



**AALBORG UNIVERSITY**  
DENMARK

**Aalborg Universitet**

## **Simuleringsprogrammet GRIDNODE**

Evensen, L.

*Publication date:*  
1992

*Document Version*  
Også kaldet Forlagets PDF

[Link to publication from Aalborg University](#)

*Citation for published version (APA):*  
Evensen, L. (1992). *Simuleringsprogrammet GRIDNODE*. Aalborg Universitetsforlag. R / Inst. for Bygningsteknik, Aalborg Universitetscenter Nr. R9205

### **General rights**

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal -

### **Take down policy**

If you believe that this document breaches copyright please contact us at [vbn@aub.aau.dk](mailto:vbn@aub.aau.dk) providing details, and we will remove access to the work immediately and investigate your claim.

---

**INSTITUTTET FOR BYGNINGSTEKNIK**  
DEPT. OF BUILDING TECHNOLOGY AND STRUCTURAL ENGINEERING  
AALBORG UNIVERSITETSCENTER • AUC • AALBORG • DANMARK

---

---

**L. EVENSEN**  
**SIMULERINGSPROGRAMMET GRIDNODE**  
**MARTS 1992**

**ISSN 0902-7513 R9205**

---

## Simuleringsprogrammet GRIDNODE

I henhold til mine tidligere skrifter [5, 6] skal simuleringsmodellen **GRIDNODE** nærmere beskrives i det efterfølgende.

Det skal fremhæves, at håndteringen af de mange knudepunktsværdier sker ved hjælp af såkaldte nøgletal, der muliggør en meget væsentlig reduktion i brugen af 3-dimensionale matricer, der ellers ville belaste arbejdslageret i computeren med store mængder af knudepunktsinformationer. Disse matricer ville endvidere løbende skulle revideres, når temperaturafhængige stofkonstanter ændrer værdi.

Det er derfor søgt at minimere lagerallokeringen ved optimering af algoritmerne. Herunder giver især anvendelsen af nøgletal enorme ressourcebesparelser og muliggør samtidig benyttelsen af databaser i form af katalogoplysninger og "arkivoplysninger" for bygningskomponenter.

L. Evensen

## Strukturen i GRIDNODE-modellen

behandles indgående i det efterfølgende med beskrivelse af simuleringsforløbet i en blokdiagramkonstruktion.

Endvidere dokumenteres modellen med et FORTRAN program:

GRIDNODE.FOR

hvor algoritmerne for modellen er indlagt.

## Nøgletallet NG

Programfunktionen er opbygget omkring et såkaldt NØGLETAL (key value) NG, der tilknyttes knudepunkterne i temperaturfeltet.

Nøgletallet angiver knudepunkternes beliggenhed i forhold til konstruktionselementernes grænseflader (rande) og angiver materialekategorien, et knudepunkt tilhører.

Nøgletallene er grupperet i delmængder af talområdet 1-999, som efterfølgende oversigt viser. Fx angiver et nøgletal i delområdet 100-199, at knudepunktet er beliggende nord (N) for rumelementet.

KNUDEPUNKTER (GRID NODES)	NØGLETAL NG (KEY VALUES NG)
Eksterne knpkt. (External nodes):	
Not active	1-99 (0 is prohibited)
N	100-199
S	200-299
E	300-399
W	400-499
T	500-599
B	600-699
Interne knpkt. (Internal nodes):	
	700-999

Effektstrømme regnes positive i retningerne:

S-N, W-E, B-T

Da hvert enkelt knudepunkt får tilskrevet et nøgletal NG (individuel tilpasset), kan alle "stofværdier" og strålingstilførte effekter QB angives som funktion af NG:

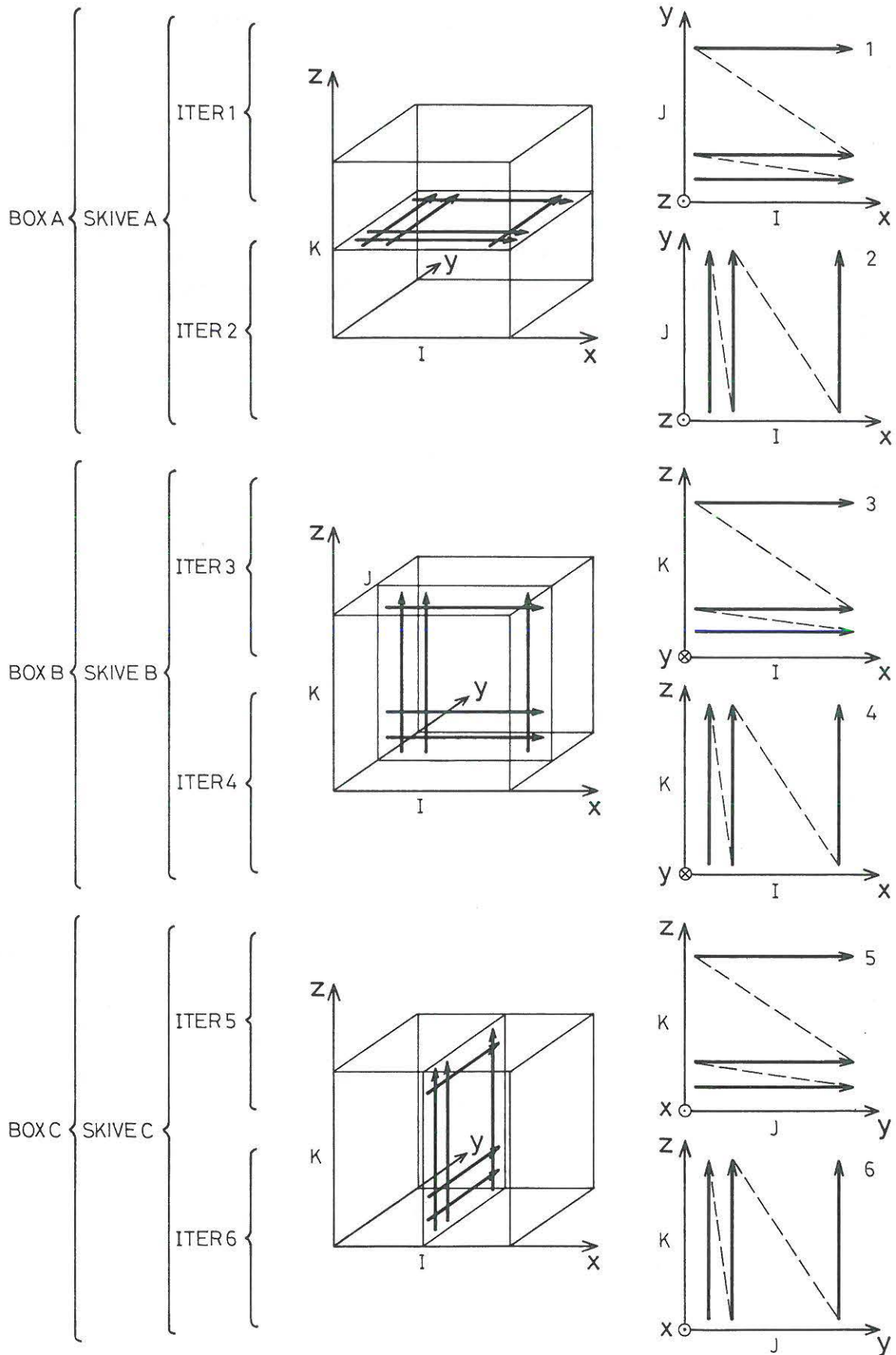
$$\lambda(\text{NG}), c_p(\text{NG}), \rho_p(\text{NG}), \alpha(\text{NG}), \text{QB}(\text{NG})$$

