TICRA

TICRA
RF Field data
in Spherical grid

TICRA
LÆDERSTRÆDE 34 · DK-1201 COPENHAGEN K DENMARK
TELEPHONE +45 33 12 45 72 TELEFAX +45 33 12 08 80
E-MAIL ticra@ticra.com http://www.ticra.com

VAT REGISTRATION NO. DK-1055 8697
TICRA FOND, CVR REG. NO. 1055 8697

1. FILE NAMES

Nominal field from the horns HFI217_1, HFI217_2, HFI217_5 and HFI217_6 is given in file names SrPp1_t.grd, SrPp2_t.grd, SrPp5_t.grd and SrPp6_t.grd, respectively.

Distorted field from the horns HFI217_1, HFI217_2, HFI217_5 and HFI217_6 is given in file names SrPp1_td.grd, SrPp2_td.grd, SrPp5_td.grd and SrPp6_td.grd, respectively.

Difference field from the horns HFI217_1, HFI217_2, HFI217_5 and HFI217_6 is given in file names SrPp1_dif_t.grd, SrPp2_dif_t.grd, SrPp5_dif_t.grd and SrPp6_dif_t.grd, respectively.

All field values are in major/minor components, see next chapter

2. FIELD DATA IN RECTANGULAR GRID

This format is used for storing field values in a rectangular grid. Files of this type are generated by objects of the classes Spherical Field Grid and the file extension is *.grd*. Data points are located on a sphere.

If the field points are located on a sphere the direction to a field point $\hat{r}$ is connected to the polar angles θ and ϕ by

$$\hat{r} = \hat{x}\sin\theta\cos\phi + \hat{y}\sin\theta\sin\phi + \hat{z}\cos\theta.$$

The field points are in all cases parameterised by two variables which are generally denoted X and Y and which run from XMIN to XMAX and from YMIN to YMAX, respectively. The file is organised so that X is varying faster than Y. The variables X and Y should be considered as general names for the actual variables which may e.g. be ϕ and θ for points on a sphere or any of the other options as specified by IGRID below.

An example of a routine reading the file is given in Figure F2.2-1

The file format is:

```
Line     Contents     Format

1     TEXT (characters)

         TEXT - Line with identification text.

         This line is repeated until a line contain-
         ing ++++ as the first 4 characters is
         reached.

2     KTYPE     (integer)

         Specifies type of file format

         KTYPE = 1 - standard format for 2D grid.
                    For files used
                     in GRASP9 this variable is al-
                    ways 1.
```

3 NSET, ICOMP, NCOMP, IGRID (4 integers)

NSET - Number of field sets or beams.
ICOMP – Control parameter of field components.
NCOMP - Number of components.
IGRID – Control parameter of field grid type.

For spherical grid:

The field components F1, F2 are controlled by the parameter ICOMP. For near fields the third component F3 always contains the radial E_r component.

ICOMP
- =1 linear E_θ and E_ϕ.
- =2 Right and left hand circular components (rhc,lhc).
- =3 Linear co and cx components (Ludwig's third definition).
- =4 Major and minor axes of polarisation ellipse.
- =5 XPD fields: E_θ/E_ϕ and E_ϕ/E_θ.
- =6 XPD fields: rhc/lhc and lhc/rhc.
- =7 XPD fields: co/cx and cx/co.
- =8 XPD fields: major/minor and minor/major.
- =9 total power $|\bar{E}|$ and $\sqrt{rhc/lhc}$.

NCOMP - Number of field components.
- =2 The file contains two field components for each point as specified above.
- =3 If the field is a near field the file also contains the third radial component.

IGRID - Type of field grid
- =1 uv-grid:
  (X,Y) = (u,v) where u and v are the

two first coordinates of the unit vector to the field point. Hence, $\hat{r} = \left(u, v, \sqrt{1-u^2-v^2}\right)$.
u and v are related to the spherical angles by
$u = \sin\theta\cos\phi, \quad v = \sin\theta\sin\phi$.

=4 Elevation over azimuth:
(X,Y)=(Az,El), where Az and El define the direction to the field point by
$\hat{r} = \left(-\sin Az \cos El, \sin El, \cos Az \cos El\right)$
.

=5 Elevation and azimuth:
(X,Y)=(Az,El), where Az and El define the direction to the field point through the relations
$Az = -\theta\cos\phi, \quad El = \theta\sin\phi$
to the spherical angles θ and ϕ.

=6 Azimuth over elevation:
(X,Y)= (Az, El), where Az and El define the direction to the field point by
$\hat{r} = \left(-\sin Az, \cos Az \sin El, \cos Az \cos El\right)$
.

