

Note No. 22

Northern Forest Research Centre

Edmonton, Alberta

FORTRAN SUBROUTINES FOR BIOMASS COMPUTATION

Forest biomass is a viable partial alternative to the rapidly depleting fossil fuels in Canada. Biomass estimates are needed to assess the energy potential of this renewable resource, which is abundant throughout the country.

Field techniques vary slightly from place to place, although the basic approach taken is similar in most cases. It is possible to compute biomass data readily and more accurately on a computer than to do lengthy and slow calculations manually. Alemdag (1980) has described the methods for collecting field and laboratory data.

This note describes the FORTRAN subroutines developed at NoFRC to compute the dry weight biomass of a tree and its stem and nonstem components from the field and laboratory data using a PDP 11/60 minicomputer. Guidelines are provided for using the subroutines with subsample data. The programs can be easily adapted to run on any computer that has a FORTRAN compiler.

Because of the size and bulk of entire trees, the field and laboratory procedures used at NoFRC (Singh 1982) are based on a subsampling concept in which disk sections are taken from the merchantable and nonmerchantable stem and nonstem tree components to provide relevant biomass information (Fig 1.).

The field and laboratory data collected for each tree component and the disk subsamples include the following basic items:

Stem and nonstem components

- total fresh weight
- height of the tree at each end of the stem and nonstem components.

Disk samples

- diameter outside bark (dob)
- fresh weight
- dry weight of bark
- dry weight of wood
- height of the tree where the stem disk subsample was taken.

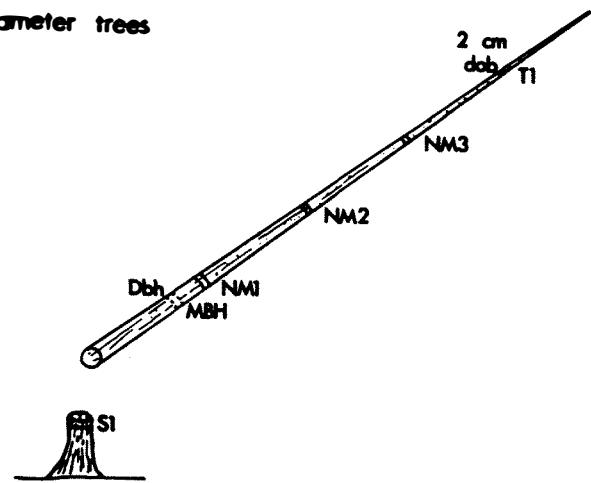
Three main subroutines (RATIOS, STEM, and NSTEM) perform the required computations for determining the dry weights of wood and bark for various components of the tree.

The number of subsamples and the subsampling procedures usually differ among tree components; consequently, the basic computing steps may vary slightly. For stem components, the tree stem samples are taken according to a variable plan: e.g., four subsamples from the merchantable portion (≥ 10 cm dob), three subsamples from the nonmerchantable stem from 10 cm dob up to a minimum specified dob limit (such as ≥ 2 cm dob), and one from near the top (< 2 cm dob). The nonstem components can also have a variable sampling plan: e.g., three or more subsamples for each component. The subroutines are written to accommodate any other sampling procedure involving more or fewer subsamples for each main stem and nonstem component, depending on the available storage space in the computer.

In all cases driver programs¹ (FORTRAN statements) have to be written specifically for the three subroutines to perform the needed computations according to the number of subsamples and tree components used in a study.

¹ Sample statements for running these subroutines at NoFRC are listed in Appendix 1.