Kun NG(I,J,K), TN(I,J,K), HSO(I,J,K) optræder som 3-dimensionale matricer, mens alle øvrige indicerede variable er en-dimensionale. (TN er afledt af  $t^{new}$ , dvs.  $t^{\tau+\Delta\tau}$ ).

Herved opnås væsentlige fordele ved indlæsning af inddata og håndtering af databaser i form af katalogoplysninger om byggematerialer og konstruktionselementer.

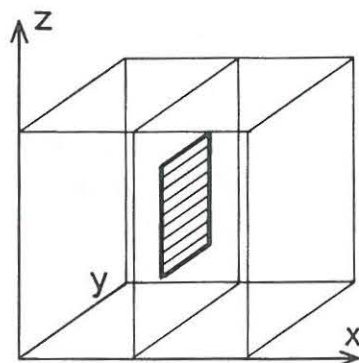
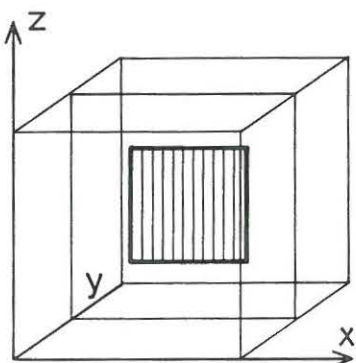
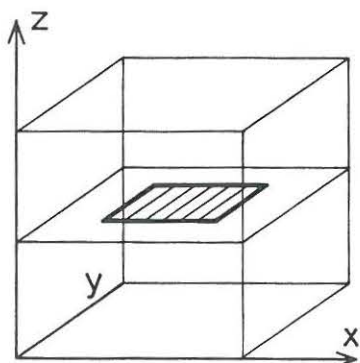
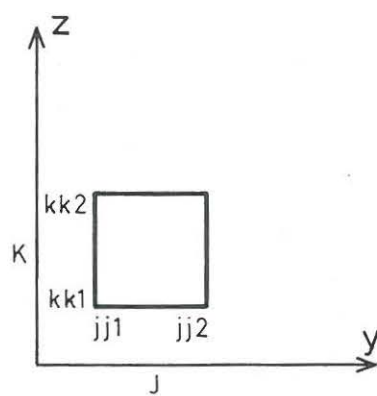
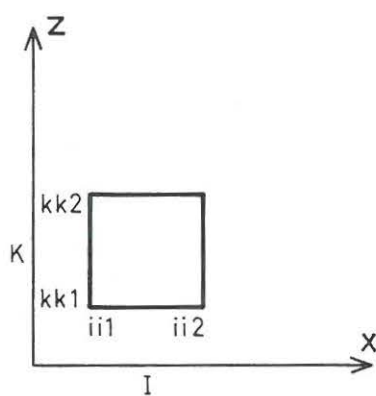
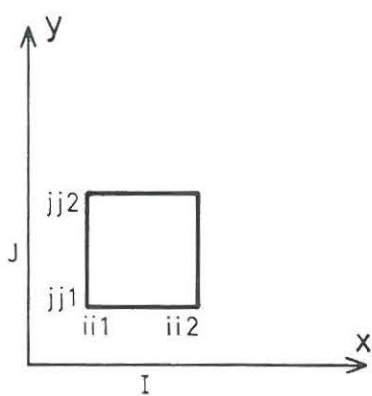
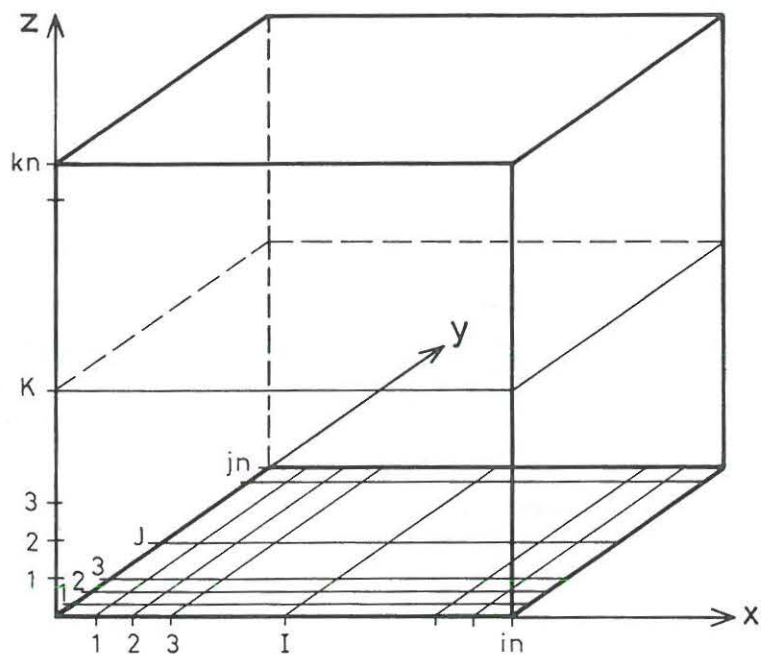
Endvidere kan der hentes detailoplysninger ved legemets rande og indre skilleflader ved struktureret brug af nøgletallene NG. Samtidig opnås en strukturering af output data, der muliggør en differentieret oversigt over væsentlige data for konstruktionens interaktion med omgivelserne og energiakkumulering.

Knudepunktsiterationerne afvikles ved KRYDSITERATION, der er struktureret således:



Efterfølgende figur viser systemidentifikationen og indlæggelsen af "WINDOWS", der afgrænser udvalgte dele af konstruktionen.

