Danmarks Statistik
MODELGRUPPEN

Arbejdspapir*

Anne Bender og Morten Werner

29. november 2005

En kort beskrivelse af modulet til beregning af kædeaggregater i ADAM - juli 2005

Resumé:

Papirets væsentligste opgave er at oplyse brugerne om kædemodulets eksistens

ABD25N05.DOC

Nøgleord: Laspeyres kædeindeks

I ADAM juli 2005 er der i eftermodellen indbygget et modul, der beregner Laspeyres kædeaggregater af en række centrale størrelser. Baggrunden for modulet er, at nationalregnskabet i juli 2005 gik over til at rapportere fastprisstørrelser beregnet ved hjælp af kædeindeks, mens fastprisstørrelserne i ADAM - juli 2005 er baseret på fastbaseaggregater, med år 2000 som basisår.

Modulet giver brugeren mulighed for at slå fastprisstørrelser målt ved kædeindeks op i ADAMs databank og rapportere fastprisstørrelser målt ved kædeindeks i forbindelse med udarbejdelse af prognoser mm. Modulet er af midlertidig karakter, idet næste version af ADAM på fastprissiden alene vil være baseret på kædede mængdeindeks.

Fastprisstørrelser aggregeret ved kædeindeks navngives med preffix *fkl*. Således hedder fx vareimporten *fklMv*.

Et eksempel på tabelmodulets relationer kan præsenteres i tilfældet, hvor det ønskes at aggregere den erhvervsfordelte produktion målt i faste priser fX^i, hvor *i* løber over ADAMs 19 erhverv, til den samlede produktion ved hjælp af et Laspeyes kædeindeks. I dette tilfælde bliver relationen

$$fklX = fklX_{-1}\left(\frac{fX_t^1}{fX_{t-1}^1}\cdot\frac{X_{t-1}^1}{X_{t-1}}+....+\frac{fX_t^{19}}{fX_{t-1}^{19}}\cdot\frac{X_{t-1}^{19}}{X_{t-1}}\right)\cdot kfklX \qquad (1)$$

Ligning (1) viser, at Laspeyres kædeindekset aggregerer ved at sammenveje vækstraterne i aggregatets komponenter med de laggede omkostningsandele. Mens aggregering i fastbaseindekset som bekendt foretages ved summation af komponenterne.

Fra (1) ses det, at det er nødvendigt at tilføje en korrektionsfaktor, *kfklX*, til relationen. Dette skyldes, at produktionen på erhvervsniveau er målt ved fastbaseindeks. Korrektionsfaktoren vil som hovedregel være større jo længere der er mellem det betragtede år og året, der anvendes som basisår i fastbase indekset. Korrektionsfaktorerne navngives alle med preffix *kfkl*, således at fx korrektionsfaktoren i vareimporten hedder *kfklMv*.

Ved fremskrivning af korrektionsfaktorerne foreslås det at anvende værdien fra det sidste datadækkede år.

For en række af aggregaterne i kædemodulet findes ikke *k*-faktorer. Dette skyldes, at der ikke umiddelbart findes et NR-aggregat for den givne variabel, hvorfor det er umuligt at beregne korrektionsfaktoren.

En del af korrektionsfaktorerne er eksakt 1. Dette skyldes, at aggregatet alene dannes udfra kædeaggregater, der er korrigeret til at have NR-værdien[1]. Et

[1] Dette afspejler, at Laspeyres kædeindeks er robuste overfor flertrinsaggregering, altså at aggregeringen med Laspeyres kædeindeks af varerne q1, q2, q3 giver samme resultat uanset om der først dannes et underaggregat af fx varerne q1 og q2 ved hjælp af LK. Dvs

$LK(q1,q2,q3) = LK(LK(q1,q2),q3)$, hvor LK er Laspeyres kædeaggregatoren

eksempel er relationen for BNP, *fklY*. *k*-faktoren er bibeholdt i disse relationer, idet den giver et godt tjek på, om datagenereringen er gået godt.

