.  # beregner Cook's distance mål
343.  outlier_Tæthed <- lm(log(Front_tæthed$FATO) ~
344.  log(Front_tæthed$OPEX_Tæt) + log(Front_tæthed$CAPEX_Tæt))
345.  j <- max(cooks.distance(outlier_Tæthed))
346.
347.
348.  # Selskaber med høje Cook's distance analyseres for hvorvidt
349.  # de bør fjernes fra frontsættelsen
350.  # På baggrund af dette, fjerens de selskaber som ikke skal
351.  # være med i frontsættelsen
352.
353.  Front_tæthed <- Data
354.  Front_tæthed <- Front_tæthed[-c(which(Front_tæthed$ID ==
355.  "V111")),]
356.  Front_tæthed <- Front_tæthed[-c(which(Front_tæthed$ID ==
357.  "V202")),]
358.  Front_tæthed <- Front_tæthed[-c(which(Front_tæthed$ID ==
359.  "V164")),]
360.  Front_tæthed <- Front_tæthed[-c(which(Front_tæthed$ID ==
361.  "V091")),]
362.
363.
364.
365.  # Laver input og output til fastsættelse af SFA-fronten med
366.  # de Tæthedskorrigerede netvolumenmål
367.
368.  Input_Front_SFA_Tæt <- Front_tæthed$FATO
369.  Output_Front_SFA-Tæt <- cbind(Front_tæthed$OPEX_Tæt,
370.  Front_tæthed$CAPEX_Tæt)
371.
372.
373.  # Laver SFA-fronten med de Tæthedskorrigerede netvolumenmål
374.
375.  O3 <- sfa(-log(Output_Front_SFA_Tæt),-
376.  log(Input_Front_SFA_Tæt))
377.
378.
379.  # Laver input og output til beregning af efficiensscorer
380.  # med de Tæthedskorrigerede netvolumenmål, hvor der er taget
381.  # højde for særlige forhold
382.
383.  Input <- Data$FATO
384.  Output <- cbind(Data$OPEX_Tæt, Data$CAPEX_Tæt)
385.
386.
387.  # Beregner SFA-scorerne med de Tæthedskorrigerede
388.  # netvolumenmål
389.
390.  B3 <- te.sfa_PB_all(O3,-log(Output),-log(Input))
391.
392.
```

```
393.
394.
395.  #================================================#
396.  #### Beregning af efficiensscore for selskaber####
397.  #### der ikke sætter fronten                  ####
398.  #================================================#
399.
400.
401.  #### Ukorrigeret netvolumenmål
402.
403.  ## Sjælsø (V164)
404.
405.  Front_Ukorr_Sjælsø <- Data
406.  Front_Ukorr_Sjælsø <- Front_Ukorr_Sjælsø[-
407.  c(which(Front_Ukorr_Sjælsø$ID == "V111")),]
408.  Front_Ukorr_Sjælsø <- Front_Ukorr_Sjælsø[-
409.  c(which(Front_Ukorr_Sjælsø$ID == "V202")),]
410.  Front_Ukorr_Sjælsø <- Front_Ukorr_Sjælsø[-
411.  c(which(Front_Ukorr_Sjælsø$ID == "V091")),]
412.
413.
414.  # Laver input og output til fastsættelse af SFA-fronten med
415.  # de ukorrigerede netvolumenmål
416.
417.  Input_Front_SFA_ukorr_Sjælsø <- Front_Ukorr_Sjælsø$FATO
418.  Output_Front_SFA_ukorr_Sjælsø <-
419.  cbind(Front_Ukorr_Sjælsø$OPEX, Front_Ukorr_Sjælsø$CAPEX)
420.  O1_Sjælsø <- sfa(-log(Output_Front_SFA_ukorr_Sjælsø),-
421.  log(Input_Front_SFA_ukorr_Sjælsø))
422.  B1_Sjælsø <- te.sfa_PB_all(O1_Sjælsø, -log(Output_ukorr),
423.  -log(Input_ukorr))
424.
425.  B1[Data$ID=="V164"] <- B1_Sjælsø[Data$ID=="V164"]
426.
427.
428.
429.  ## Kalundborg (V111)
430.
431.  Front_Ukorr_Kalu <- Data
432.  Front_Ukorr_Kalu <- Front_Ukorr_Kalu[-
433.  c(which(Front_Ukorr_Kalu$ID == "V164")),]
434.  Front_Ukorr_Kalu <- Front_Ukorr_Kalu[-
435.  c(which(Front_Ukorr_Kalu$ID == "V202")),]
436.  Front_Ukorr_Kalu <- Front_Ukorr_Kalu [-
437.  c(which(Front_Ukorr_Kalu $ID == "V091")),]
438.
439.
440.  # Laver input og output til fastsættelse af SFA-fronten med
441.  # de ukorrigerede netvolumenmål
442.
443.  Input_Front_SFA_ukorr_Kalu <- Front_Ukorr_Kalu$FATO
444.  Output_Front_SFA_ukorr_Kalu <- cbind(Front_Ukorr_Kalu$OPEX,
445.  Front_Ukorr_Kalu$CAPEX)
446.  O1_Kalu <- sfa(-log(Output_Front_SFA_ukorr_Kalu),-
447.  log(Input_Front_SFA_ukorr_Kalu))
448.  B1_Kalu <- te.sfa_PB_all(O1_Kalu,-log(Output_ukorr),-
```

