==========

MorphCol Supplement #12 – Program Grid_to_Vox

By Michael Knappertsbusch, 16. February, 2009
Modified, 7. December 2009

History

Comprehensive illustration of variation of bivariate or multivariate morphometric data through geological time, such as morphological evolution of microfossil shells, is often difficult in classical printed media because of their plane format. For example, bivariate data sets occupy a complex structure in morphospace, if they are discretized by gridding of temporal sequences of bivariate X,Y,Z scatter points to bivariate frequency distributions ΔX,ΔY,Z,F, with X and Y being the coordinates of bivariate morphometric scatter points, Z being the geological age, and F being the bivariate frequency of points per grid-cell with a length of ΔX and a width of ΔY (see, for example, Knappertsbusch, 2000). The gridding procedure (i.e. the discretization of continuus measurements in X and Y direction) of the X,Y scatter data is performed with program Grid2.2.out from Knappertsbusch. Investigation of internal structures of such ΔX,ΔY,Z,F data structures requires visualization by series of contoured slices through the data volume, which are not easy to interprete (Knappertsbusch, 2000). The ΔX,ΔY,Z,F data is nothing else than a density distribution of frequency F in the three-dimensional space spanned by morphometric intervals of ΔX, ΔY and time Z, and so represents a volume element (voxel). An example of such data is illustrated in Figure 1 and such representations can ideally be explored with 3-D Software like Voxler from Golden Software, Inc. (http://www.goldensoftware.com).

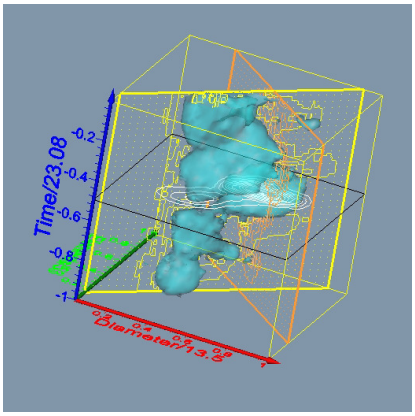

Figure 1.
Example of the bivariate frequency distribution through time of *C. leptoporus* measurements in form of a density distribution. The data were generated with program Grid_to_Vox2 and then were plotted with software Voxler.

The input format for Voxler is just columns of ΔX,ΔY,Z,F, each delimited by tabs. In order to transform gridded data sets, that were generated with program Grid2.2.out, to a format that can be imported to Voxler, a set of Fortran program versions were created, that are described further below. Because the author is still exploratoring for an ideal graphical representation of such complex data, the encoded dimensions, axis ranges and input and output formats are designed primarily for *Calcidiscus leptoporus* and *Globorotalia menardii*, which are our study objects. For application of the programs to other species and measurements, the codes must therefore be slightly modified, but the effort for this is minimal.

The current set of programs consists of four versions for Grid_toVox: Grid_to_Vox1.f, Grid_to_Vox2.f, Grid_to_Vox3.f and Grid_to_Vox4.f (see Table 1). The versions Grid_to_Vox1.f and Grid_to_Vox2.f prepare the data for graphical output with non-normalized frequencies of the frequency distributions, i.e., these non-normalized data represent absolute frequencies of specimens per grid-cell. Grid_to_Vox1.f applies for the evolution of spiral height (=DX) versus axial length (=DY) of shells in keel view (see Knappertsbusch (2007) of *Globorotalia menardii*. Version Grid_to_Vox2.f applies for distributions of coccolith diameter (=DX) versus number of elements in the distal shield (=DY) of *Calcidiscus leptoporus* (see Knappertsbusch (2000). A number of illustrations for *G. menardii* and *C. leptoporus*, that were created using versions Grid_to_Vox1.f and Grid_to_Vox2.f are given in Knappertsbusch (2009).

The versions Grid_to_Vox3.f and Grid_to_Vox4.f prepare the same input data for graphical representation with Voxler but with normalized frequency values for the same species. Normalized frequencies are given as relative frequency per grid-cell, i.e. in % of the total number of specimens in that particular sample. Normalization is necessary for the correct representation of differential frequency changes from one time to the next. Grid_toVox3.f treats measurements for *G. menardii* while version Grid_to_Vox4.f applies for those of *C. leptoporus*. All four program versions follow the same logic and sequence of operations. Input is always a matrix with gridded morphometric measurements that represent a bivariate frequency distribution of the morphometric parameters.

Program version	Applicable to	Normalization
Grid_toVox1.f	*G. menardii*	Without normalzation
Grid_toVox2.f	*C. leptoporus*	Without normalzation
Grid_toVox3.f	*G. menardii*	With normalzation
Grid_toVox4.f	*C. leptoporus*	With normalzation

Table 1. Overview of versions of Grid_to_Vox

Input to Grid_to_Vox:
All Grid_to_Vox versions work in batch operating mode. This means that a large number of gridded input files can be processed one after the other. Two types of input files are required: First, one text file called List_of_files, which contains a list of the age (in Ma) of the sample and the corresponding name of the file with the gridded data matrix per sample. The gridded data matrix contains the frequency distribution of the bivariate set of measurements. The age must be written in digits of five characters, followed by a comma, followed by the name of the gridded input matrix. The name of the gidded input data is 16 characters long. The second type of input files are the files with the gridded data matrices (one file per sample). The gridded data matrices need to be unformatted, i.e. without any header or column information (these must first be removed by manual editing).

Example for Grid_to_Vox1.out (Globorotalia menardii):

File List_of_files:
00.340,input1xxxxx_grid
56.781,input2xxxxx_grid

Gridded data matrix (for *Globorotalia menardii*):
A 14x16 matrix (14 columns, 16 rows).
ΔX goes in horizontal direction (mid-points at 25, 75, 125,..., 675µm).
[Intervals of $\delta\Delta X$=50µm].
ΔY goes in vertical direction (mid-points at 50, 150, 250,...,1550µm).
[$\delta\Delta Y$=100µm].

File input1xxxxx_grid:

1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	16	17	18	19	20	21	22	23	24	25	26	27	28
29	30	31	32	33	34	35	36	37	38	39	40	41	42
43	44	45	46	47	48	49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80	81	82	83	84
85	86	87	88	89	90	91	92	93	94	95	96	97	98
99	100	101	102	103	104	105	106	107	108	109	110	111	112
113	114	115	116	117	118	119	120	121	122	123	124	125	126
127	128	129	130	131	132	133	134	135	136	137	138	139	140
140	141	142	143	144	145	146	147	148	149	150	151	152	153
154	155	156	157	158	159	160	161	162	163	164	165	166	167
167	168	169	170	171	172	173	174	175	176	177	178	179	180
181	182	183	184	185	186	187	188	189	190	191	192	193	194
195	196	197	198	199	200	201	202	203	204	205	206	207	208
209	210	211	212	213	214	215	216	217	218	219	220	221	222

Output

The format of the output data, which can be imported in Voxler is
ΔX, ΔY, Age (Ma), Frequency

Example for file input1xxxxx_grid:

25.	50.	.34	1.
25.	150.	.34	15.
25.	250.	.34	29.
25.	350.	.34	43.
25.	450.	.34	57.
25.	550.	.34	71.
25.	650.	.34	85.
25.	750.	.34	99.
25.	850.	.34	113.
25.	950.	.34	127.
25.	1050.	.34	140.
25.	1150.	.34	154.
25.	1250.	.34	167.
25.	1350.	.34	181.
25.	1450.	.34	195.
25.	1550.	.34	209.
75.	50.	.34	2.
75.	150.	.34	16.
75.	250.	.34	30.
75.	350.	.34	44.
75.	450.	.34	58.
75.	550.	.34	72.
75.	650.	.34	86.
75.	750.	.34	100.
75.	850.	.34	114.
75.	950.	.34	128.
75.	1050.	.34	141.
75.	1150.	.34	155.