### Systemidentifikation



## GRIDNODE

Fortran programmet GRIDNODE er opbygget subroutinebaseret, idet "blokkene" i blokdiagrammet udgør selvstændige subrutiner. Herved undgås den mangel på overblik, der kendetegner hidtil publicerede programstrukturer. Samtidig er de hidtil publicerede programmer opbygget over simple, unuancerede konstruktionselementer, bl.a. med henblik på at kunne anvende automatiske gitterværks-generatorer. Er konstruktionen imidlertid uhyre kompliceret som en bygningsdel med ydervægspartier med vindueselementer, gulv- og loftskonstruktioner, m.v., svigter automatikken.

Forskningsmæssigt må der derfor ved udarbejdelse af en BST-model tages hensyn til konstruktionselementernes "uhåndterlige" form især ved "3-dimensionale" temperaturfelter.

Det er disse komplicerede forløb, BST-programmet håndterer med anvendelse af rimeligt overskuelige indlæsningsrutiner og uddateringer, muliggjort ved anvendelsen af nøgletallene NG (I,J,K). Endvidere muliggør blokdiagramoversigten indlæggelse og tilføjelse af systemrutiner efter ønske samt udskiftning og forbedring af programafsnit uden at miste overblikket.

I denne version af Fortran programmet GRIDNODE er det valgt at overføre konstruktionsgeometrien og materialespecifikationerne fra en ekstern fil (GINITIA.FOR) samtidig med oversættelsen af hovedprogrammet (ved hjælp af en include-sætning).

Programmet GRIDNODE vil i en senere version blive revideret, således at inddateringen organiseres ved hjælp af et menu-organiseret, interaktivt indlæseprogram, der standardiserer indlæsningen af geometri, materialeoplysninger og styringsparametre for den dynamiske temperaturberegning og samtidig styrer udlæsningsmængderne.

Programmet kan i den angivne form forholdsvis let udvides til at beregne/udskrive overfladetemperaturer, effektstrømme og akkumulerede energimængder for ønskede dele af bygningskonstruktionen ved at indlægge beregningsrutiner i subrutinen tidsstep.

Eksemplet der indlæses i gridnode ved hjælp af GINITIA.FOR omhandler en stålterning med nøgletallet 700 og  $\lambda = 52$ ,  $\rho = 7850$ ,  $c_p = 460$ .

Stålterningen har fri sidevægge med  $\alpha = 7,7$  mod omgivelserne, der overalt har temperaturen  $0^\circ\text{C}$  og nøgletallet 1.

Starttemperaturen er  $100^\circ\text{C}$  for stålblokken, hvori der er indlagt 10 knudepunktsplaner, hvoraf de 8 forløber i stålblokkens indre (2-9).

Temperaturfordelingen beregnes for tidsstep  $\Delta\tau = 3600$  sek. = 1 time.

Kommentarlinierne i GRIDNODE programmet (source text) forklarer handlingsforløbet med antydninger af udskriftsmulighederne og placeringen heraf. Læseren kan selv modificere disse udskrifter og mere sig med at indlægge andre mere komplicerede konstruktionselementer.



**Publikationsliste**

1. Louis Evensen: Forskningsteoretisk oplæg inden for klimateknikken, august 1986, ISSN0105-7421 R8614.
2. Louis Evensen: Energiligningen på Computer-form, Diskussion af energiligningen, august 1988, ISSN0902-7513 R8815.
3. Louis Evensen: Det hygrotermiske grundlag for bygningskonstruktioner, november 1988, ISSN0902-7513 R8837.
4. Louis Evensen: Simulering af temperaturforløb i teglstensydervæg, februar 1989, ISSN0902-7513 R8903.
5. Louis Evensen: Effektbalanceligningen for det generelle rumelement september 1990, ISSN0902-7513 R9030.
6. Louis Evensen: Simuleringsmodel til dynamisk temperaturberegning for rumlig bygningskonstruktion september 1990, ISSN0902-7513 R9031.



SIMULERINGSPROGRAMMET  
GRIDNODE

Parameter part  
Declaration part  
Common part  
Initialization part

```

C      Parameter part: gparam.for
      parameter(in=10,jn=10,kn=10)
C      Declaration part: gdecl.for
      integer i,j,k,ii1,jj1,kk1,ii2,jj2,kk2
      integer stnr,it,maxit
      integer f1,fe,f2,f3,f4,f5
      integer f(max(in,jn,kn)),ff(max(in,jn,kn))
      integer ng(in,jn,kn)
      real rt,rp,tid,dt,st,rf,bt
      real dp,rr,vp,ax,ay,az
      real lp,lw,wp,le,pe,ls,sp,lnn,pn,lb,bp,lt,pt
      real sum,ftp,cc,qbwp,qbpe,qbsp,qbpn,qbbp,qbpt
      real x(in),y(jn),z(kn)
      real to(max(in,jn,kn)),am(max(in,jn,kn)),cm(max(in,jn,kn))
      real la(999),al(999),ca(999),rho(999),qb(999)
      real hso(in,jn,kn)
      real*8 tn(in,jn,kn)
C      Common part: gcommon.for
      common i,j,k,ii1,jj1,kk1,ii2,jj2,kk2,
+ ii,jj,kk,
+ stnr,it,maxit,
+ f1,fe,f2,f3,f4,f5,
+ f,ff,ng,
+ rt,rp,tid,dt,st,rf,bt,
+ dp,rr,vp,ax,ay,az,
+ lp,lw,wp,le,pe,ls,sp,lnn,pn,lb,bp,lt,pt,
+ sum,ftp,cc,qbwp,qbpe,qbsp,qbpn,qbbp,qbpt,
+ x,y,z,to,am,cm,la,al,ca,rho,qb,hso,tn
C      Initialization part: ginitia.for
      data x/1.0,1.1,1.2,1.3,1.4,1.5,1.6,1.7,1.8,1.9/
      data y/1.0,1.1,1.2,1.3,1.4,1.5,1.6,1.7,1.8,1.9/
      data z/1.0,1.1,1.2,1.3,1.4,1.5,1.6,1.7,1.8,1.9/
      ii1=2
      ii2=9
      jj1=2
      jj2=9
      kk1=2
      kk2=9 !slut index
      maxit=100
      rt=1.0 !residual limit value
      tid=0.0 !starttid i sek
      dt=3600 !tidsstep i sek
      st=14400 !sluttid i sek
      rf=1.0 !weighting factor
      bt=1.0 !reduction factor for back adjustment
C      data:
      al(1)=7.7
      la(1)=0.0
      ca(1)=0.0