```
log(Input_ukorr))

B1[Data$ID=="V111"] <- B1_Kalu[Data$ID=="V111"]



## Nordvestsjælland (V202)

Front_Ukorr_Nord <- Data
Front_Ukorr_Nord <- Front_Ukorr_Nord[-
c(which(Front_Ukorr_Nord$ID == "V164")),]
Front_Ukorr_Nord <- Front_Ukorr_Nord[-
c(which(Front_Ukorr_Nord$ID == "V111")),]
Front_Ukorr_Nord <- Front_Ukorr_Nord[-
c(which(Front_Ukorr_Nord$ID == "V091")),]


# Laver input og output til fastsættelse af SFA-fronten med
# de ukorrigerede netvolumenmål

Input_Front_SFA_ukorr_Nord <- Front_Ukorr_Nord$FATO
Output_Front_SFA_ukorr_Nord <- cbind(Front_Ukorr_Nord$OPEX,
Front_Ukorr_Nord$CAPEX)
O1_Nord <- sfa(-log(Output_Front_SFA_ukorr_Nord),-
log(Input_Front_SFA_ukorr_Nord))
B1_Nord <- te.sfa_PB_all(O1_Nord,-log(Output_ukorr),-
log(Input_ukorr))

B1[Data$ID=="V202"] <- B1_Nord[Data$ID=="V202"]



## HOFOR Vand København (V091)

Front_Ukorr_Kbh <- Data
Front_Ukorr_Kbh <- Front_Ukorr_Kbh [-
c(which(Front_Ukorr_Kbh$ID == "V164")),]
Front_Ukorr_Kbh <- Front_Ukorr_Kbh[-
c(which(Front_Ukorr_Kbh$ID == "V111")),]
Front_Ukorr_Kbh <- Front_Ukorr_Kbh[-
c(which(Front_Ukorr_Kbh$ID == "V202")),]


# Laver input og output til fastsættelse af SFA-fronten med
# de ukorrigerede netvolumenmål

Input_Front_SFA_ukorr_Kbh <- Front_Ukorr_Kbh$FATO
Output_Front_SFA_ukorr_Kbh <- cbind(Front_Ukorr_Kbh$OPEX,
Front_Ukorr_Kbh$CAPEX)
O1_Kbh <- sfa(-log(Output_Front_SFA_ukorr_Kbh),-
log(Input_Front_SFA_ukorr_Kbh))
B1_Kbh <- te.sfa_PB_all(O1_Kbh,-log(Output_ukorr),-
log(Input_ukorr))