75.	1250.	.34	168.
75.	1350.	.34	182.
75.	1450.	.34	196.
75.	1550.	.34	210.
125.	50.	.34	3.
125.	150.	.34	17.
125.	250.	.34	31.
125.	350.	.34	45.
125.	450.	.34	59.
125.	550.	.34	73.
125.	650.	.34	87.
125.	750.	.34	101.
125.	850.	.34	115.
125.	950.	.34	129.
125.	1050.	.34	142.
125.	1150.	.34	156.
125.	1250.	.34	169.
125.	1350.	.34	183.
125.	1450.	.34	197.
125.	1550.	.34	211.
175.	50.	.34	4.
175.	150.	.34	18.
175.	250.	.34	32.
175.	350.	.34	46.
175.	450.	.34	60.
175.	550.	.34	74.
175.	650.	.34	88.
175.	750.	.34	102.
175.	850.	.34	116.
175.	950.	.34	130.
175.	1050.	.34	143.
175.	1150.	.34	157.
175.	1250.	.34	170.
175.	1350.	.34	184.
175.	1450.	.34	198.
175.	1550.	.34	212.
225.	50.	.34	5.
225.	150.	.34	19.
225.	250.	.34	33.
225.	350.	.34	47.
225.	450.	.34	61.
225.	550.	.34	75.
225.	650.	.34	89.
225.	750.	.34	103.
225.	850.	.34	117.
225.	950.	.34	131.
225.	1050.	.34	144.
225.	1150.	.34	158.
225.	1250.	.34	171.
225.	1350.	.34	185.
225.	1450.	.34	199.
225.	1550.	.34	213.
275.	50.	.34	6.
275.	150.	.34	20.
275.	250.	.34	34.
275.	350.	.34	48.
275.	450.	.34	62.

275.	550.	.34	76.
275.	650.	.34	90.
275.	750.	.34	104.
275.	850.	.34	118.
275.	950.	.34	132.
275.	1050.	.34	145.
275.	1150.	.34	159.
275.	1250.	.34	172.
275.	1350.	.34	186.
275.	1450.	.34	200.
275.	1550.	.34	214.
325.	50.	.34	7.
325.	150.	.34	21.
325.	250.	.34	35.
325.	350.	.34	49.
325.	450.	.34	63.
325.	550.	.34	77.
325.	650.	.34	91.
325.	750.	.34	105.
325.	850.	.34	119.
325.	950.	.34	133.
325.	1050.	.34	146.
325.	1150.	.34	160.
325.	1250.	.34	173.
325.	1350.	.34	187.
325.	1450.	.34	201.
325.	1550.	.34	215.
375.	50.	.34	8.
375.	150.	.34	22.
375.	250.	.34	36.
375.	350.	.34	50.
375.	450.	.34	64.
375.	550.	.34	78.
375.	650.	.34	92.
375.	750.	.34	106.
375.	850.	.34	120.
375.	950.	.34	134.
375.	1050.	.34	147.
375.	1150.	.34	161.
375.	1250.	.34	174.
375.	1350.	.34	188.
375.	1450.	.34	202.
375.	1550.	.34	216.
425.	50.	.34	9.
425.	150.	.34	23.
425.	250.	.34	37.
425.	350.	.34	51.
425.	450.	.34	65.
425.	550.	.34	79.
425.	650.	.34	93.
425.	750.	.34	107.
425.	850.	.34	121.
425.	950.	.34	135.
425.	1050.	.34	148.
425.	1150.	.34	162.
425.	1250.	.34	175.
425.	1350.	.34	189.

425.	1450.	.34	203.
425.	1550.	.34	217.
475.	50.	.34	10.
475.	150.	.34	24.
475.	250.	.34	38.
475.	350.	.34	52.
475.	450.	.34	66.
475.	550.	.34	80.
475.	650.	.34	94.
475.	750.	.34	108.
475.	850.	.34	122.
475.	950.	.34	136.
475.	1050.	.34	149.
475.	1150.	.34	163.
475.	1250.	.34	176.
475.	1350.	.34	190.
475.	1450.	.34	204.
475.	1550.	.34	218.
525.	50.	.34	11.
525.	150.	.34	25.
525.	250.	.34	39.
525.	350.	.34	53.
525.	450.	.34	67.
525.	550.	.34	81.
525.	650.	.34	95.
525.	750.	.34	109.
525.	850.	.34	123.
525.	950.	.34	137.
525.	1050.	.34	150.
525.	1150.	.34	164.
525.	1250.	.34	177.
525.	1350.	.34	191.
525.	1450.	.34	205.
525.	1550.	.34	219.
575.	50.	.34	12.
575.	150.	.34	26.
575.	250.	.34	40.
575.	350.	.34	54.
575.	450.	.34	68.
575.	550.	.34	82.
575.	650.	.34	96.
575.	750.	.34	110.
575.	850.	.34	124.
575.	950.	.34	138.
575.	1050.	.34	151.
575.	1150.	.34	165.
575.	1250.	.34	178.
575.	1350.	.34	192.
575.	1450.	.34	206.
575.	1550.	.34	220.
625.	50.	.34	13.
625.	150.	.34	27.
625.	250.	.34	41.
625.	350.	.34	55.
625.	450.	.34	69.
625.	550.	.34	83.
625.	650.	.34	97.

625.	750.	.34	111.
625.	850.	.34	125.
625.	950.	.34	139.
625.	1050.	.34	152.
625.	1150.	.34	166.
625.	1250.	.34	179.
625.	1350.	.34	193.
625.	1450.	.34	207.
625.	1550.	.34	221.
675.	50.	.34	14.
675.	150.	.34	28.
675.	250.	.34	42.
675.	350.	.34	56.
675.	450.	.34	70.
675.	550.	.34	84.
675.	650.	.34	98.
675.	750.	.34	112.
675.	850.	.34	126.
675.	950.	.34	140.
675.	1050.	.34	153.
675.	1150.	.34	167.
675.	1250.	.34	180.
675.	1350.	.34	194.
675.	1450.	.34	208.
675.	1550.	.34	222.

References cited:

Knappertsbusch, M. (2000). Morphologic evolution of the coccolithophorid *Calcidiscus leptoporus* from the Early Miocene to Recent. Journal of Paleontology, vol. 74, No.3, pp. 712-730. Supplementary data.