=7 θϕ-grid:
(X,Y)=(ϕ,θ), where θ and ϕ are the spherical angles of the direction to the field point.

4. (IX(I), IY(I), I=1, NSET) (2 integers on each line)

IX,IY - Centre of set or beam No. I. See the following line for explanation.

All the following lines are repeated NSET times

5. XS, YS, XE, YE (4 real numbers)

Limits of 2D grid. The grid points (X,Y) run through the values
X = XCEN + XS +DX*(I-1)
Y = YCEN + YS +DY*(J-1)
where

DX = (XE-XS)/(NX-1), DY = (YE-YS)/(NY-1)
and
XCEN = DX*IX, YCEN = DY*IY .
The number of grid values NX and NY and the range of the index I and J are defined in the following lines.

6 NX, NY, KLIMIT (3 integers)

NX - Number of columns
NY - Number of rows
KLIMIT - Specification of limits in a 2D grid
=0 Each row contains data for all NX columns.
=1 The number of data points for each row is defined in the following lines.

The following lines 7 and 8 are repeated NY times (J=1,NY) for each beam

If KLIMIT = 1 line 7 is read

7 IS, IN (2 integers)

IS - Column number of first data point in row J
IN - Number of data points in row J

If KLIMIT = 0, IS and IN are always assumed to be 1 and NX, respectively.

if IN = 0 skip lines No. 8. and repeat from line No. 7.
If IN > 0 continue at line No. 8.2 or 8.3 with IE = IS+IN-1.

If NCOMP = 2

8.2 (F1(I), F2(I)), I = IS, IE
(4 real numbers on each line)

F1,F2 - Complex field with two components.

If NCOMP = 3

```
8.3  (F1(I), F2(I), F3(I)), I = IS, IE
          (6 real numbers on each line)

        F1,F2,F3 - Complex field with three compo-
                   nents.

             -----end of data file-----
```

For ICOMP=1, 2, 3, 5, 6 or 7 F1, F2 (and optionally F3) contain the real and imaginary parts of the field in linear scale.

For ICOMP=4 F1 and F2 contain

Real part of F1 is major axes of polarisation ellipse (linear scale)

Real part of F2 is minor axes of polarisation ellipse (linear scale)

Imaginary part of F1 and F2 is zero

For ICOMP=8

Real part of F1 is the major axis divided by the minor axis of the polarisation ellipse (linear scale).

Real part of F2 is the minor axis divided by the major axis of the polarisation ellipse (linear scale).

Imaginary parts of F1 and F2 are zero.

For ICOMP=9 F1 and F2 contain

Real part of F1 is total power $|\bar{E}|$ of field (linear scale)

Imaginary part of F1 is zero

F2 is the complex square root of the ratio rhc/lhc. The phase of this value is the rotation angle of the polarisation ellipse.

Example of routine reading a grid file.

```
SUBROUTINE READ_FIELD_GRID
! Example of reading field data from a 2D-grid file

  ! declarations
  REAL :: &
    XS, YS, XE, YE   ,&     ! Grid limits
    A1, A2, A3, A4, A5, A6  ! Temporary variables

  INTEGER :: &
    UNIT_NO=15, & ! Unit no of open input file
    IER       , & ! Error code
    KTYPE     , & ! Type of file format
    NSET      , & ! Number of data sets
    ICOMP     , & ! Control of field components
    NCOMP     , & ! Number of field components
    IGRID     , & ! Control of grid type
    I_SET     , & ! Index of data set
    NX, NY    , & ! Number of grid points
    KLIMIT    , & ! Limits of 2D grid
    IS, IN, IE, & ! Limits and number of points in row
    I, J          ! Loop index

  CHARACTER(LEN=72) :: &
    MESS             , & ! message
    TEXT             , & ! File header
    FILE_FORM            ! Format of file

  COMPLEX :: AJ = (0.,1.)
  COMPLEX, POINTER, SAVE :: &
    F1(:,:), & ! Array for storing first component of field
data
    F2(:,:), & ! Array for storing second component of
field data
    F3(:,:)    ! Array for storing third component of field
data

  INTEGER, POINTER, SAVE :: &
    I_X(:), I_Y(:)   ! Beam centers

  LOGICAL :: INIT=.TRUE. ! Initialisation flag

  ! Initialisation
  IF (INIT) THEN
    NULLIFY( &
           I_X, I_Y, &
           F1, F2, F3)
    INIT = .FALSE.
  ENDIF
```