I kædemodulet dannes forsyningsbalancekomponenterne samt en række investeringskomponenter som Laspeyres kædeaggregater. En samlet variabelliste er vist nedenfor:

Forbrug:
fklCo Offentligt forbrug
fklCp Privat forbrug i alt

Investeringer:
fklIt Investeringer i stambesætninger
fklIbp Investeringer i bygninger og anlæg i private erhverv
fklImp Investeringer i maskiner og transportmidler i private erhverv
fklIl Lagerinvesteringer i alt
fklIb Investeringer i bygninger og anlæg
fklIm Investeringer i maskiner, transportmidler og inventar
fklIp Investeringer i private erhverv
fklIo Investeringer i offentligt erhver

fklIf Faste bruttoinvesteringer i alt
Beregning LK(*fklim*, *fklib*, *fklit*)
hvor LK er laspeyres kædeaggregatoren
fklI Investeringer i alt
Beregning: LK(fklif,fklil)

Produktion
fklX Produktionsværdi i alt

Udenrigshandel
fklMv Vareimport i alt
fklMst Import af tjenester inkl. turisme
fklM Import i alt
Beregning: LK(fklMv, fklMst)

fklEv Vareeksport i alt
fklEst Tjenesteeksport i alt
fklE Eksport i alt
Beregning: LK(fklEv, fklEst)

Denne pointe er væsentlig at fremføre, fordi man ofte støder på den misforståelse at LK's manglende additivitet er ensbetyende med, at man ikke kan danne fx sine egne aggregater uden kendskab til NRs mikrodata. Det kan man. Man skal blot bruge LK-aggregatoren på underaggregaterne altså fx nationalregnskabets offentliggjorte forbrugsgrupper. Denne egenskab bærer i øvrigt over fra flertrinsaggregeringen af fastbaseindeks, således at også Paasche, men ikke Tørnqvist og Fischer kædeindeks, er robuste overfor flertrinsaggregering. Se fx Diewert "Indexnumber Theory and Measurement in economics"

Varekøb

fklV Varekøb i alt
berening : LK over erhvervenes køb af materialer og energi)

Bruttoværditilvækst

fklyf Bruttoværditilvækst i alt

BNP

fkly BNP
Beregning: LK(fklcp,fklco,fkli,-fklm,fkle)