B1[Data$ID=="V091"] <- B1_Kbh[Data$ID=="V091"]

```

```
505.
506.  #### Alderkorrigeret netvolumenmål
507.
508.  ## Sjælsø (V164)
509.
510.  Front_alder_Sjælsø <- Data
511.  Front_alder_Sjælsø <- Front_alder_Sjælsø[-
512.  c(which(Front_alder_Sjælsø$ID == "V111")),]
513.  Front_alder_Sjælsø <- Front_alder_Sjælsø[-
514.  c(which(Front_alder_Sjælsø$ID == "V202")),]
515.  Front_alder_Sjælsø <- Front_alder_Sjælsø[-
516.  c(which(Front_alder_Sjælsø$ID == "V091")),]
517.
518.
519.  # Laver input og output til fastsættelse af SFA-fronten med
520.  # de ukorrigerede netvolumenmål
521.
522.  Input_Front_SFA_alder_Sjælsø <- Front_alder_Sjælsø$FATO
523.  Output_Front_SFA_alder_Sjælsø <-
524.  cbind(Front_alder_Sjælsø$OPEX, Front_alder_Sjælsø$CAPEX)
525.  O2_Sjælsø <- sfa(-log(Output_Front_SFA_alder_Sjælsø),-
526.  log(Input_Front_SFA_alder_Sjælsø))
527.  B2_Sjælsø <- te.sfa_PB_all(O2_Sjælsø,-log(Output_alder),-
528.  log(Input_alder))
529.
530.  B2[Data$ID=="V164"] <- B2_Sjælsø[Data$ID=="V164"]
531.
532.
533.
534.  ## Kalundborg (V111)
535.
536.  Front_alder_Kalu <- Data
537.  Front_alder_Kalu <- Front_alder_Kalu[-
538.  c(which(Front_alder_Kalu$ID == "V164")),]
539.  Front_alder_Kalu <- Front_alder_Kalu[-
540.  c(which(Front_alder_Kalu$ID == "V202")),]
541.  Front_alder_Kalu <- Front_alder_Kalu[-
542.  c(which(Front_alder_Kalu$ID == "V091")),]
543.
544.
545.  # Laver input og output til fastsættelse af SFA-fronten med
546.  # de ukorrigerede netvolumenmål
547.
548.  Input_Front_SFA_alder_Kalu <- Front_alder_Kalu$FATO
549.  Output_Front_SFA_alder_Kalu <- cbind(Front_alder_Kalu$OPEX,
550.  Front_alder_Kalu$CAPEX)
551.  O2_Kalu <- sfa(-log(Output_Front_SFA_alder_Kalu),-
552.  log(Input_Front_SFA_alder_Kalu))
553.  B2_Kalu <- te.sfa_PB_all(O2_Kalu,-log(Output_alder),-
554.  log(Input_alder))
555.
556.  B2[Data$ID=="V111"] <- B2_Kalu[Data$ID=="V111"]
557.
558.
559.
560.  ## Nordvestsjælland (V202)
```