```

```
rho(1)=0.0
qb(1)=0.0
al(700)=0.0
la(700)=52
rho(700)=7850
ca(700)=460
qb(700)=0.0
do kk=1,kn
do jj=1,jn
do ii=1,in
ng(ii,jj,kk)=1 !extern
tn(ii,jj,kk)=0.0 !omg.temp=0 c
enddo
enddo
enddo
do kk=kk1,kk2
do jj=jj1,jj2
do ii=ii1,ii2
ng(ii,jj,kk)=700 !internal node
tn(ii,jj,kk)=100.0 !start temp
enddo
enddo
enddo
```

SIMULERINGSPROGRAMMET  
GRIDNODE

Hovedprogrammet  
temp-program

```

C      program temp_program
C      open(unit=1,file='gg.dat',status='unknown')
C      temp_prog Lines 1-99 MAIN-program
C      *****
C      Declaration part: gdecl.for
C      integer i,j,k,ii1,jj1,kk1,ii2,jj2,kk2
C      integer stnr,it,maxit
C      integer f1,fe,f2,f3,f4,f5
C      integer f(max(in,jn,kn)),ff(max(in,jn,kn))
C      integer ng(in,jn,kn)
C      real rt,rp,tid,dt,st,rf,bt
C      real dp,rr,vp,ax,ay,az
C      real lp,lw,wp,le,pe,ls,sp,lmn,pn,lb,bp,lt,pt
C      real sum,ftp,cc,qbwp,qbpe,qbsp,qbpn,qbbp,qbpt
C      real x(in),y(jn),z(kn)
C      real to(max(in,jn,kn)),am(max(in,jn,kn)),cm(max(in,jn,kn))
C      real la(999),al(999),ca(999),rho(999),qb(999)
C      real hso(in,jn,kn)
C      real*8 tn(in,jn,kn)
C      *****
C      Parameter list: gparam.for
C      parameter(in=10,jn=10,kn=10) F.eks.
C      *****
C      Common list: gcommon.for
C      common i,j,k,ii1,jj1,kk1,ii2,jj2,kk2,
C      + ii,jj,kk,
C      + stnr,it,maxit,
C      + f1,fe,f2,f3,f4,f5,
C      + f,ff,ng,
C      + rt,rp,tid,dt,st,rf,bt,
C      + dp,rr,vp,ax,ay,az,
C      + lp,lw,wp,le,pe,ls,sp,lmn,pn,lb,bp,lt,pt,
C      + sum,ftp,cc,qbwp,qbpe,qbsp,qbpn,qbbp,qbpt,
C      + x,y,z,to,am,cm,la,al,ca,rho,qb,hso,tn
C      *****
C      GRID NODES (Knpkt) KEY VALUES (noegletal)
C      EXTERNAL VALUES:
C      NO ACTION 1-99 0 is prohibited
C      N 100-199
C      S 200-299
C      E 300-399
C      W 400-499
C      T 500-599
C      B 600-699
C      INTERNAL NODES: 700-999
C      *****
C      include'gparam.for'
C      include'gdecl.for'
C      include'gcommon.for'

```

```

include'ginitia.for'
call temp_prog
stop
end
*****
*****

Subroutine temp_prog
C temp_prog Lines 100-999
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C temp_prog_start
!!!!!!!!!!begin
i=ii1
j=jj1
k=kk1
stnr=1 !stepnr
it=1 !Iteration no
tid=tid+dt
f1=0
fe=0 !error signal
!!!!!!!!!!end
200 continue
call tidsstep
if (f1.eq.0)goto200
C call temp_prog_slut
!!!!!!!!!!begin
if (fe.eq.1)write(6,*)' ERROR: iterationerne konvergerer ',
+ 'ikke for ITi=MAXIT'
write(6,*)' —BST—END', 'TID=',tid, 'STEPNO:',stnr, 'IT=',it
!!!!!!!!!!end
return
end
*****

Subroutine tidsstep
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C tidsstep Lines 1000-1999
C call tidsstep_start
!!!!!!!!!!begin
call hso1
C Revision of QB(NG) to TID + DT
call hso2
f2=0
!!!!!!!!!!end
1200 continue
call iteration
if (f2.eq.0)goto1200

```



```

C      call tidsstep_slut
      !!!!!!!!!!!begin
C      Udskriv tn values paa extern fil tillige med TID og STNR
      if (fe.eq.1)then
      f1=1
      goto1300
      endif
C      Beregn/udskriv overfladetemp., effektstroemme og energiakk.
C      paa extern fil eller paa output (ved batch job)
      open(unit=1,file='gg.dat',status='unknown')
      write(1,*)'it no=',it,(tn(iii,5,5),iii=1,in)
      write(1,*)(tn(5,jjj,5),jjj=1,jn)
      write(1,*)(tn(5,5,kkk),kkk=1,kn)
      if ((tid+dt).gt.st)f1=1
      if (f1.eq.0)then
      stnr=stnr+1
      tid=tid+dt
      it=1
C      Korrigjer/revise alfa, lambda, capacity, rho to tid+dt
      write(6,*)'Slut paa tidsstep no:',stnr
      endif
1300  continue
      !!!!!!!!!!!end
      return
      end
      *****

      Subroutine iteration
      include'gparam.for'
      include'gdecl.for'
      include'gcommon.for'
C      iteration Lines 2000-2999
C      call iteration_start
      !!!!!!!!!!!begin
      rp=0.0
      !!!!!!!!!!!end
      call boxa
      call boxb
      call boxc
C      call iteration_slut
      !!!!!!!!!!!begin
      write(6,*)' END of BOX-ITERATION'
      write(6,*)'rp=',rp
      write(6,*)'f2=',f2
      if (rp.le.rt)f2=1
      if (rp.gt.rt.and.(it+1).gt.maxit)then
      f2=1
      fe=1
      endif
      if (f2.eq.0)it=it+1

```

```

if (f2.eq.1)write(6,*)'End of iteration no:',it
!!!!!!!!!!end
return
end
*****

Subroutine boxa
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C boxa Lines 3000-3999
C call boxa_start
!!!!!!!!!!begin
k=kk1
f3=0
!!!!!!!!!!end
3200 continue
call skivea
if (f3.eq.0)goto3200
C call boxa_slut
!!!!!!!!!!begin
C Udskriv evt. monitorline
C write(6,*)(tn(ii,5,5),ii=1,in)
C write(6,*)(tn(5,jj,5),jj=1,jn)
C write(6,*)(tn(5,5,kk),kk=1,kn)
C PAUSE'BOXA TERMINATED,TYPE CONTINUE OR EXIT TO RESUME EXECUTION'
!!!!!!!!!!end
return
end
*****

Subroutine skivea
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C skivea Lines4000-4999
C call skivea_start
!!!!!!!!!!begin
!!!!!!!!!!end
call skivea_it1
call skivea_it2
C call skivea_slut
!!!!!!!!!!begin
k=k+1
if (k.gt.kk2)f3=1
!!!!!!!!!!end
return
end
*****

Subroutine skivea_it1
include'gparam.for'