Modulets formler er vist i appendiks.

```
() Modul til beregning af Mængdeindeks med kædemængdeindeks, udvalgte variabler
()
FRML Ifklco            fKlco            = fCo*kfklco $
FRML Ifklit            fKlit            = fIt*kfklit $
FRML IImk              IMK              = Im-Imp-Imo $
FRML IfklCp            fKlcp            = fKlcp(-1)*( fCh/fCh(-1)*Ch(-1)/Cp(-1) + fCf/fCf(-1)*Cf(-1)/Cp(-1)
                                          + fCn/fCn(-1)*Cn(-1)/Cp(-1) + fCi/fCi(-1)*Ci(-1)/Cp(-1)
                                          + fCe/fCe(-1)*Ce(-1)/Cp(-1) + fCg/fCg(-1)*Cg(-1)/Cp(-1)
                                          + fCb/fCb(-1)*Cb(-1)/Cp(-1) + fCk/fCk(-1)*Ck(-1)/Cp(-1)
                                          + fCv/fCv(-1)*Cv(-1)/Cp(-1) + fCs/fCs(-1)*Cs(-1)/Cp(-1)
                                          + fCt/fCt(-1)*Ct(-1)/Cp(-1) - fEt/fEt(-1)*Et(-1)/Cp(-1))*kfklcp  $
FRML Ifklibp           fklibp           = fklibp(-1)*(fIba/fIba(-1)*Iba(-1)/Ibp(-1) + fIbe/fIbe(-1)*Ibe(-1)/Ibp(-1)
                                          + fIbb/fIbb(-1)*Ibb(-1)/Ibp(-1) + fIbng/fIbng(-1)*Ibng(-1)/Ibp(-1)
                                          + fIbne/fIbne(-1)*Ibne(-1)/Ibp(-1) + fIbnf/fIbnf(-1)*Ibnf(-1)/Ibp(-1)
                                          + fIbnn/fIbnn(-1)*Ibnn(-1)/Ibp(-1) + fIbnb/fIbnb(-1)*Ibnb(-1)/Ibp(-1)
                                          + fIbnm/fIbnm(-1)*Ibnm(-1)/Ibp(-1) + fIbnt/fIbnt(-1)*Ibnt(-1)/Ibp(-1)
                                          + fIbnk/fIbnk(-1)*Ibnk(-1)/Ibp(-1) + fIbnq/fIbnq(-1)*Ibnq(-1)/Ibp(-1)
                                          + fIbqh/fIbqh(-1)*Ibqh(-1)/Ibp(-1)
                                          + (fIbqs+fIbqt)/(fIbqs(-1)+fIbqt(-1))*(Ibqs(-1)+Ibqt(-1))/Ibp(-1)
                                          + fIbqf/fIbqf(-1)*Ibqf(-1)/Ibp(-1) + fIbqq/fIbqq(-1)*Ibqq(-1)/Ibp(-1)) $
FRML Ifklimp           fklimp           = fklimp(-1)*(fIma/fIma(-1)*Ima(-1)/Imp(-1) + fIme/fIme(-1)*Ime(-1)/Imp(-1)
                                          + fImh/fImh(-1)*Imh(-1)/Imp(-1)    + fImb/fImb(-1)*Imb(-1)/Imp(-1)
                                          + fImng/fImng(-1)*Imng(-1)/Imp(-1) + fImne/fImne(-1)*Imne(-1)/Imp(-1)
                                          + fImnf/fImnf(-1)*Imnf(-1)/Imp(-1) + fImnn/fImnn(-1)*Imnn(-1)/Imp(-1)
                                          + fImnm/fImnm(-1)*Imnm(-1)/Imp(-1) + fImnb/fImnb(-1)*Imnb(-1)/Imp(-1)
                                          + fImnt/fImnt(-1)*Imnt(-1)/Imp(-1) + fImnk/fImnk(-1)*Imnk(-1)/Imp(-1)
                                          + fImnq/fImnq(-1)*Imnq(-1)/Imp(-1) + fImqh/fImqh(-1)*Imqh(-1)/Imp(-1)
                                          + fImqs/fImqs(-1)*Imqs(-1)/Imp(-1) + fImqt/fImqt(-1)*Imqt(-1)/Imp(-1)
                                          + fImqf/fImqf(-1)*Imqf(-1)/Imp(-1) + fImqq/fImqq(-1)*Imqq(-1)/Imp(-1)) $
FRML IfkIl             fKlil            = fKlil(-1)*( fIla*pxa(-1)/Il(-1)    + fIle*pxe(-1)/Il(-1)
                                          + fIlng*pxng(-1)/Il(-1)   + fIlne*pxne(-1)/Il(-1)