14. **Knappertsbusch, M.** (2001). A method of illustrating the morphological evolution of coccoliths using 3D animations applied to *Calcidiscus leptoporus*. Paleontologia Electronica, vol. 4, Issue 1. URL: http://palaeo-electronica.org/2001_1/k2/issue1_01.htm

Knappertsbusch, M. (2007). Morphological variability of *Globorotalia menardii* (planktonic foraminiferan) in two DSDP cores from the Caribbean Sea and the Eastern Equatorial Pacific. Carnets de Géologie / Notebooks on Geology, Brest, Article 2007/04 (CG2007_A04). URL: http://paleopolis.rediris.es/cg/CG2007_A04/index.html

Knappertsbusch, M. (2009). Anatomy of morphological evolution: Old data seen in new light. Poster presentation during *The Foraminifera and Nannofossil Groups's Joint Spring Meeting 2009*, "Integrated Studies of evolution, taxonomy, ecology and geochemistry", 4-5 June 2009, ETH Zürich, Switzerland. Download poster as pdf file (29 MB) under http://pages.unibas.ch/museum/microfossils/Research/MORPHCOL/SUPPL_12.pdf

Appendix – Program Codes

Code for Grid_to_Vox1 (*Globorotalia menardii*, without normalization of frequencies):

```
      PROGRAM GRID_TO_VOX
C
C     Version 1, February 5, 2009
C     By Michael Knappertsbusch
C
C     Batch mode of operation
C
C     For Globorotalia menardii:
C     Conversion of gridded data to XYZF data for Voxler:
C     Input is a 14x16 matrix U of gridded X,Y,Z,F data, which was obtained as the output
C     from program Grid.out.
C     The X and Y indicate the size of the grid-cell, i.e. - for G. menardii in
C     keel view - X stands for mid-intervals of DeltaX (=spiral height) and
C     Y stands for mid-intervals of DeltaY (=axial length).
C     Z is the geological age in million years ago, and F is the frequency
C     of specimens per grid-cell. X is arranged in horizontal rows, Y is arranged
C     in vertical direction (columns). Values of F (i.e. the matrix numbers) indicate the
C     frequency in specimens per grid cell.
C
C     Output is a re-organized 4x224 matrix with the same frequency values of F, but with X (=mid-
C     points of DeltaX) being in the first column, Y (=mid-points of DeltaY in the second column,
C     Z (=age) in the third column, and F (=frequency) in the fourthcolumn.
C     This format can be imported to commercial program Voxler (http://www.goldensoftware.com)
C     in order to generate animated 3D graphical representations of the data.
C
C
      INTEGER N,I,J,P
      INTEGER MOA,MOB
      REAL F(1:14,1:16)          ! Input matrix containing Frequency values
      REAL AGE                   ! Age of sample in Ma
      REAL X(1:16),Y(1:14)
      CHARACTER*16 LIST          ! Name of list containing the ages and the file names of the samples
      CHARACTER*16 INPUT   ! Name of file with gridded data
      CHARACTER*16 OUTPUT        ! Name of file with XYZF data ready for Voxler
      CHARACTER*1 TAB
C
      TAB=CHAR(9)                ! Tabulator on a iMac (machine dependent value)
C
C     Midpoints of intervals of DeltaX (=spiral height) or DeltaY (axial length):
C
      DATA (X(I),I=1,16)/50,150,! Intervals for rows for G. menardii
     1  250,350,450,550,650,750,
     2  850,950,1050,1150,1250,
     3  1350,1450,1550/
C
C
      DATA (Y(I),I=1,14)/25,75,125,      ! Intervals for columns for G. menardii
     1  175,225,275,325,375,
     2  425,475,525,575,625,675/
C
C
C
      WRITE(9,*) '*********************************************'
      WRITE(9,*) '*                                           *'
      WRITE(9,*) '*            Grid_to_Vox                    *'
      WRITE(9,*) '*     Preparing gridded data for Voxler     *'
      WRITE(9,*) '*                                           *'
      WRITE(9,*) '* By Michael Knappertsbusch, February 5, 2009 *'
```

```
      WRITE(9,*) '* Version 1, batch processing operation mode  *'
      WRITE(9,*) '*                                             *'
      WRITE(9,*) '*   Adapted for menardiform globorotalids     *'
      WRITE(9,*) '*                                             *'
      WRITE(9,*) '*********************************************'
C
C
C
      WRITE(9,*) '. . .Enter the list with samples. . .'
      WRITE(9,*) '         (max 16 chars)'
      READ(9,3) LIST
 3    FORMAT(A16)
C
C     Reading from LIST the age and the name of input file containing
C     the gridded data:
C
      P=0                                       ! Initialize counter for number of files
      OPEN(14,FILE=LIST,STATUS='OLD')
 5    READ(14,6,END=100) AGE,INPUT
      P=P+1                                     ! Count number of files treated
 6    FORMAT(F6.3,1X,A16)
C
C
C     Determine the name of the output file with XYZF data, then
C     open a new file with that name:
C
      OUTPUT=INPUT(1:11)//'_XYZF'
      OPEN(16,FILE=OUTPUT,STATUS='NEW')
C
C     Now the matrix manipulations for the activ input matrix begin.
C     Reading first the input matrix with Frequency values:
C
      OPEN(15,FILE=INPUT,STATUS='OLD')
      DO 10 J=1,16
        READ(15,*) (F(I,J),I=1,14)
C       WRITE(9,*) (F(I,J),I=1,14)
 10   CONTINUE
C     WRITE(9,*) ' '
C
C
C
C     Calculate indices and write results to output file:
C
      DO 20 N=1,224             ! 224=14 columns * 16 rows
        I=MOA(N)
        J=MOB(N)
        WRITE(16,50) Y(J),TAB,X(I),TAB,AGE,TAB,F(J,I)
 20   CONTINUE
C
 50   FORMAT(F6.0,A1,F6.0,A1,F6.3,A1,F4.0)
C
C
      CLOSE(15)                 ! Closing active input matrix
      GOTO 5                    ! Read age and name of next sample
C
 100  WRITE(9,101) P
 101  FORMAT('. . .',I3,' files treated. . .')
C
      PAUSE 200
 200  CONTINUE
C
      STOP
      END
```

```
      INTEGER FUNCTION MOA(M)
C
C     Function to generate sequences of integer numbers like
C     1,2,3,...16,1,2,3,4,5,...,16,1,2,... as a function of N.
C
      INTEGER M
C
      IF (MOD(M,16).LT.1) THEN
C
C     A note on the above if-statement: The condition needs to be Mod(M,16)=0;
C     however, experimentation showed, that with the modulo expression being set
C     to zero the program did not return the correct result (reason not fully
C     understood): For M=16 or integer multiples of 16 the program always returned
C     a value of 1+Int(M/16) instead of Int(M/16).
C     I think the reason is that the calculated modulo expression does not return
C     absolute zero but something like 0.00000000012357.... This means, that the
C     if statement Mod(M,16)=0 can never be met. Therefore, I use the
C     above alternative by using the if statement being Mod(M,16) < 1.
C
        MOA=16
      ELSE
        MOA=MOD(M,16)
      END IF
C
      RETURN
      END




      INTEGER FUNCTION MOB(M)
C
C     Function to generate sequences of integer numbers like
C     1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
C     2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
C     3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
C     4,4,4,4,... as a function of N.
C
      INTEGER M
C
      IF (MOD(M,16).LT.1) THEN
C
C     A note on the above if-statement: The condition needs to be Mod(M,16)=0;
C     however, experimentation showed, that with the modulo expression being set
C     to zero the program did not return the correct result (reason not fully
C     understood): For M=16 or integer multiples of 16 the program always returned
C     a value of 1+Int(M/16) instead of Int(M/16).
C     I think the reason is that the calculated modulo expression does not return
C     absolute zero but something like 0.00000000012357.... This means, that the
C     if statement Mod(M,16)=0 can never be met. Therefore, I use the
C     above alternative by using the if statement being Mod(M,16) < 1.
C
        MOB=   INT(M/16)
      ELSE
        MOB= 1+INT(M/16)
      END IF
C
      RETURN
      END
```