```
561.
562.  Front_alder_Nord <- Data
563.  Front_alder_Nord <- Front_alder_Nord[-
564.  c(which(Front_alder_Nord$ID == "V164")),]
565.  Front_alder_Nord <- Front_alder_Nord[-
566.  c(which(Front_alder_Nord$ID == "V111")),]
567.  Front_alder_Nord <- Front_alder_Nord[-
568.  c(which(Front_alder_Nord$ID == "V091")),]
569.
570.
571.  # Laver input og output til fastsættelse af SFA-fronten med
572.  # de ukorrigerede netvolumenmål
573.
574.  Input_Front_SFA_alder_Nord <- Front_alder_Nord$FATO
575.  Output_Front_SFA_alder_Nord <- cbind(Front_alder_Nord$OPEX,
576.  Front_alder_Nord$CAPEX)
577.  O2_Nord <- sfa(-log(Output_Front_SFA_alder_Nord),-
578.  log(Input_Front_SFA_alder_Nord))
579.  B2_Nord <- te.sfa_PB_all(O2_Nord,-log(Output_alder),-
580.  log(Input_alder))
581.
582.  B2[Data$ID=="V202"] <- B2_Nord[Data$ID=="V202"]
583.
584.
585.
586.  ## HOFOR Vand København (V091)
587.
588.  Front_alder_Nord <- Data
589.  Front_alder_Kbh <- Front_alder_Kbh[-
590.  c(which(Front_alder_Kbh$ID == "V164")),]
591.  Front_alder_Kbh <- Front_alder_Kbh[-
592.  c(which(Front_alder_Kbh$ID == "V111")),]
593.  Front_alder_Kbh <- Front_alder_Kbh[-
594.  c(which(Front_alder_Kbh$ID == "V202")),]
595.
596.
597.  # Laver input og output til fastsættelse af SFA-fronten med
598.  # de ukorrigerede netvolumenmål
599.
600.  Input_Front_SFA_alder_Kbh <- Front_alder_Kbh$FATO
601.  Output_Front_SFA_alder_Kbh <- cbind(Front_alder_Kbh$OPEX,
602.  Front_alder_Kbh$CAPEX)
603.  O2_Kbh <- sfa(-log(Output_Front_SFA_alder_Kbh),-
604.  log(Input_Front_SFA_alder_Kbh))
605.  B2_Kbh <- te.sfa_PB_all(O2_Kbh,-log(Output_alder),-
606.  log(Input_alder))
607.
608.  B2[Data$ID=="V091"] <- B2_Kbh[Data$ID=="V091"]
609.
610.
611.
612.
613.  #### Tæthedskorrigeret netvolumenmål
614.
615.  ## Sjælsø (V164)
616.
```

```
Front_tæt_Sjælsø <- Data
Front_tæt_Sjælsø <- Front_tæt_Sjælsø[-
c(which(Front_tæt_Sjælsø$ID == "V111")),]
Front_tæt_Sjælsø <- Front_tæt_Sjælsø[
-c(which(Front_tæt_Sjælsø$ID == "V202")),]
Front_tæt_Sjælsø <- Front_tæt_Sjælsø[
-c(which(Front_tæt_Sjælsø$ID == "V091")),]


# Laver input og output til fastsættelse af SFA-fronten med
# de ukorrigerede netvolumenmål

Input_Front_SFA_tæt_Sjælsø <- Front_tæt_Sjælsø$FATO
Output_Front_SFA_tæt_Sjælsø <- cbind(Front_tæt_Sjælsø$OPEX,
Front_tæt_Sjælsø$CAPEX)
O3_Sjælsø <- sfa(-log(Output_Front_SFA_tæt_Sjælsø),-
log(Input_Front_SFA_tæt_Sjælsø))
B3_Sjælsø <- te.sfa_PB_all(O3_Sjælsø, -log(Output_tæt),
-log(Input_tæt))

B3[Data$ID=="V164"] <- B3_Sjælsø[Data$ID=="V164"]



## Kalundborg (V111)

Front_tæt_Kalu <- Data
Front_tæt_Kalu <- Front_tæt_Kalu[-c(which(Front_tæt_Kalu$ID
== "V164")),]
Front_tæt_Kalu <- Front_tæt_Kalu[-c(which(Front_tæt_Kalu$ID
== "V202")),]
Front_tæt_Kalu <- Front_tæt_Kalu[-c(which(Front_tæt_Kalu$ID
== "V091")),]


# Laver input og output til fastsættelse af SFA-fronten med
# de ukorrigerede netvolumenmål

Input_Front_SFA_tæt_Kalu <- Front_tæt_Kalu$FATO
Output_Front_SFA_tæt_Kalu <- cbind(Front_tæt_Kalu$OPEX,
Front_tæt_Kalu$CAPEX)
O3_Kalu <- sfa(-log(Output_Front_SFA_tæt_Kalu),-
log(Input_Front_SFA_tæt_Kalu))
B3_Kalu <- te.sfa_PB_all(O3_Kalu,-log(Output_tæt),-
log(Input_tæt))