```

```

include'gdecl.for'
include'gcommon.for'
C skivea_iteration1 Lines 5000-5999
C call skivea_it1_start
!!!!!!!!!!!!begin
j=jj1
do jj=1,jn
ff(jj)=1
enddo
f4=0
!!!!!!!!!!!!end
5200 continue
call linieit1
if (f4.eq.0)goto5200
C call skivea_it1_slut
!!!!!!!!!!!!begin
C Udskriv evt. monitorline
!!!!!!!!!!!!end
return
end
*****

Subroutine linieit1
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C linieiteration1 Lines 6000-6999
C call linieit1.start
!!!!!!!!!!!!begin
i=ii1
do ii=1,in
f(ii)=1 !external
am(ii)=0.0
cm(ii)=tn(ii,j,k)
enddo
f5=0
!!!!!!!!!!!!end
6200 continue
call cell.c1
if (f5.eq.0)goto6200
C call linieit1_slut
!!!!!!!!!!!!begin
C Test if the column is internal 0 or external 1
do ii=ii1,ii2
if (f(ii).eq.0)ff(j)=0
enddo
C If ff(j)=0 (internal column) then perform back substitution
if (ff(j).eq.0)then
do ii=ii1,ii2
to(ii)=tn(ii,j,k) !keep tn-values in to for bt-reduction

```



```

rp=rp+abs(rr)
endif
return
end
*****

Subroutine skivea_it2
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C skivea_iteration2 Lines 9000-9999
C call skivea_it2_start
!!!!!!!!!!begin
i=ii1
do ii=1,in
ff(ii)=1
enddo
f4=0
!!!!!!!!!!end
9200 continue
call linieit2
if (f4.eq.0)goto9200
C call skivea_it2_slut
!!!!!!!!!!begin
C Udskriv evt. monitorline
!!!!!!!!!!end
return
end
*****

Subroutine linieit2
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C linieiteration2 Lines 10000-10999
C call linieit2_start
!!!!!!!!!!begin
j=jj1
do jj=1,jn
f(jj)=1
am(jj)=0.0
cm(jj)=tn(i,jj,k)
enddo
f5=0
!!!!!!!!!!end
10200 continue
call cell_c2
if (f5.eq.0)goto10200
C call linieit2_slut
!!!!!!!!!!begin
C back substitution

```

```

do jj=jj1,jj2
if (f(jj).eq.0)ff(i)=0
enddo
C If ff(i)=0 (internal column) then perform back substitution
if (ff(j).eq.0)then
do jj=jj1,jj2
to(jj)=tn(i,jj,k) !keep tn-values in to for bt-reduction
enddo
do jj=jj2,jj1,-1
tn(i,jj,k)=am(jj)*tn(ii,j+1,k)+cm(jj)
enddo
C Perform bt-reduction
if (bt.lt.1.0)then
do jj=jj1,jj2
tn(i,jj,k)=bt*tn(i,jj,k)+(1.0-bt)*to(jj)
enddo
endif
C End of back substitution
endif
i=i+1
if (i.gt.ii2)f4=1
!!!!!!!!!!end
return
end
*****

Subroutine cell_c2
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C cell_calc2 Lines 11000-11999
C call cell_c2_start
!!!!!!!!!!begin
if (ng(i,j,k).ge.700)f(j)=0 !INTERNAL GRID NODE
!!!!!!!!!!end
call trans2 !Lines 12000-12999
C call cell_c2_slut
!!!!!!!!!!begin
j=j+1
if (j.gt.jj2)f5=1
!!!!!!!!!!end
return
end
*****

Subroutine trans2
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C trans2 Lines 12000-12999
if (f(j).eq.0)then

```

```

C      Calculation of am(j), cm(j) and rp (residualsum)
      call transsub
      am(j)=rf*pn/(ftp-rf*sp*am(j-1))
      cc=hso(i,j,k)+rf*wp*tn(i-1,j,k)+rf*pe*tn(i+1,j,k)+
+ rf*bp*tn(i,j,k-1)+rf*pt*tn(i,j,k+1)
      cm(j)=(cc+rf*sp*cm(j-1))/(ftp-rf*sp*am(j-1))
      rr=ftp*tn(i,j,k)-rf*sp*tn(i,j-1,k)-rf*pn*tn(i,j+1,k)-cc
      rp=rp+abs(rr)
      endif
      return
      end
      *****
      *****
      *****
      Subroutine boxb
      include'gparam.for'
      include'gdecl.for'
      include'gcommon.for'
C      boxb Lines 13000-13999
C      call boxb_start
      !!!!!!!!!!!begin
      j=jj1
      f3=0
      !!!!!!!!!!!end
13200  continue
      call skiveb
      if (f3.eq.0)goto13200
C      call boxb_slut
      !!!!!!!!!!!begin
C      write(6,*)(tn(ii,5,5),ii=1,in)
C      write(6,*)(tn(5,jj,5),jj=1,jn)
C      write(6,*)(tn(5,5,kk),kk=1,kn)
C      PAUSE'BOXB TERMINATED,TYPE CONTINUE OR EXIT TO RESUME EXECUTION'
      !!!!!!!!!!!end
      return
      end
      *****
      Subroutine skiveb
      include'gparam.for'
      include'gdecl.for'
      include'gcommon.for'
C      skiveb Lines 14000-14999
C      call skiveb_start
      !!!!!!!!!!!begin
      !!!!!!!!!!!end
      call skiveb_it3
      call skiveb_it4
C      call skiveb_slut
      !!!!!!!!!!!begin

```

```

j=j+1
if (j.gt,jj2)f3=1
!!!!!!!!!!end
return
end
*****

Subroutine skiveb_it3
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C skiveb_iteration3 Lines 15000-15999
C call skiveb_it3_start
!!!!!!!!!!begin
k=kk1
do kk=1,kn
ff(kk)=1
enddo
f4=0
!!!!!!!!!!end
15200 continue
call linieit3
if (f4.eq.0)goto15200
C call skiveb_it3_slut
!!!!!!!!!!begin
C Udskriv evt. monitorline
!!!!!!!!!!end
return
end
*****

Subroutine linieit3
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C linieiteration3 Lines 16000-16999
C call linieit3_start
!!!!!!!!!!begin
i=ii1
do ii=1,in
f(ii)=1
am(ii)=0.0
cm(ii)=tn(ii,j,k)
enddo
f5=0
!!!!!!!!!!end
16200 continue
call cell_c3
if (f5.eq.0)goto16200
C call linieit3_slut
!!!!!!!!!!begin