Figure F2.2-1 Example of program for reading 2D field data.

```
      IF (FILE_FORM == 'formatted') THEN
        ! Formatted input
        DO
          READ(UNIT_NO, '(A)', ERR=9990, END=9990) TEXT
          IF (TEXT(1:4) == '++++') EXIT
        END DO
        READ(UNIT_NO, *, ERR=9990, END=9990) KTYPE
        READ(UNIT_NO, *, ERR=9990, END=9990) NSET, ICOMP,
NCOMP, IGRID
      ELSE
        ! Unformatted data
        DO
          READ(UNIT_NO, ERR=9990, END=9990) TEXT
          IF (TEXT(1:4) == '++++') EXIT
        ENDDO
        READ(UNIT_NO, ERR=9990, END=9990) KTYPE
        READ(UNIT_NO, ERR=9990, END=9990) NSET, ICOMP, NCOMP,
IGRID
      ENDIF

      ! storage allocation
      IF (ASSOCIATED(I_X)) DEALLOCATE(I_X)
      IF (ASSOCIATED(I_Y)) DEALLOCATE(I_Y)
      ALLOCATE(I_X(NSET),I_Y(NSET))

      IF (FILE_FORM == 'formatted') THEN
        ! Formatted input
        DO I = 1, NSET
            READ(UNIT_NO, *, ERR=9990, END=9990) I_X(I), I_Y(I)
        END DO
      ELSE
        ! Unformatted data
        DO I = 1, NSET
            READ(UNIT_NO, ERR=9990, END=9990) I_X(I), I_Y(I)
        END DO
      END IF

      ! Loop through data sets. The following lines are read
NSET times
      sets_loop: &
      DO I_SET = 1, NSET
        IF (FILE_FORM == 'formatted') THEN
          ! Formatted input
          READ(UNIT_NO, *, ERR=9990, END=9990) XS, YS, XE, YE
          READ(UNIT_NO, *, ERR=9990, END=9990) NX, NY, KLIMIT
        ELSE
          ! Unformatted data
            READ(UNIT_NO, ERR=9990, END=9990) XS, YS, XE, YE
            READ(UNIT_NO, ERR=9990, END=9990) NX, NY, KLIMIT
        END IF

        ! storage allocation
        IF (ASSOCIATED(F1)) DEALLOCATE(F1)
        IF (ASSOCIATED(F2)) DEALLOCATE(F2)
        IF (ASSOCIATED(F3)) DEALLOCATE(F3)
        ALLOCATE(F1(NX,NY), F2(NX,NY), F3(NX,NY), STAT=IER)
        IF (IER /= 0) THEN
          MESS = 'Could not allocate enough memory'; GO TO 9998
        END IF

        ! Loop through columns
        column_loop: &
        DO J = 1, NY
          IF (KLIMIT == 1) THEN
            IF (FILE_FORM == 'formatted') THEN
              ! Formatted input
                READ(UNIT_NO, *, ERR=9990, END=9990) IS, IN
            ELSE
              ! Unformatted data
              READ(UNIT_NO, ERR=9990, END=9990) IS, IN
            END IF
            IE = IS + NX - 1
          ELSE
            IS = 1
            IE = NX
          END IF
```

Figure F2.2-1 continued

```
          IF (FILE_FORM == 'formatted') THEN
            ! Formatted input
            IF (NCOMP == 2) THEN
              DO I = IS, IE
                READ(UNIT_NO, *, ERR=9990, END=9990) A1, A2,
A3, A4
                F1(I,J) = A1+AJ*A2; F2(I,J) = A3+AJ*A4
              END DO
            ELSE
              DO I = IS, IE
                READ(UNIT_NO, *, ERR=9990, END=9990) A1, A2,
A3, A4, A5, A6
                F1(I,J)=A1+AJ*A2; F2(I,J)=A3+AJ*A4;
F3(I,J)=A5+AJ*A6
              END DO
            END IF

          ELSE
            ! unformatted
            IF (NCOMP == 2) THEN
              READ(UNIT_NO, ERR=9990, END=9990) &
              (F1(I,J), F2(I,J), I=IS,IE)
            ELSE
              READ(UNIT_NO, ERR=9990, END=9990) &
              (F1(I,J), F2(I,J), F3(I,J), I=IS,IE)
            END IF
          END IF

        ENDDO column_loop

      ENDDO sets_loop

      GO TO 9999

      ! Errors
      9990 CONTINUE; MESS='Error in field data file'
      9998 CONTINUE; IER=1
      CALL OM_ERROR('READ_FIELD_GRID' ,'FATAL' , MESS)

      9999 CONTINUE
      END SUBROUTINE READ_FIELD_GRID
```

Figure F2.2-1 continued