B3[Data$ID=="V111"] <- B3_Kalu[Data$ID=="V111"]



## Nordvestsjælland (V202)

Front_tæt_Nord <- Data
Front_tæt_Nord <- Front_tæt_Nord[-c(which(Front_tæt_Nord$ID
== "V164")),]
Front_tæt_Nord <- Front_tæt_Nord[-c(which(Front_tæt_Nord$ID
```

```
673.  == "V111")),]
674.  Front_tæt_Nord <- Front_tæt_Nord[-c(which(Front_tæt_Nord$ID
675.  == "V091")),]
676.
677.
678.  # Laver input og output til fastsættelse af SFA-fronten med
679.  # de ukorrigerede netvolumenmål
680.
681.  Input_Front_SFA_tæt_Nord <- Front_tæt_Nord$FATO
682.  Output_Front_SFA_tæt_Nord <- cbind(Front_tæt_Nord$OPEX,
683.  Front_tæt_Nord$CAPEX)
684.  O3_Nord <- sfa(-log(Output_Front_SFA_tæt_Nord),-
685.  log(Input_Front_SFA_tæt_Nord))
686.  B3_Nord <- te.sfa_PB_all(O3_Nord,-log(Output_tæt),-
687.  log(Input_tæt))
688.
689.  B3[Data$ID=="V202"] <- B3_Nord[Data$ID=="V202"]
690.
691.
692.
693.  ## HOFOR Vand København (V091)
694.
695.  Front_tæt_Kbh <- Data
696.  Front_tæt_Kbh <- Front_tæt_Kbh[-c(which(Front_tæt_Kbh$ID
697.  == "V164")),]
698.  Front_tæt_Kbh <- Front_tæt_Kbh[-c(which(Front_tæt_Kbh$ID
699.  == "V111")),]
700.  Front_tæt_Kbh <- Front_tæt_Kbh[-c(which(Front_tæt_Kbh$ID
701.  == "V202")),]
702.
703.
704.  # Laver input og output til fastsættelse af SFA-fronten med
705.  # de ukorrigerede netvolumenmål
706.
707.  Input_Front_SFA_tæt_Kbh <- Front_tæt_Kbh$FATO
708.  Output_Front_SFA_tæt_Kbh <- cbind(Front_tæt_Kbh$OPEX,
709.  Front_tæt_Kbh$CAPEX)
710.  O3_Kbh <- sfa(-log(Output_Front_SFA_tæt_Kbh),-
711.  log(Input_Front_SFA_tæt_Kbh))
712.  B3_Kbh <- te.sfa_PB_all(O3_Kbh,-log(Output_tæt),-
713.  log(Input_tæt))
714.
715.  B3[Data$ID=="V091"] <- B3_Kbh[Data$ID=="V091"]
716.
717.
718.
719.  #### Finder den højeste SFA score ####
720.
721.  B <- rep(NA,dim(Data)[1])
722.  for(i in 1:length(B)){
723.    B[i] <- max(B1[i],B2[i],B3[i])
724.  }
```

Kapitel 3
Costdriversammensætning

3.1 R-koder til brug for analyse af costdriversammensætning

Dette kapitel viser R-koderne brugt til at lave analysen af costdriversammensætningen. I *Bilag 1 – Cosdriversammensætning* beskrives metoden for analysen samt resultaterne for regressionerne.

3.2 Kode for costdriversammensætning

```
725.  ##################################################
726.  library(car)
727.
728.  # Indlæsning af datasæt
729.  Data <- "Data fil indeholdende efficiensscorer og
730.  #OPEX netvolumenbidrag fra costdrivere og CAPEX
731.  #netvolumebridrag fra typer"
732.
733.  # Data$Score = "Best of two efficiensscore fra DEA og SFA
734.  # bencmarking"
735.
736.
737.  #================================================#
738.  ############### OPEX ##############################
739.  #================================================#
740.
741.  #### Boringer
742.
743.  # Regression
744.  model <- lm(Data$Score ~ Data$Boring)
745.
746.  # Finder mulige outliers med Cook's distance
747.  j <- max(cooks.distance(model))
748.
749.  # Modellens resultat
750.  # (Eventuelle outliers fjernes og modellen køres igen)
751.  summary(model)
752.
753.
754.
755.  #### Vandværker
756.
757.  # Regression
758.  model <- lm(Data$Score ~ Data$Vand)
759.
760.  # Finder mulige outliers med Cook's distance
761.  j <- max(cooks.distance(model))
762.
763.  # Modellens resultat
```