```



```

C      back substitution
      do ii=ii1,ii2
      if (f(ii).eq.0)ff(k)=0
      enddo
C      If ff(k)=0 (internal column) then perform back substitution
      if (ff(k).eq.0)then
      do ii=ii1,ii2
      to(ii)=tn(ii,j,k) !keep tn-values in to for bt-reduction
      enddo
      do ii=ii2,ii1,-1
      tn(ii,j,k)=am(ii)*tn(ii+1,j,k)+cm(ii)
      enddo
C      Perform bt-reduction
      if (bt.lt.1.0)then
      do ii=ii1,ii2
      tn(ii,j,k)=bt*tn(ii,j,k)+(1.0-bt)*to(ii)
      enddo
      endif
C      End of back substitution
      endif
      k=k+1
      if (k.gt.kk2)f4=1
      !!!!!!!!!end
      return
      end
      *****
      Subroutine cell_c3
      include'gparam.for'
      include'gdecl.for'
      include'gcommon.for'
C      cell_calc3 Lines 17000-17999
C      call cell_c3_start
      !!!!!!!!!begin
      if (ng(i,j,k).ge.700)f(i)=0 !INTERNAL GRID NODE
      !!!!!!!!!end
      call trans3 !Lines 18000-18999
C      call cell_c3_slut
      !!!!!!!!!begin
      i=i+1
      if (i.gt.ii2)f5=1
      !!!!!!!!!end
      return
      end
      *****
      Subroutine trans3
      include'gparam.for'
      include'gdecl.for'
      include'gcommon.for'
C      trans3 Lines 18000-18999

```

```

if (f(i).eq.0)then
C   Calculation of am(i), cm(i) and rp (residualsum)
   call transsub
   am(i)=rf*pe/(ftp-rf*wp*am(i-1))
   cc=hso(i,j,k)+rf*sp*tn(i,j-1,k)+rf*pn*tn(i,j+1,k)+
+ rf*bp*tn(i,j,k-1)+rf*pt*tn(i,j,k+1)
   cm(i)=(cc+rf*wp*cm(i-1))/(ftp-rf*wp*am(i-1))
   rr=ftp*tn(i,j,k)-rf*wp*tn(i-1,j,k)-rf*pe*tn(i+1,j,k)-cc
   rp=rp+abs(rr)
   endif
   return
   end
   *****
   *****

Subroutine skiveb_it4
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C   skiveb_iteration4 Lines 19000-19999
C   call skiveb_it4_start
   !!!!!!!!!begin
   i=ii1
   do ii=1,in
   ff(ii)=1
   enddo
   f4=0
   !!!!!!!!!end
19200 continue
   call linieit4
   if (f4.eq.0)goto19200
C   call skiveb_it4_slut
   !!!!!!!!!begin
C   Udskriv monitorline
   !!!!!!!!!end
   return
   end
   *****

Subroutine linieit4
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C   linieiteration4 Lines 20000-20999
C   call linieit4_start
   !!!!!!!!!begin
   k=kk1
   do kk=1,kn
   f(kk)=1
   am(kk)=0.0
   cm(kk)=tn(i,j,kk)

```

```

        enddo
        f5=0
        !!!!!!!!!!!end
20200 continue
        call cell_c4
        if (f5.eq.0)goto20200
C      call linieit4_slut
        !!!!!!!!!!!begin
C      back substitution
        do kk=kk1,kk2
        if (f(kk).eq.0)ff(i)=0
        enddo
C      If ff(i)=0 (internal column) then perform back substitution
        if (ff(i).eq.0)then
        do kk=kk1,kk2
        to(kk)=tn(i,j,kk) !keep tn-values in to for bt-reduction
        enddo
        do kk=kk2,kk1,-1
        tn(i,j,kk)=am(kk)*tn(i,j,kk+1)+cm(kk)
        enddo
C      Perform bt-reduction
        if (bt.lt.1.0)then
        do kk=kk1,kk2
        tn(i,j,kk)=bt*tn(i,j,kk)+(1.0-bt)*to(kk)
        enddo
        endif
C      End of back substitution
        endif
        i=i+1
        if (i.gt.ii2)f4=1
        !!!!!!!!!!!end
        return
        end
        *****

Subroutine cell_c4
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C      cell_calc4 Lines 21000-21999
C      call cell_c4_start
        !!!!!!!!!!!begin
        if (ng(i,j,k).ge.700)f(k)=0
        !!!!!!!!!!!end
        call trans4 !Lines 22000-22999
C      call cell_c4_slut
        !!!!!!!!!!!begin
        k=k+1
        if (k.gt.kk2)f5=1
        !!!!!!!!!!!end

```

```

return
end
*****

Subroutine trans4
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C   trans4 Lines 22000-22999
if (f(i).eq.0)then
C   Calculation of am(i), cm(i) and rp (residualsum)
call transsub
am(k)=rf*pt/(ftp-rf*bp*am(k-1))
cc=hso(i,j,k)+rf*sp*tn(i,j-1,k)+rf*pn*tn(i,j+1,k)+
+ rf*wp*tn(i-1,j,k)+rf*pe*tn(i+1,j,k)
cm(k)=(cc+rf*bp*cm(k-1))/(ftp-rf*bp*am(k-1))
rr=ftp*tn(i,j,k)-rf*bp*tn(i,j,k-1)-rf*pt*tn(i,j,k+1)-cc
rp=rp+abs(rr)
endif
return
end
*****
*****
*****

Subroutine boxc
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C   boxc Lines 23000-23999
C   call boxc_start
!!!!!!!begin
i=ii1
f3=0
!!!!!!!end
23200 continue
call skivec
if (f3.eq.0)goto23200
C   call boxc_slut
!!!!!!!begin
C   open(unit=1,file='gg.dat',status='unknown')
C   write(1,*),'it no=',it,(tn(iii,5,5),iii=1,in)
C   write(1,*),(tn(5,jjj,5),jjj=1,jn)
C   write(1,*),(tn(5,5,kkk),kkk=1,kn)
C   PAUSE'BOXC TERMINATED,TYPE CONTINUE OR EXIT TO RESUME EXECUTION'
!!!!!!!end
return
end
*****

Subroutine skivec
include'gparam.for'

```

```

include'gdecl.for'
include'gcommon.for'
C skivec Lines24000-24999
C call skivec_start
!!!!!!!!!!begin
!!!!!!!!!!end
call skivec_it5
call skivec_it6
C call skivec_slut
!!!!!!!!!!begin
i=i+1
if (i.gt.ii2)f3=1
!!!!!!!!!!end
return
end
*****

Subroutine skivec_it5
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C skivec_iteration5 Lines 25000-25999
C call skivec_it5_start
!!!!!!!!!!begin
k=kk1
do kk=1,kn
ff(kk)=1
enddo
f4=0
!!!!!!!!!!end
25200 continue
call linieit5
if (f4.eq.0)goto25200
C call skivec_it5_slut
!!!!!!!!!!begin
C Udskriv monitorline
!!!!!!!!!!end
return
end
*****

Subroutine linieit5
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C linieiteration5 Lines 26000-26999
C call linieit5_start
!!!!!!!!!!begin
j=jj1
do jj=1,jn
f(jj)=1