                                          + fIlnf*pxnf(-1)/Il(-1)   + fIlnn*pxnn(-1)/Il(-1)
                                          + fIlnb*pxnb(-1)/Il(-1)   + fIlnm*pxnm(-1)/Il(-1)
                                          + fIlnt*pxnt(-1)/Il(-1)   + fIlnk*pxnk(-1)/Il(-1)
                                          + fIlnq*pxnq(-1)/Il(-1)   + fIlqh*pxqh(-1)/Il(-1)
                                          + fIlqq*pxqq(-1)/Il(-1)            + fIlm0*(pm0(-1)+tm0(-1))/Il(-1)
                                          + fIlm1*(pm1(-1)+tm1(-1))/Il(-1)    + fIlm2*(pm2(-1)+tm2(-1))/Il(-1)
                                          + fIlm3r*(pm3r(-1)+tm3r(-1))/Il(-1) + fIlm3k*(pm3k(-1)+tm3k(-1))/Il(-1)
                                          + fIlm3q*(pm3q(-1)+tm3q(-1))/Il(-1) + fIlm5*(pm5(-1)+tm5(-1))/Il(-1)
                                          + fIlm6m*(pm6m(-1)+tm6m(-1))/Il(-1) + fIlm6q*(pm6m(-1)+tm6m(-1))/Il(-1)
                                          + fIlm7b*(pm7b(-1)+tm7b(-1))/Il(-1) + fIlm7q*(pm7q(-1)+tm7q(-1))/Il(-1)
                                          + fIlm7y*(pm7y(-1)+tm7y(-1))/Il(-1) + fIlm8*(pm8(-1)+tm8(-1))/Il(-1)
                                          + fIlsi*pil(-1)/Il(-1))*kfklil                $
FRML IfklIb            fklib            = fklib(-1)*(fklibp/fklibp(-1)*Ibp(-1)/(Ibp(-1)+Ibo(-1)+Ibh(-1))
                                          + fIbo/fIbo(-1)*Ibo(-1)/(Ibp(-1)+Ibo(-1)+Ibh(-1))
                                          + fIbh/fIbh(-1)*Ibh(-1)/(Ibp(-1)+Ibo(-1)+Ibh(-1))) $
FRML IfklIm            fklim            = fklim(-1)*(fklimp/fklimp(-1)*Imp(-1)/(Imp(-1)+Imo(-1)+IMK(-1))
                                          + fImo/fImo(-1)*Imo(-1)/(Imp(-1)+Imo(-1)+IMK(-1))
                                          + fImk/fImk(-1)*IMK(-1)/(Imp(-1)+Imo(-1)+IMK(-1))) $
FRML IIp               Ip               = Imp + IMK + Ibp + Ibh + It + Il $
FRML IfklIp            fklip            = fklip(-1)*(fklimp/fklimp(-1)*Imp(-1)/Ip(-1) + fImk/fImk(-1)*IMK(-1)/Ip(-1)
                                          + fklibp/fklibp(-1)*Ibp(-1)/Ip(-1) + fIbh/fIbh(-1)*Ibh(-1)/Ip(-1)
                                          + fKlit/fKlit(-1)*It(-1)/Ip(-1)    + fKlil/fKlil(-1)*Il(-1)/Ip(-1)) $
FRML IIo               Io               = Imo + Ibo $
FRML IfklIo            fklio            = fklio(-1)*(fImo/fImo(-1)*Imo(-1)/Io(-1) + fIbo/fIbo(-1)*Ibo(-1)/Io(-1)) $
FRML Ifklif            fklif            = fKlif(-1)*(fklim/fklim(-1)*Im(-1)/If(-1) + fklib/fklib(-1)*Ib(-1)/If(-1)
                                          + fKlit/fKlit(-1)*It(-1)/If(-1))*kfklif $
FRML IfkI              fKli             = fKli(-1)*(fKlif/fKlif(-1)*If(-1)/I(-1)
                                          + fKlil/fKlil(-1)*Il(-1)/I(-1))*kfkli $
FRML IfklX             fKlx             = fKlx(-1)*( fXa/fXa(-1)*Xa(-1)/X(-1)    + fXe/fXe(-1)*Xe(-1)/X(-1)
                                          + fXng/fXng(-1)*Xng(-1)/X(-1) + fXne/fXne(-1)*Xne(-1)/X(-1)
                                          + fXnf/fXnf(-1)*Xnf(-1)/X(-1) + fXnn/fXnn(-1)*Xnn(-1)/X(-1)