Code for Grid_to_Vox2 (*Calcidiscus leptoporus*, without normalization of frequencies):

```
      PROGRAM GRID_TO_VOX
C
C     Version 2, February 12, 2009
C     By Michael Knappertsbusch
C
C     Batch mode of operation
C
C     For Calcidiscus leptoporus:
C     Conversion of gridded data to XYZF data for Voxler:
C     Input is a 14x27 matrix U of gridded X,Y,Z,F data, which was obtained as the output
C     from program Grid.out.
C     The X and Y indicate the size of the grid-cell, i.e. - for C. leptoporus:
C     X stands for mid-intervals of DeltaX (=Diameter of the distal shield) and
C     Y stands for mid-intervals of DeltaY (=number of elements in the distal shield).
C     Z is the geological age in million years ago, and F is the frequency
C     of specimens per grid-cell. X is arranged in horizontal rows, Y is arranged
C     in vertical direction (columns). Values of F (i.e. the matrix numbers) indicate the
C     frequency in specimens per grid cell.
C
C     Output is a re-organized 4x378 matrix with the same frequency values of F, but with X (=mid-
C     points of DeltaX) being in the first column, Y (=mid-points of DeltaY in the second column,
C     Z (=age) in the third column, and F (=frequency) in the fourthcolumn.
C     This format can be imported to commercial program Voxler (http://www.goldensoftware.com)
C     in order to generate animated 3D graphical representations of the data.
C
C
      INTEGER N,I,J,P
      INTEGER MOA,MOB
      REAL F(1:14,1:27)          ! Input matrix containing Frequency values
      REAL AGE                   ! Age of sample in Ma
      REAL X(1:27),Y(1:14)
      CHARACTER*16 LIST     ! Name of list containing the ages and the file names of the samples
      CHARACTER*16 INPUT    ! Name of file with gridded data
      CHARACTER*16 OUTPUT        ! Name of file with XYZF data ready for Voxler
      CHARACTER*1 TAB
C
      TAB=CHAR(9)                              ! Tabulator on a iMac (machine dependent value)
C
C     Midpoints of intervals of DeltaX (=Diameter) or DeltaY (Number of elements):
C
      DATA (X(I),I=1,27)/1.0,3.0,         ! Intervals for rows (Elements) for C. leptoporus
     1  5.0,7.0,9.0,11.0,13.0,15.0,
     2  17.0,19.0,21.0,23.0,25.0,
     3  27.0,29.0,31.0,33.0,35.0,
     4  37.0,39.0,41.0,43.0,45.0,
     5  47.0,49.0,51.0,53.0/
C
C
      DATA (Y(I),I=1,14)/0.5,1.5,2.5,              ! Intervals for columns (Diameters) for C. leptoporus
     1  3.5,4.5,5.5,6.5,7.5,
     2  8.5,9.5,10.5,11.5,12.5,13.5/
C
C
C
      WRITE(9,*) '***********************************************'
      WRITE(9,*) '*                                             *'
      WRITE(9,*) '*            Grid_to_Vox                      *'
      WRITE(9,*) '*     Preparing gridded data for Voxler       *'
      WRITE(9,*) '*                                             *'
      WRITE(9,*) '* By Michael Knappertsbusch, February 12, 2009 *'
      WRITE(9,*) '* Version 2, batch processing operation mode   *'
```

```
      WRITE(9,*) '*                                          *'
      WRITE(9,*) '*     Adapted for Calcidiscus leptoporus       *'
      WRITE(9,*) '*                                          *'
      WRITE(9,*) '************************************************'
C
C
C
      WRITE(9,*) '. . .Enter the list with samples. . .'
      WRITE(9,*) '         (max 16 chars)'
      READ(9,3) LIST
 3    FORMAT(A16)
C
C     Reading from LIST the age and the name of input file containing
C     the gridded data:
C
      P=0                                        ! Initialize counter for number of files
      OPEN(14,FILE=LIST,STATUS='OLD')
 5    READ(14,6,END=100) AGE,INPUT
      P=P+1                                      ! Count number of files treated
 6    FORMAT(F6.3,1X,A16)
C
C
C     Determine the name of the output file with XYZF data, then
C     open a new file with that name:
C
      OUTPUT=INPUT(1:11)//'_XYZF'
      OPEN(16,FILE=OUTPUT,STATUS='NEW')
C
C     Now the matrix manipulations for the activ input matrix begin.
C     Reading first the input matrix with Frequency values:
C
      OPEN(15,FILE=INPUT,STATUS='OLD')
      DO 10 J=1,27
        READ(15,*) (F(I,J),I=1,14)
C       WRITE(9,*) (F(I,J),I=1,14)
 10   CONTINUE
C     WRITE(9,*) ' '
C
C
C
C     Calculate indices and write results to output file:
C
      DO 20 N=1,378              ! 378=14 columns * 27 rows
        I=MOA(N)
        J=MOB(N)
        WRITE(16,50) Y(J),TAB,X(I),TAB,AGE,TAB,F(J,I)
 20   CONTINUE
C
 50   FORMAT(F4.1,A1,F4.1,A1,F6.3,A1,F4.0)
C
C
      CLOSE(15)                  ! Closing active input matrix
      GOTO 5                     ! Read age and name of next sample
C
 100  WRITE(9,101) P
 101  FORMAT('. . .',I3,' files treated. . .')
C
      PAUSE 200
 200  CONTINUE
C
      STOP
      END
```

```
      INTEGER FUNCTION MOA(M)
C
C     Function to generate sequences of integer numbers like
C     1,2,3,...27,1,2,3,4,5,...,27,1,2,... as a function of N.
C
      INTEGER M
C
      IF (MOD(M,27).LT.1) THEN
C
C     A note on the above if-statement: The condition needs to be Mod(M,27)=0;
C     however, experimentation showed, that with the modulo expression being set
C     to zero the program did not return the correct result (reason not fully
C     understood): For M=27 or integer multiples of 27 the program always returned
C     a value of 1+Int(M/27) instead of Int(M/27).
C     I think the reason is that the calculated modulo expression does not return
C     absolute zero but something like 0.00000000012357.... This means, that the
C     if statement Mod(M,27)=0 can never be met. Therefore, I use the
C     above alternative by using the if statement being Mod(M,27) < 1.
C
        MOA=27
      ELSE
        MOA=MOD(M,27)
      END IF
C
      RETURN
      END




      INTEGER FUNCTION MOB(M)
C
C     Function to generate sequences of integer numbers like
C     1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
C     2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
C     3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
C     4,4,4,4,... as a function of N.
C
      INTEGER M
C
      IF (MOD(M,27).LT.1) THEN
C
C     A note on the above if-statement: The condition needs to be Mod(M,27)=0;
C     however, experimentation showed, that with the modulo expression being set
C     to zero the program did not return the correct result (reason not fully
C     understood): For M=27 or integer multiples of 27 the program always returned
C     a value of 1+Int(M/27) instead of Int(M/27).
C     I think the reason is that the calculated modulo expression does not return
C     absolute zero but something like 0.00000000012357.... This means, that the
C     if statement Mod(M,27)=0 can never be met. Therefore, I use the
C     above alternative by using the if statement being Mod(M,27) < 1.
C
        MOB=   INT(M/27)
      ELSE
        MOB= 1+INT(M/27)
      END IF
C
      RETURN
      END
```