```

```

am(jj)=0.0
cm(jj)=tn(i,jj,k)
enddo
f5=0
!!!!!!!!!!end
26200 continue
call cell_c5
if (f5.eq.0)goto26200
C call linieit5_slut
!!!!!!!!!!begin
C back substitution
do jj=jj1,jj2
if (f(jj).eq.0)ff(k)=0
enddo
C If ff(k)=0 (internal column) then perform back substitution
if (ff(k).eq.0)then
do jj=jj1,jj2
to(jj)=tn(i,jj,k) !keep tn-values in to for bt-reduction
enddo
do jj=jj2,jj1,-1
tn(i,jj,k)=am(jj)*tn(i,jj+1,k)+cm(jj)
enddo
C Perform bt-reduction
if (bt.lt.1.0)then
do jj=jj1,jj2
tn(i,jj,k)=bt*tn(i,jj,k)+(1.0-bt)*to(jj)
enddo
endif
C End of back substitution
endif
k=k+1
if (k.gt.kk2)f4=1
!!!!!!!!!!end
return
end
*****
Subroutine cell_c5
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C cell_calc5 Lines 27000-27999
C call cell_c5_start
!!!!!!!!!!begin
if (ng(i,j,k).ge.700)f(j)=0
!!!!!!!!!!end
call trans5 !Lines 28000-28999
C call cell_c5_slut
!!!!!!!!!!begin
j=j+1

```

```

if (j.gt.jj2)f5=1
!!!!!!!end
return
end
*****

Subroutine trans5
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C trans5 Lines 28000-28999
if (f(k).eq.0)then
C Calculation of am(j), cm(j) and rp (residualsum)
call transsub
am(j)=rf*pn/(ftp-rf*sp*am(j-1))
cc=hso(i,j,k)+rf*wp*tn(i-1,j,k)+rf*pe*tn(i+1,j,k)+
+ rf*bp*tn(i,j,k-1)+rf*pt*tn(i,j,k+1)
cm(j)=(cc+rf*sp*cm(j-1))/(ftp-rf*sp*am(j-1))
rr=ftp*tn(i,j,k)-rf*sp*tn(i,j-1,k)-rf*pn*tn(i,j+1,k)-cc
rp=rp+abs(rr)
endif
return
end
*****
*****

Subroutine skivec_it6
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C skivec_iteration6 Lines 29000-29999
C call skivec_it6_start
!!!!!!!begin
j=jj1
do jj=1,jn
ff(jj)=1
enddo
f4=0
!!!!!!!end
29200 continue
call linieit6
if (f4.eq.0)goto29200
C call skivec_it6_slut
!!!!!!!begin
C Udskriv evt monitorline
!!!!!!!end
return
end
*****

Subroutine linieit6
include'gparam.for'

```

```

include'gdecl.for'
include'gcommon.for'
C   linieiteration6 Lines 30000-30999
C   call linieit6_start
      !!!!!!!!!!!begin
      k=kk1
      do kk=1,kn
      f(kk)=1
      am(kk)=0.0
      cm(kk)=tn(i,j,kk)
      enddo
      f5=0
      !!!!!!!!!!!end
30200 continue
      call cell_c6
      if (f5.eq.0)goto30200
C   call linieit6_slut
      !!!!!!!!!!!begin
C   back substitution
      do kk=kk1,kk2
      if (f(kk).eq.0)ff(j)=0
      enddo
C   If ff(j)=0 (internal column) then perform back substitution
      if (ff(j).eq.0)then
      do kk=kk1,kk2
      to(kk)=tn(i,j,kk) !keep tn-values in to for bt-reduction
      enddo
      do kk=kk2,kk1,-1
      tn(i,j,kk)=am(kk)*tn(i,j,kk+1)+cm(kk)
      enddo
C   Perform bt-reduction
      if (bt.lt.1.0)then
      do kk=kk1,kk2
      tn(i,j,kk)=bt*tn(i,j,kk)+(1.0-bt)*to(kk)
      enddo
      endif
C   End of back substitution
      endif
      j=j+1
      if (j.gt.jj2)f4=1
      !!!!!!!!!!!end
      return
      end
      *****

Subroutine cell_c6
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C   cell_calc6 Lines 31000-31999

```



```

C      call cell_c6_start
      !!!!!!!!!!!begin
      if (ng(i,j,k).ge.700)f(k)=0 !INTERNAL GRID NODE
      !!!!!!!!!!!end
      call trans6 !Lines 32000-32999
C      call cell_c6_slut
      !!!!!!!!!!!begin
      k=k+1
      if (k.gt.kk2)f5=1
      !!!!!!!!!!!end
      return
      end
      *****

      Subroutine trans6
      include'gparam.for'
      include'gdecl.for'
      include'gcommon.for'
C      trans6 Lines 32000-32999
      if (f(j).eq.0)then
C      Calculation of am(k), cm(k) and rp (residualsum)
      call transsub
      am(k)=rf*pt/(ftp-rf*bp*am(k-1))
      cc=hso(i,j,k)+rf*sp*tn(i,j-1,k)+rf*pn*tn(i,j+1,k)+
+ rf*wp*tn(i-1,j,k)+rf*pe*tn(i+1,j,k)
      cm(k)=(cc+rf*bp*cm(k-1))/(ftp-rf*bp*am(k-1))
      rr=ftp*tn(i,j,k)-rf*bp*tn(i,j,k-1)-rf*pt*tn(i,j,k+1)-cc
      rp=rp+abs(rr)
      endif
      return
      end
      *****
      *****
      *****

      Subroutine transsub
      include'gparam.for'
      include'gdecl.for'
      include'gcommon.for'
C      transsub lines 40000-40999
      vp=(x(i+1)-x(i-1))*(y(j+1)-y(j-1))*(z(k+1)-z(k-1))/8.0
      ax=(y(j+1)-y(j-1))*(z(k+1)-z(k-1))/4.0
      ay=(x(i+1)-x(i-1))*(z(k+1)-z(k-1))/4.0
      az=(x(i+1)-x(i-1))*(y(j+1)-y(j-1))/4.0
      lp=la(ng(i,j,k))
C      *****
      lw=la(ng(i-1,j,k))+al(ng(i-1,j,k))*(x(i)-x(i-1))/2.0
      wp=2.0*lw*lp*ax/((lw+lp)*(x(i)-x(i-1)))
C      *****
      le=la(ng(i+1,j,k))+al(ng(i+1,j,k))*(x(i+1)-x(i))/2.0
      pe=2.0*lp*le*ax/((lp+le)*(x(i+1)-x(i)))