```
764.  # (Eventuelle outliers fjernes og modellen køres igen)
765.  summary(model)
766.
767.
768.
769.  #### Trykforøgerstationer
770.
771.  # Regression
772.  model <- lm(Data$Score ~ Data$Tryk)
773.
774.  # Finder mulige outliers med Cook's distance
775.  j <- max(cooks.distance(model))
776.
777.  # Modellens resultat
778.  # (Eventuelle outliers fjernes og modellen køres igen)
779.  summary(model)
780.
781.
782.
783.  #### Ledninger og stik
784.
785.  # Regression
786.  model <- lm(Data$Score ~ Data$Ledning_Stik)
787.
788.  # Finder mulige outliers med Cook's distance
789.  j <- max(cooks.distance(model))
790.
791.  # Modellens resultat
792.  # (Eventuelle outliers fjernes og modellen køres igen)
793.  summary(model)
794.
795.
796.
797.  #### Kunder og målere
798.
799.  # Regression
800.  model <- lm(Data$Score ~ Data$kunder)
801.
802.  # Finder mulige outliers med Cook's distance
803.  j <- max(cooks.distance(model))
804.
805.  # Modellens resultat
806.  # (Eventuelle outliers fjernes og modellen køres igen)
807.  summary(model)
808.
809.
810.
811.  #### Generel administration
812.
813.  # Regression
814.  model <- lm(Data$Score ~ Data$Debiteret_vand)
815.
816.  # Finder mulige outliers med Cook's distance
817.  j <- max(cooks.distance(model))
818.
819.  # Modellens resultat
```

```
820.  # (Eventuelle outliers fjernes og modellen køres igen)
821.  summary(model)
822.
823.
824.
825.  #### Ledninger og stik, Målere og kunder samt Generel
826.  #administration
827.
828.  # Regression
829.  model <- lm(Data$Score ~ Data$Ledning_Stik + Data$kunder +
830.   Data$Debiteret_vand)
831.
832.  # Finder mulige outliers med Cook's distance
833.  j <- max(cooks.distance(model))
834.
835.  # Modellens resultat
836.  # (Eventuelle outliers fjernes og modellen køres igen)
837.  summary(model)
838.
839.
840.
841.
842.  #================================================#
843.  ############### CAPEX ##############################
844.  #================================================#
845.
846.  #### Produktionsanlæg
847.
848.  # Regression
849.  model <- lm(Data$Score ~ Data$Produktionsanlæg)
850.
851.  # Finder mulige outliers med Cook's distance
852.  j <- max(cooks.distance(model))
853.
854.  # Modellens resultat
855.  # (Eventuelle outliers fjernes og modellen køres igen)
856.  summary(model)
857.
858.
859.
860.  #### Distributionsanlæg
861.
862.  # Regression
863.  model <- lm(Data$Score ~ Data$Distributionsanlæg)
864.
865.  # Finder mulige outliers med Cook's distance
866.  j <- max(cooks.distance(model))
867.
868.  # Modellens resultat
869.  # (Eventuelle outliers fjernes og modellen køres igen)
870.  summary(model)
871.
872.
873.
874.  #### Fællesfunktionsanlæg
875.
```

```
# Regression
model <- lm(Data$Score ~ Data$Distributionsanlæg)

# Finder mulige outliers med Cook's distance
j <- max(cooks.distance(model))

# Modellens resultat
# (Eventuelle outliers fjernes og modellen køres igen)
summary(model)
```