Code for Grid_to_Vox3 (*Globorotalia menardii*):

```
      PROGRAM GRID_TO_VOX
C
C     Version 3
C     Modified from version 1, April 1, 2009
C     By Michael Knappertsbusch
C
C     Batch mode of operation
C
C     For Globorotalia menardii:
C     Conversion of gridded data to XYZF data for Voxler WITH normalization for frequency:
C     Input is a 14x16 matrix U of gridded X,Y,Z,F data, which was obtained as the output
C     from program Grid.out.
C     The X and Y indicate the size of the grid-cell, i.e. - for G. menardii in
C     keel view - X stands for mid-intervals of DeltaX (=spiral height) and
C     Y stands for mid-intervals of DeltaY (=axial length).
C     Z is the geological age in million years ago, and F is the frequency
C     of specimens per grid-cell. X is arranged in horizontal rows, Y is arranged
C     in vertical direction (columns). Values of F (i.e. the matrix numbers) indicate the
C     frequency in specimens per grid cell.
C
C     Output is a re-organized 4x224 matrix with normalized frequency values of F (in % of the total
C     number of specimens in that sample), but with X (=mid-
C     points of DeltaX) being in the first column, Y (=mid-points of DeltaY in the second column,
C     Z (=age) in the third column, and F (=frequency, in %) in the fourth column.
C     This format can be imported to commercial program Voxler (http://www.goldensoftware.com)
C     in order to generate animated 3D graphical representations of the data.
C
C     For reasons of good scaling of the X,Y and Z axes there are three options for the output:
C     Option 1 (ANS=1) is with scaling of the axes.
C       The X axis is divided by the maximum value of X (=675 or Y(14) ),
C       the Y axis is divided by the maximum value of Y (=1550 or X(16) ),
C       and the Z axis is divided by the maximum age (ZMAX) given in List_of_files.
C       The frequencies are written to file as per cent of the specimen number in that sample.
C       This way, the X,Y,Z values are normalized to values between 0 and 1.
C
C     Option 2 (ANS=2) is without scaling of the X,Y and Z axes, and only the frequencies are
C       normalized (per cent of the specimens per sample).
C
C     Option 3 (ANS=3) is without scaling of the X,Y and Z axes, and without normalization
C       of the frequencies (option 3 corresponds to version Grid_to_Vox1.out)
C
C
C
C     Additional options for the output:
C     The results can be written to external files in two ways:
C
C     Way 1 (BANS=1): For each individual sample input1xxxxx_grid a result file is created (either with
C     names of input1xxxxx_XYZFsn, input1xxxxx_XYZFn, or input1xxxxx_XYZF according to the C
      previously chosen options 1,2 or 3.
```

```
C
C       Way 2 (BANS=2): One single result file is created with the results of all treated input files
C       pasted together. The result file has the name ALL_XYZFsn, ALL_XYZFn or ALL_XYZF according to
C       the previously chosen options 1,2 or 3.
C
C
C
      INTEGER N,I,J,P,ANS,BANS
      INTEGER MOA,MOB
      REAL NSPEC            ! Number of specimens in a particular sample
      REAL F(1:14,1:16)     ! Input matrix containing Frequency values
      REAL AGE,MAXAGE       ! Age of sample, and maximum age of sample in input file list (in Ma)
      REAL X(1:16),Y(1:14)
      REAL ZMAX             ! Maximum age common to all cores (for scaling the Z axis in Voxler, in Ma)
      CHARACTER*16 LIST     ! Name of list containing the ages and the file names of the samples
      CHARACTER*16 INPUT        ! Name of file with gridded data
      CHARACTER*18 OUTPUT       ! Name of file with XYZF data ready for Voxler
      CHARACTER*1 TAB
C
      TAB=CHAR(9)           ! Tabulator on a iMac (machine dependent value)
C
C       Midpoints of intervals of DeltaX (=spiral height) or DeltaY (axial length):
C
      DATA (X(I),I=1,16)/50,150,        ! Intervals for rows for G. menardii
     1  250,350,450,550,650,750,
     2  850,950,1050,1150,1250,
     3  1350,1450,1550/
C
C
      DATA (Y(I),I=1,14)/25,75,125,     ! Intervals for columns for G. menardii
     1  175,225,275,325,375,
     2  425,475,525,575,625,675/
C
C
C
      WRITE(9,*) '************************************************'
      WRITE(9,*) '*                                              *'
      WRITE(9,*) '*              Grid_to_Vox                     *'
      WRITE(9,*) '*      Preparing gridded data for Voxler       *'
      WRITE(9,*) '*                                              *'
      WRITE(9,*) '*  By Michael Knappertsbusch, February 5, 2009  *'
      WRITE(9,*) '* Version 3.1, batch processing operation mode  *'
      WRITE(9,*) '*                                              *'
      WRITE(9,*) '*    Adapted for menardiform globorotalids     *'
      WRITE(9,*) '*       With normalization of frequency        *'
      WRITE(9,*) '*                                              *'
      WRITE(9,*) '************************************************'
C
C
C
C
C
      WRITE(9,*) '. . .Enter the list with samples. . .'
      WRITE(9,*) '         (max 16 chars)'
      READ(9,3) LIST
 3    FORMAT(A16)
C
```

```
C
      WRITE(9,*) '. . .With scaling of X,Y,Z axes ?. . .'
      WRITE(9,*) 'Choose an option (1, 2 or 3)'
      WRITE(9,*) '1=with scaling X,Y,Z AND with normalization F'
      WRITE(9,*) '2=without scaling X,Y,Z but with normalization F'
      WRITE(9,*) '3=without scaling X,Y,Z AND
     * without normalization F'
      READ(9,*) ANS
C
      WRITE(9,*) ' '
      WRITE(9,*) '. . .Options for output. . .'
      WRITE(9,*) 'Choose an option (1 or 2)'
      WRITE(9,*) '1=Separate file for each sample'
      WRITE(9,*) '2=All samples combined in one single file'
      READ(9,*) BANS
C
C     In case of scaling of axes determine the maximum age (=MAXAGE) from the input list:
C
      IF (ANS.EQ.1) THEN
        OPEN(13,FILE=LIST,STATUS='OLD')
        READ(13,6) AGE,INPUT                    ! Read first line
        MAXAGE=AGE                              ! Initialize MAXAGE
 7      READ(13,6,END=111) AGE,INPUT            ! Read next line
        IF (AGE.GT.MAXAGE) THEN
          MAXAGE=AGE
        END IF