```

```

C      *****
      ls=la(ng(i,j-1,k))+al(ng(i,j-1,k))*(y(j)-y(j-1))/2.0
      sp=2.0*ls*lp*ay/((ls+lp)*(y(j)-y(j-1)))
C      *****
      lnn=la(ng(i,j+1,k))+al(ng(i,j+1,k))*(y(j+1)-y(j))/2.0
      pn=2.0*lp*lnn*ay/((lp+lnn)*(y(j+1)-y(j)))
C      *****
      lb=la(ng(i,j,k-1))+al(ng(i,j,k-1))*(z(k)-z(k-1))/2.0
      bp=2.0*lb*lp*az/((lb+lp)*(z(k)-z(k-1)))
C      *****
      lt=la(ng(i,j,k+1))+al(ng(i,j,k+1))*(z(k+1)-z(k))/2.0
      pt=2.0*lp*lt*az/((lp+lt)*(z(k+1)-z(k)))
C      *****
      dp=ca(ng(i,j,k))*rho(ng(i,j,k))*vp/dt
      sum=wp+pe+sp+pn+bp+pt
      ftp=dp+rf*sum
      return
      end
      *****
      *****

Subroutine hso1
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C      hso1 Lines 41000-41999
      do kk=kk1,kk2
      do jj=jj1,jj2
      do ii=ii1,ii2
      if (rf.lt.0.9)then
      call trans
      qbwp=lp*ax/(lp+lw)
      qbpe=lp*ax/(lp+le)
      qbsp=lp*ay/(lp+ls)
      qbpn=lp*ay/(lp+lnn)
      qbbp=lp*az/(lp+lb)
      qbpt=lp*az/(lp+lt)
      hso(ii,jj,kk)=(1-rf)*
      + (wp*tn(ii-1,j,k)+pe*tn(ii+1,j,k)+
      + sp*tn(ii,jj-1,kk)+pn*tn(ii,jj+1,kk)+
      + bp*tn(ii,jj,kk-1)+pt*tn(ii,jj,kk+1))+
      + (dp-(1-rf)*sum)*tn(ii,jj,kk)+
      + qbwp*(1-rf)*qb(ng(ii-1,jj,kk))+
      + qbpe*(1-rf)*qb(ng(ii+1,jj,kk))+
      + qbsp*(1-rf)*qb(ng(ii,jj-1,kk))+
      + qbpn*(1-rf)*qb(ng(ii,jj+1,kk))+
      + qbbp*(1-rf)*qb(ng(ii,jj,kk-1))+
      + qbpt*(1-rf)*qb(ng(ii,jj,kk+1))
      else !rf.ge.0.9
      vp=(x(ii+1)-x(ii-1))*(y(jj+1)-y(jj-1))*(z(kk+1)-z(kk-1))/8.0

```

```

dp=ca(ng(ii,jj,kk))*rho(ng(ii,jj,kk))*vp/dt
hso(ii,jj,kk)=dp*tn(ii,jj,kk)
endif
enddo
enddo
enddo
return
end
*****
*****

Subroutine hso2
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C hso2 Lines 42000-42999
if (rf.gt.0.1)then
do kk=kk1,kk2
do jj=jj1,jj2
do ii=ii1,ii2
call trans
qbwp=lp*ax/(lp+lw)
qbpe=lp*ax/(lp+le)
qbbsp=lp*ay/(lp+ls)
qbpn=lp*ay/(lp+lcn)
qbbp=lp*az/(lp+lb)
qbpt=lp*az/(lp+lt)
hso(ii,jj,kk)=hso(ii,jj,kk)+
+ qbwp*rf*qb(ng(ii-1,jj,kk))+
+ qbpe*rf*qb(ng(ii+1,jj,kk))+
+ qbbsp*rf*qb(ng(ii,jj-1,kk))+
+ qbpn*rf*qb(ng(ii,jj+1,kk))+
+ qbbp*rf*qb(ng(ii,jj,kk-1))+
+ qbpt*rf*qb(ng(ii,jj,kk+1))
enddo
enddo
enddo
endif
return
end
*****
*****
*****
*****

Subroutine trans
include'gparam.for'
include'gdecl.for'
include'gcommon.for'
C trans Lines 43000-43999
vp=(x(ii+1)-x(ii-1))*(y(jj+1)-y(jj-1))*(z(kk+1)-z(kk-1))/8.0

```

```

ax=(y(jj+1)-y(jj-1))*(z(kk+1)-z(kk-1))/4.0
ay=(x(ii+1)-x(ii-1))*(z(kk+1)-z(kk-1))/4.0
az=(x(ii+1)-x(ii-1))*(y(jj+1)-y(jj-1))/4.0
lp=la(ng(ii,jj,kk))
C *****
lw=la(ng(ii-1,jj,kk))+al(ng(ii-1,jj,kk))*(x(ii)-x(ii-1))/2.0
wp=2.0*lw*lp*ax/((lw+lp)*(x(ii)-x(ii-1)))
C *****
le=la(ng(ii+1,jj,kk))+al(ng(ii+1,jj,kk))*(x(ii+1)-x(ii))/2.0
pe=2.0*lp*le*ax/((lp+le)*(x(ii+1)-x(ii)))
C *****
ls=la(ng(ii,jj-1,kk))+al(ng(ii,jj-1,kk))*(y(jj)-y(jj-1))/2.0
sp=2.0*ls*lp*ay/((ls+lp)*(y(jj)-y(jj-1)))
C *****
lnn=la(ng(ii,jj+1,kk))+al(ng(ii,jj+1,kk))*(y(jj+1)-y(jj))/2.0
pn=2.0*lp*lnn*ay/((lp+lnn)*(y(jj+1)-y(jj)))
C *****
lb=la(ng(ii,jj,kk-1))+al(ng(ii,jj,kk-1))*(z(kk)-z(kk-1))/2.0
bp=2.0*lb*lp*az/((lb+lp)*(z(kk)-z(kk-1)))
C *****
lt=la(ng(ii,jj,kk+1))+al(ng(ii,jj,kk+1))*(z(kk+1)-z(kk))/2.0
pt=2.0*lp*lt*az/((lp+lt)*(z(kk+1)-z(kk)))
C *****
dp=ca(ng(ii,jj,kk))*rho(ng(ii,jj,kk))*vp/dt
sum=wp+pe+sp+pn+bp+pt
ftp=dp+rf*sum
return
end

```