                                          + fXnb/fXnb(-1)*Xnb(-1)/X(-1) + fXnm/fXnm(-1)*Xnm(-1)/X(-1)
                                          + fXnt/fXnt(-1)*Xnt(-1)/X(-1) + fXnk/fXnk(-1)*Xnk(-1)/X(-1)
                                          + fXnq/fXnq(-1)*Xnq(-1)/X(-1) + fXb/fXb(-1)*Xb(-1)/X(-1)
                                          + fXqh/fXqh(-1)*Xqh(-1)/X(-1) + fXqs/fXqs(-1)*Xqs(-1)/X(-1)
                                          + fXqt/fXqt(-1)*Xqt(-1)/X(-1) + fXqf/fXqf(-1)*Xqf(-1)/X(-1)
                                          + fXqq/fXqq(-1)*Xqq(-1)/X(-1) + fXh/fXh(-1)*Xh(-1)/X(-1)
                                          + fXo/fXo(-1)*Xo(-1)/X(-1))*kfklx                    $
FRML IfklMv            fklmv            = fKlmv(-1)*(fM0/fM0(-1)*M0(-1)/Mv(-1)    + fM1/fM1(-1)*M1(-1)/Mv(-1)
                                          + fM2/fM2(-1)*M2(-1)/Mv(-1)    + fM3k/fM3k(-1)*M3k(-1)/Mv(-1)
                                          + fM3r/fM3r(-1)*M3r(-1)/Mv(-1) + fM3q/fM3q(-1)*M3q(-1)/Mv(-1)
                                          + fM5/fM5(-1)*M5(-1)/Mv(-1)    + fM6m/fM6m(-1)*M6m(-1)/Mv(-1)
                                          + fM6q/fM6q(-1)*M6q(-1)/Mv(-1) + fM7b/fM7b(-1)*M7b(-1)/Mv(-1)
                                          + fM7y/fM7y(-1)*M7y(-1)/Mv(-1) + fM7q/fM7q(-1)*M7q(-1)/Mv(-1)
                                          + fM8/fM8(-1)*M8(-1)/Mv(-1))*kfklmv                          $
FRML IfklMst           fKlmst           = fKlmst(-1)*(fMs/fMs(-1)*Ms(-1)/(Ms(-1)+Mt(-1))
                                          + fMt/fMt(-1)*Mt(-1)/(Ms(-1)+Mt(-1)))*kfklmst $
FRML IfklM             fKlm             = fKlm(-1)*(fKlmv/fKlmv(-1)*Mv(-1)/M(-1)
                                          + fKlmst/fKlmst(-1)*Mst(-1)/M(-1))*kfklm $
FRML IfklEv            fKlev            = fKlev(-1)*(fE0/fE0(-1)*E0(-1)/Ev(-1)    + fE1/fE1(-1)*E1(-1)/Ev(-1)
                                          + fE2/fE2(-1)*E2(-1)/Ev(-1)    + fE3/fE3(-1)*E3(-1)/Ev(-1)
                                          + fE5/fE5(-1)*E5(-1)/Ev(-1)    + fE6/fE6(-1)*E6(-1)/Ev(-1)
                                          + fE7y/fE7y(-1)*E7y(-1)/Ev(-1) + fE7q/fE7q(-1)*E7q(-1)/Ev(-1)
                                          + fE8/fE8(-1)*E8(-1)/Ev(-1))*kfklev                        $
FRML IfklEst           fKlest           = fKlest(-1)*(fEss/fEss(-1)*Ess(-1)/(Es(-1)+Et(-1))
                                          + fEsq/fEsq(-1)*Esq(-1)/(Es(-1)+Et(-1))
                                          + fEt/fEt(-1)*Et(-1)/(Es(-1)+Et(-1)))*kfklest $
FRML IfklE             fKle             = fKle(-1)*(fKlev/fKlev(-1)*Ev(-1)/E(-1)
                                          + fKlest/fKlest(-1)*Est(-1)/E(-1))*kfkle   $
FRML IfklV             fKlv             = fKlv(-1)*( fVma/fVma(-1)*Vma(-1)/V(-1)    + fVme/fVme(-1)*Vme(-1)/V(-1)
                                          + fVmng/fVmng(-1)*Vmng(-1)/V(-1) + fVmne/fVmne(-1)*Vmne(-1)/V(-1)
                                          + fVmnf/fVmnf(-1)*Vmnf(-1)/V(-1) + fVmnn/fVmnn(-1)*Vmnn(-1)/V(-1)
                                          + fVmnb/fVmnb(-1)*Vmnb(-1)/V(-1) + fVmnm/fVmnm(-1)*Vmnm(-1)/V(-1)