        GOTO 7
C
 111    CLOSE(13)
      END IF
C
      WRITE(9,13) MAXAGE
 13   FORMAT('The maximum age is ',F6.3,' Ma')
C
      WRITE(9,*) '. . .Enter the maximum age common
     1 to all cores for scaling the age axis. . .'
      READ(9,*) ZMAX                            ! ZMAX is used to scale the Z-axis in option 1
C
C
C     Reading from LIST the age and the name of input file containing
C     the gridded data:
C
      P=0                                       ! Initialize counter for number of files
      OPEN(14,FILE=LIST,STATUS='OLD')
 5    READ(14,6,END=100) AGE,INPUT
      P=P+1                                     ! Count number of files treated
 6    FORMAT(F6.3,1X,A16)
C
C
C     Determine the name of the output file with XYZF data depending on the options chosen, then
C     open a new file with that name:
C
C     Case BANS=2 (output to one single file):
C
      IF (BANS.EQ.2) THEN
        IF (ANS.EQ.1) THEN
          OUTPUT='ALL_XYZFsn'
```

```
          ELSE IF (ANS.EQ.2) THEN
            OUTPUT='ALL_XYZFn'
          ELSE IF (ANS.EQ.3) THEN
            OUTPUT='ALL_XYZF'
          END IF
          GOTO 9
        END IF
C
C       Case BANS=1 (output to individual files):
C
        IF (ANS.EQ.1) THEN
          OUTPUT=INPUT(1:11)//'_XYZFsn'              ! Option 1 (Scaling axes AND Normalization of F)
        ELSE IF (ANS.EQ.2) THEN
          OUTPUT=INPUT(1:11)//'_XYZFn'               ! Option 2 (only Normalization of F)
        ELSE IF (ANS.EQ.3) THEN
          OUTPUT=INPUT(1:11)//'_XYZF'                ! Option 3 (neither scaling of axes nor normalization
C                                                    ! of F)
        END IF
C
 9      OPEN(16,FILE=OUTPUT,STATUS='NEW')
C
C       Now the matrix manipulations for the activ input matrix begin.
C       Reading first the input matrix with Frequency values:
C
        OPEN(15,FILE=INPUT,STATUS='OLD')
        DO 10 J=1,16
          READ(15,*) (F(I,J),I=1,14)
 10     CONTINUE
C
C
C       Calculate the sum (=NSPEC) of all elements in matrix F
C
        NSPEC=0.0
        DO 12 J=1,16
          DO 11 I=1,14
            NSPEC=NSPEC+F(I,J)                       ! Calculates the number of specimens in the sample
 11       CONTINUE
 12     CONTINUE
C
C
C
C
C       Calculate indices and write results to output file:
C
C*****  Option 1:
        IF (ANS.EQ.1) THEN     ! Output with scaling of X, Y and Z axes and with normalization of F
          DO 20 N=1,224                ! 224=14 columns * 16 rows
            I=MOA(N)
            J=MOB(N)
            WRITE(16,50) Y(J)/Y(14),TAB,X(I)/X(16),TAB,-AGE/ZMAX,
     1 TAB,100*F(J,I)/NSPEC
 20         CONTINUE
        END IF
C
 50     FORMAT(F6.4,A1,F6.4,A1,F7.3,A1,F6.2)
C
C
```

```
C*****  Option 2:
        IF (ANS.EQ.2) THEN     ! Output without scaling of X, Y and Z axes but with normalization of F
          DO 30 N=1,224                ! 224=14 columns * 16 rows
            I=MOA(N)
            J=MOB(N)
            WRITE(16,60) Y(J),TAB,X(I),TAB,-AGE,TAB,100*F(J,I)/NSPEC
 30         CONTINUE
        END IF
C
 60     FORMAT(F6.0,A1,F6.0,A1,F7.3,A1,F6.2)
C
C
C*****  Option 3:
        IF (ANS.EQ.3) THEN     ! Output without scaling of X, Y and Z and without normalization of F
          DO 40 N=1,224                ! 224=14 columns * 16 rows
            I=MOA(N)
            J=MOB(N)
            WRITE(16,70) Y(J),TAB,X(I),TAB,-AGE,TAB,F(J,I)
 40         CONTINUE
        END IF
C
 70     FORMAT(F6.0,A1,F6.0,A1,F7.3,A1,F6.2)
C
C
        CLOSE(15)              ! Closing active input matrix
        GOTO 5                       ! Read age and name of next sample
C
C
C
 100    IF (BANS.EQ.2) THEN
          P=1
        END IF
        WRITE(9,101) P
 101    FORMAT('. . .',I3,' files treated. . .')
C
        PAUSE 200
 200    CONTINUE
C
        STOP
        END




        INTEGER FUNCTION MOA(M)
C
C       Adapted for Voxler plots of G. menardii
C       Function to generate sequences of integer numbers like
C       1,2,3,...16,1,2,3,4,5,...,16,1,2,... as a function of N.
C
        INTEGER M
C
        IF (MOD(M,16).LT.1) THEN
C
C       A note on the above if-statement: The condition needs to be Mod(M,16)=0;
```

```
C       however, experimentation showed, that with the modulo expression being set
C       to zero the program did not return the correct result (reason not fully
C       understood): For M=16 or integer multiples of 16 the program always returned
C       a value of 1+Int(M/16) instead of Int(M/16).
C       I think the reason is that the calculated modulo expression does not return
C       absolute zero but something like 0.00000000012357.... This means, that the
C       if statement Mod(M,16)=0 can never be met. Therefore, I use the
C       above alternative by using the if statement being Mod(M,16) < 1.
C
         MOA=16
        ELSE
         MOA=MOD(M,16)
        END IF
C
        RETURN
        END




        INTEGER FUNCTION MOB(M)
C
C       Adapted for Voxler plots of G. menardii
C       Function to generate sequences of integer numbers like
C       1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
C       2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
C       3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
C       4,4,4,4,... as a function of N.
C
        INTEGER M
C
        IF (MOD(M,16).LT.1) THEN
C
C       A note on the above if-statement: The condition needs to be Mod(M,16)=0;
C       however, experimentation showed, that with the modulo expression being set
C       to zero the program did not return the correct result (reason not fully
C       understood): For M=16 or integer multiples of 16 the program always returned
C       a value of 1+Int(M/16) instead of Int(M/16).
C       I think the reason is that the calculated modulo expression does not return
C       absolute zero but something like 0.00000000012357.... This means, that the
C       if statement Mod(M,16)=0 can never be met. Therefore, I use the
C       above alternative by using the if statement being Mod(M,16) < 1.
C
         MOB=   INT(M/16)
        ELSE
         MOB= 1+INT(M/16)
        END IF
C
        RETURN
        END
```

Code for Grid_to_Vox4 (*Calcidiscus leptoporus*, with normalization of frequencies):

```
      PROGRAM GRID_TO_VOX
C
C     Version 4
C     Modified from version 2, April 1, 2009
C     By Michael Knappertsbusch
C
C     Batch mode of operation
C
C     For Calcidiscus leptoporus:
C     Conversion of gridded data to XYZF data for Voxler:
C     Input is a 14x27 matrix U of gridded X,Y,Z,F data, which was obtained as the output
C     from program Grid.out.
C     The X and Y indicate the size of the grid-cell, i.e. - for C. leptoporus:
C     X stands for mid-intervals of DeltaX (=Diameter of the distal shield) and
C     Y stands for mid-intervals of DeltaY (=number of elements in the distal shield).
C     Z is the geological age in million years ago, and F is the frequency
C     of specimens per grid-cell. X is arranged in horizontal rows, Y is arranged
C     in vertical direction (columns). Values of F (i.e. the matrix numbers) indicate the
C     frequency in specimens per grid cell.
C
C     Output is a re-organized 4x378 matrix with normalized frequency values of F (in % of the
C     total number of specimens in that sample), but with X (=mid-
C     points of DeltaX) being in the first column, Y (=mid-points of DeltaY in the second column,
C     Z (=age) in the third column, and F (=frequency, in %) in the fourth column.
C     This format can be imported to commercial program Voxler (http://www.goldensoftware.com)
C     in order to generate animated 3D graphical representations of the data.
C
C     For reasons of good scaling of the X,Y and Z axes there are three options for the output:
C     Option 1 (ANS=1) is with scaling of the axes.
C       The X axis is divided by the maximum value of X (=13.5 or Y(14) ),
C       the Y axis is divided by the maximum value of Y (=53 or X(27) ),
C       and the Z axis is divided by the maximum age (ZMAX) given in List_of_files.
C       The frequencies are written to file as per cent of the specimen number in that sample.
C       This way, the X,Y,Z values are normalized to values between 0 and 1.
C
C     Option 2 (ANS=2) is without scaling of the X,Y and Z axes, and only the frequencies are
C       normalized (per cent of the specimens per sample).
C
C     Option 3 (ANS=3) is without scaling of the X,Y and Z axes, and without normalization
C       of the frequencies (option 3 corresponds to version Grid_to_Vox2.out)
C
C
C
C     Additional options for the output:
C     The results can be written to external files in two ways:
C
C     Way 1 (BANS=1): For each individual sample input1xxxxx_grid a result file is created (either with
C     names of input1xxxxx_XYZFsn, input1xxxxx_XYZFn, or input1xxxxx_XYZF according to the C
      previously chosen options 1,2 or 3.
C
C     Way 2 (BANS=2): One single result file is created with the results of all treated input files
C     pasted together. The result file has the name ALL_XYZFsn, ALL_XYZFn or ALL_XYZF according to
C     the previously chosen options 1,2 or 3.
C
```

```
C
C
      INTEGER N,I,J,P,ANS,BANS
      INTEGER MOA,MOB
      REAL NSPEC            ! Number of specimens in a particular sample
      REAL F(1:14,1:27)     ! Input matrix containing Frequency values
      REAL AGE,MAXAGE             ! Age of sample, and maximum age of sample in input file list (in Ma)
      REAL X(1:27),Y(1:14)
      REAL ZMAX             ! Maximum age common to all cores (for scaling the Z axis in Voxler, in Ma)
      CHARACTER*16 LIST  ! Name of list containing the ages and the file names of the samples
      CHARACTER*16 INPUT        ! Name of file with gridded data
      CHARACTER*18 OUTPUT       ! Name of file with XYZF data ready for Voxler
      CHARACTER*1 TAB
C
      TAB=CHAR(9)                           ! Tabulator on a iMac (machine dependent value)
C
C     Midpoints of intervals of DeltaX (=Diameter) or DeltaY (Number of elements):
C
      DATA (X(I),I=1,27)/1.0,3.0,           ! Intervals for rows (Elements) for C. leptoporus
     1  5.0,7.0,9.0,11.0,13.0,15.0,
     2  17.0,19.0,21.0,23.0,25.0,
     3  27.0,29.0,31.0,33.0,35.0,
     4  37.0,39.0,41.0,43.0,45.0,
     5  47.0,49.0,51.0,53.0/
C
C
      DATA (Y(I),I=1,14)/0.5,1.5,2.5,       ! Intervals for columns (Diameters) for C. leptoporus
     1  3.5,4.5,5.5,6.5,7.5,
     2  8.5,9.5,10.5,11.5,12.5,13.5/
C
C
C
      WRITE(9,*) '************************************************'
      WRITE(9,*) '*                                              *'
      WRITE(9,*) '*             Grid_to_Vox                      *'
      WRITE(9,*) '*      Preparing gridded data for Voxler       *'
      WRITE(9,*) '*                                              *'
      WRITE(9,*) '*  By Michael Knappertsbusch, February 12, 2009  *'
      WRITE(9,*) '* Version 4.1, batch processing operation mode   *'
      WRITE(9,*) '*                                              *'
      WRITE(9,*) '*      Adapted for Calcidiscus leptoporus      *'
      WRITE(9,*) '*       With normalization of frequency        *'
      WRITE(9,*) '*                                              *'
      WRITE(9,*) '************************************************'
C
C
C
C
C
      WRITE(9,*) '. . .Enter the list with samples. . .'
      WRITE(9,*) '         (max 16 chars)'
      READ(9,3) LIST
 3    FORMAT(A16)
C
C
      WRITE(9,*) '. . .With scaling of X,Y,Z axes ?. . .'
      WRITE(9,*) 'Choose an option (1, 2 or 3)'
```

```
      WRITE(9,*) '1=with scaling X,Y,Z AND with normalization F'
      WRITE(9,*) '2=without scaling X,Y,Z but with normalization F'
      WRITE(9,*) '3=without scaling X,Y,Z AND
     * without normalization F'
      READ(9,*) ANS
C
      WRITE(9,*) ' '
      WRITE(9,*) '. . .Options for output. . .'
      WRITE(9,*) 'Choose an option (1 or 2)'
      WRITE(9,*) '1=Separate file for each sample'
      WRITE(9,*) '2=All samples combined in one single file'
      READ(9,*) BANS
C
C     In case of scaling of axes determine the maximum age (=MAXAGE) from the input list:
C
      IF (ANS.EQ.1) THEN
        OPEN(13,FILE=LIST,STATUS='OLD')
        READ(13,6) AGE,INPUT                    ! Read first line
        MAXAGE=AGE                              ! Initialize MAXAGE
 7      READ(13,6,END=111) AGE,INPUT            ! Read next line
        IF (AGE.GT.MAXAGE) THEN
          MAXAGE=AGE
        END IF
        GOTO 7