                                          + fVmnt/fVmnt(-1)*Vmnt(-1)/V(-1) + fVmnk/fVmnk(-1)*Vmnk(-1)/V(-1)
                                          + fVmnq/fVmnq(-1)*Vmnq(-1)/V(-1) + fVmb/fVmb(-1)*Vmb(-1)/V(-1)
                                          + fVmqh/fVmqh(-1)*Vmqh(-1)/V(-1) + fVmqs/fVmqs(-1)*Vmqs(-1)/V(-1)
                                          + fVmqt/fVmqt(-1)*Vmqt(-1)/V(-1) + fVmqf/fVmqf(-1)*Vmqf(-1)/V(-1)
                                          + fVmqq/fVmqq(-1)*Vmqq(-1)/V(-1) + fVmh/fVmh(-1)*Vmh(-1)/V(-1)
                                          + fVmo/fVmo(-1)*Vmo(-1)/V(-1)
                                          + fVea/fVea(-1)*Vea(-1)/V(-1)    + fVee/fVee(-1)*Vee(-1)/V(-1)
                                          + fVeng/fVeng(-1)*Veng(-1)/V(-1) + fVene/fVene(-1)*Vene(-1)/V(-1)
                                          + fVenf/fVenf(-1)*Venf(-1)/V(-1) + fVenn/fVenn(-1)*Venn(-1)/V(-1)
                                          + fVenb/fVenb(-1)*Venb(-1)/V(-1) + fVenm/fVenm(-1)*Venm(-1)/V(-1)
                                          + fVent/fVent(-1)*Vent(-1)/V(-1) + fVenk/fVenk(-1)*Venk(-1)/V(-1)
                                          + fVenq/fVenq(-1)*Venq(-1)/V(-1) + fVeb/fVeb(-1)*Veb(-1)/V(-1)
                                          + fVeqh/fVeqh(-1)*Veqh(-1)/V(-1) + fVeqs/fVeqs(-1)*Veqs(-1)/V(-1)
                                          + fVeqt/fVeqt(-1)*Veqt(-1)/V(-1) + fVeqf/fVeqf(-1)*Veqf(-1)/V(-1)
                                          + fVeqq/fVeqq(-1)*Veqq(-1)/V(-1) + fVeh/fVeh(-1)*Veh(-1)/V(-1)
                                          + fVeo/fVeo(-1)*Veo(-1)/V(-1))*kfklv                       $
FRML Ifklyf            fKlyf            = fKlyf(-1)*( fYfa/fYfa(-1)*Yfa(-1)/Yf(-1)    + fYfe/fYfe(-1)*Yfe(-1)/Yf(-1)
                                          + fYfng/fYfng(-1)*Yfng(-1)/Yf(-1) + fYfne/fYfne(-1)*Yfne(-1)/Yf(-1)
                                          + fYfnf/fYfnf(-1)*Yfnf(-1)/Yf(-1) + fYfnn/fYfnn(-1)*Yfnn(-1)/Yf(-1)
                                          + fYfnb/fYfnb(-1)*Yfnb(-1)/Yf(-1) + fYfnm/fYfnm(-1)*Yfnm(-1)/Yf(-1)
                                          + fYfnt/fYfnt(-1)*Yfnt(-1)/Yf(-1) + fYfnk/fYfnk(-1)*Yfnk(-1)/Yf(-1)
                                          + fYfnq/fYfnq(-1)*Yfnq(-1)/Yf(-1) + fYfb/fYfb(-1)*Yfb(-1)/Yf(-1)
                                          + fYfqh/fYfqh(-1)*Yfqh(-1)/Yf(-1) + fYfqs/fYfqs(-1)*Yfqs(-1)/Yf(-1)
                                          + fYfqt/fYfqt(-1)*Yfqt(-1)/Yf(-1) + fYfqf/fYfqf(-1)*Yfqf(-1)/Yf(-1)
                                          + fYfqq/fYfqq(-1)*Yfqq(-1)/Yf(-1) + fYfh/fYfh(-1)*Yfh(-1)/Yf(-1)
                                          + fYfo/fYfo(-1)*Yfo(-1)/Yf(-1))*kfklyf                       $
FRML IfklY             fKly             = fKly(-1)*(fKlcp/fKlcp(-1)*Cp(-1)/Y(-1) + fKlco/fKlco(-1)*Co(-1)/Y(-1)
                                          + fKli/fKli(-1)*I(-1)/Y(-1) - fKlm/fKlm(-1)*M(-1)/Y(-1)
                                          + fKle/fKle(-1)*E(-1)/Y(-1))*kfkly $
```