C
 111    CLOSE(13)
      END IF
C
      WRITE(9,13) MAXAGE
 13   FORMAT('The maximum age is ',F6.3,' Ma')
C
      WRITE(9,*) '. . .Enter the maximum age common
     1 to all cores for scaling the age axis. . .'
      READ(9,*) ZMAX                            ! ZMAX is used to scale the Z-axis in option 1
C
C
C     Reading from LIST the age and the name of input file containing
C     the gridded data:
C
      P=0                                       ! Initialize counter for number of files
      OPEN(14,FILE=LIST,STATUS='OLD')
 5    READ(14,6,END=100) AGE,INPUT
      P=P+1                                     ! Count number of files treated
 6    FORMAT(F6.3,1X,A16)
C
C
C     Determine the name of the output file with XYZF data depending on the option chosen, then
C     open a new file with that name:
C
C     Case BANS=2 (output to one single file):
C
      IF (BANS.EQ.2) THEN
        IF (ANS.EQ.1) THEN
          OUTPUT='ALL_XYZFsn'
        ELSE IF (ANS.EQ.2) THEN
          OUTPUT='ALL_XYZFn'
        ELSE IF (ANS.EQ.3) THEN
```

```
            OUTPUT='ALL_XYZF'
           END IF
           GOTO 9
          END IF
C
C         Case BANS=1 (output to individual files):
C
          IF (ANS.EQ.1) THEN
           OUTPUT=INPUT(1:11)//'_XYZFsn'              ! Option 1 (Scaling axes AND Normalization of F)
          ELSE IF (ANS.EQ.2) THEN
           OUTPUT=INPUT(1:11)//'_XYZFn'               ! Option 2 (only Normalization of F)
          ELSE IF (ANS.EQ.3) THEN
           OUTPUT=INPUT(1:11)//'_XYZF'                ! Option 3 (neither scaling of axes nor normalization
C                                                     ! of F)
          END IF
C
 9        OPEN(16,FILE=OUTPUT,STATUS='NEW')
C
C         Now the matrix manipulations for the activ input matrix begin.
C         Reading first the input matrix with Frequency values:
C
          OPEN(15,FILE=INPUT,STATUS='OLD')
          DO 10 J=1,27
           READ(15,*) (F(I,J),I=1,14)
C          WRITE(9,*) (F(I,J),I=1,14)
 10       CONTINUE
C
C
C         Calculate the sum (=NSPEC) of all elements in matrix F
C
          NSPEC=0.0
          DO 12 J=1,27
           DO 11 I=1,14
            NSPEC=NSPEC+F(I,J)                        ! Calculates the number of specimens in the sample
 11        CONTINUE
 12       CONTINUE
C
C
C         Calculate indices and write results to output file:
C
C*****    Option 1:
          IF (ANS.EQ.1) THEN     ! Output with scaling of X, Y and Z axes and with normalization of F
           DO 20 N=1,378                  ! 378=14 columns * 27 rows
            I=MOA(N)
            J=MOB(N)
            WRITE(16,50) Y(J)/Y(14),TAB,X(I)/X(27),TAB,-AGE/ZMAX,
     1 TAB,100*F(J,I)/NSPEC
 20         CONTINUE
          END IF
C
 50       FORMAT(F6.4,A1,F6.4,A1,F7.3,A1,F6.2)
C
C
C*****    Option 2:
          IF (ANS.EQ.2) THEN     ! Output without scaling of X, Y and Z axes but with normalization of F
           DO 30 N=1,378                  ! 378=14 columns * 27 rows
            I=MOA(N)
```

```
            J=MOB(N)
            WRITE(16,60) Y(J),TAB,X(I),TAB,-AGE,TAB,100*F(J,I)/NSPEC
 30         CONTINUE
           END IF
C
 60        FORMAT(F6.1,A1,F6.1,A1,F7.3,A1,F6.2)
C
C
C*****     Option 3:
           IF (ANS.EQ.3) THEN     ! Output without scaling of X, Y and Z and without normalization of F
            DO 40 N=1,378                   ! 378=14 columns * 27 rows
             I=MOA(N)
             J=MOB(N)
             WRITE(16,70) Y(J),TAB,X(I),TAB,-AGE,TAB,F(J,I)
 40          CONTINUE
           END IF
C
 70        FORMAT(F6.1,A1,F6.1,A1,F7.3,A1,F6.2)
C
C
           CLOSE(15)               ! Closing active input matrix
           GOTO 5                          ! Read age and name of next sample
C
 100       IF (BANS.EQ.2) THEN
            P=1
           END IF
           WRITE(9,101) P
 101       FORMAT('. . .',I3,' files treated. . .')
C
           PAUSE 200
 200       CONTINUE
C
           STOP
           END




           INTEGER FUNCTION MOA(M)
C
C          Adapted for Voxler plots of C. leptoporus
C          Function to generate sequences of integer numbers like
C          1,2,3,...27,1,2,3,4,5,...,27,1,2,... as a function of N.
C
           INTEGER M
C
           IF (MOD(M,27).LT.1) THEN
C
C          A note on the above if-statement: The condition needs to be Mod(M,27)=0;
C          however, experimentation showed, that with the modulo expression being set
C          to zero the program did not return the correct result (reason not fully
C          understood): For M=27 or integer multiples of 27 the program always returned
C          a value of 1+Int(M/27) instead of Int(M/27).
C          I think the reason is that the calculated modulo expression does not return
C          absolute zero but something like 0.00000000012357.... This means, that the
```

```
C       if statement Mod(M,27)=0 can never be met. Therefore, I use the
C       above alternative by using the if statement being Mod(M,27) < 1.
C
          MOA=27
        ELSE
          MOA=MOD(M,27)
        END IF
C
        RETURN
        END




        INTEGER FUNCTION MOB(M)
C
C       Adapted for Voxler plots of C. leptoporus
C       Function to generate sequences of integer numbers like
C       1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
C       2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
C       3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
C       4,4,4,4,... as a function of N.
C
        INTEGER M
C
        IF (MOD(M,27).LT.1) THEN
C
C       A note on the above if-statement: The condition needs to be Mod(M,27)=0;
C       however, experimentation showed, that with the modulo expression being set
C       to zero the program did not return the correct result (reason not fully
C       understood): For M=27 or integer multiples of 27 the program always returned
C       a value of 1+Int(M/27) instead of Int(M/27).
C       I think the reason is that the calculated modulo expression does not return
C       absolute zero but something like 0.00000000012357.... This means, that the
C       if statement Mod(M,27)=0 can never be met. Therefore, I use the
C       above alternative by using the if statement being Mod(M,27) < 1.
C
          MOB=   INT(M/27)
        ELSE
          MOB= 1+INT(M/27)
        END IF
C
        RETURN
        END
```