Small diameter trees



Large diameter trees

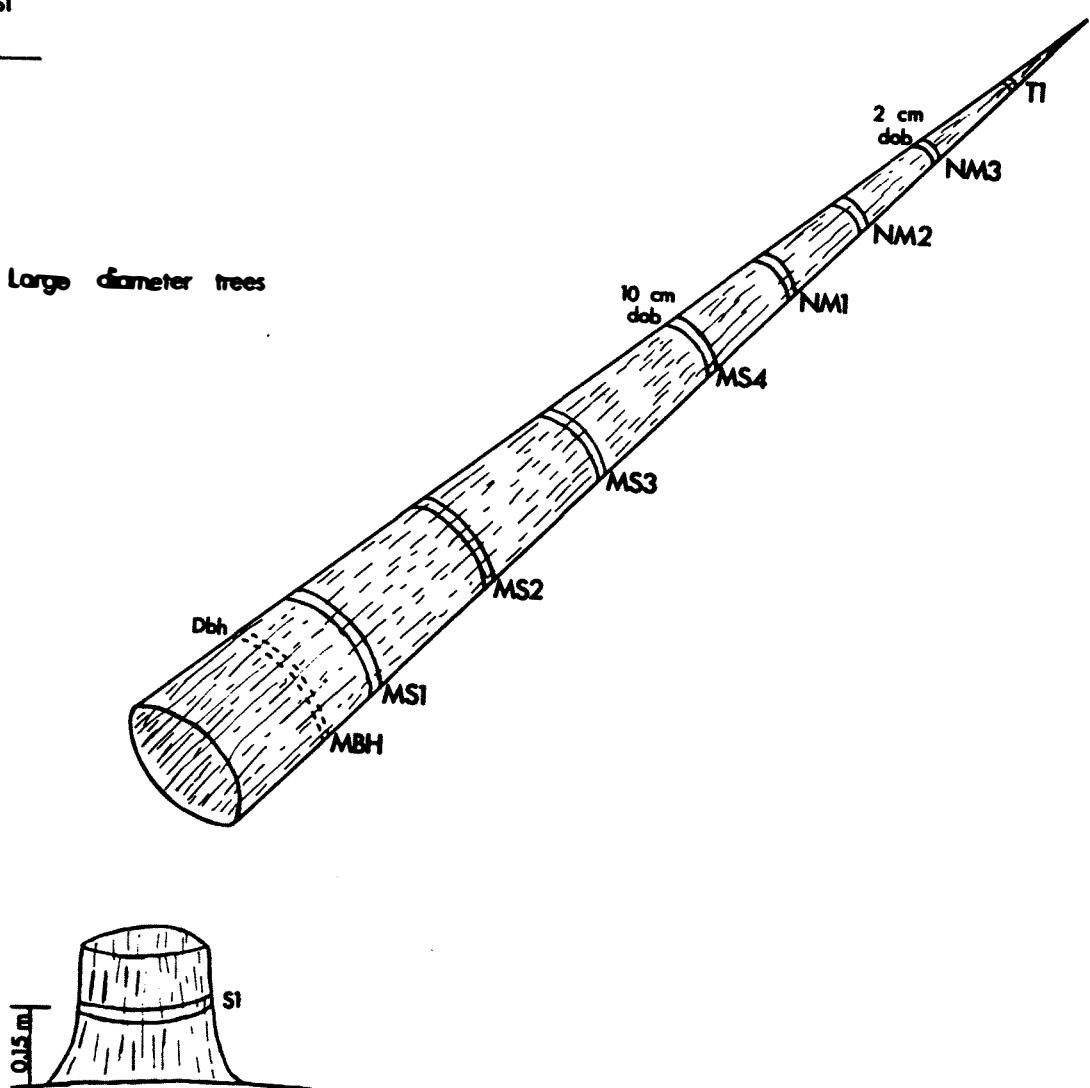


Figure 1. Locations of stem subsamples taken from the merchantable and nonmerchantable trees.

Guidelines for Program Usage

1. Data preparation, screening, and evaluation are important initial steps in any data analysis. Often there are sporadic items that seem to represent impossible conditions (e.g., dry weight \geq fresh weight), items that are inadvertently included with the data, or items that are data entry errors. A data base management system such as DATATRIEVE for PDP or SPIRES for IBM 360/370 can be helpful in locating such oddities or errors, which can then be reexamined, corrected, or deleted from the data file. RATIOS subroutine also has a built-in feature for locating such impossible items, which are subsequently dealt with as missing values.

2. The user must provide the FORTRAN statements to run the computer program subroutines listed in this publication. These statements must include information on the number of subsamples taken for each tree component, the total number of trees, and related information according to the subroutine requirements.

3. The results from the subroutine RATIOS should be checked thoroughly for wild points or outliers, which can be helpful in pinpointing data errors. Any such error must be carefully scrutinized and corrected in the data file before running the rest of the program.

4. The running of program segments containing STEM and NSTEM subroutines will give the biomass information on the dry weight of bark and wood for the stem and nonstem components of the tree. This information can then be summarized or used for deriving prediction equations based on the chosen predictor variables.

Comments are included in each subroutine to help the user in understanding the constraints involved and to assist with any desired modifications for individual applications.

Subroutine RATIOS

The ratios R_i and r_i are computed by this subroutine. All computed ratios are to be stored in the computer for later use in STEM and NSTEM subroutines.

The ratios R_i and r_i are determined as follows for each stem and nonstem subsample:

$$R_i = \frac{\text{Dry weight of wood} + \text{dry weight of bark}}{\text{Fresh weight of wood and bark}}$$

$$r_i = \frac{\text{Dry weight of bark}}{\text{Dry weight of wood and bark}}$$

If any ratio R_i or r_i is ≥ 1 , or any value of the enti-

ties in the numerator or denominator is zero, the information from that subsample is ignored and treated as a missing value. Such a missing value is then estimated as an average of the two computed adjacent ratios.

A listing of the subroutine is given below.

```

SUBROUTINE RATIOS(N,MBH,DISKS,RMBH,R,IDB,RDB)
C   This subroutine calculates the ratios
C   ( dry-wt / fresh-weight ) and ( dry-bark/ dry-wt )
C   for sample disks
C   Variables
C   N - number of disks
C   N(1) - number of disks in stump
C          1st tree component
C   N(2) - number of disks in merchantable stem
C          2nd tree component
C   N(3) - number of disks in nonmerchantable stem
C          3rd tree component
C   N(4) - number of disks in nonmerchantable top
C          4th tree component
C   MBH - MBH sample disk, this disk is treated
C          separately because it can be anywhere
C          in merchantable stem or nonmerchantable
C          stem
C   MBH(1) - diameter over bark
C   MBH(2) - height
C   MBH(3) - fresh weight
C   MBH(4) - dry weight wood
C   MBH(5) - dry weight bark
C   DISKS - data from sample disks
C   DISKS(i,1,K) - i'th sample disk diameter
C                  over bark for k'th tree component
C   DISKS(i,2,K) - height
C   DISKS(i,3,K) - fresh weight
C   DISKS(i,4,K) - dry weight wood
C   DISKS(i,5,K) - dry weight bark
C   R - ratios
C   R(i,1,K) - i'th sample disk ( dry-wt / fresh-wt )
C              for k'th tree component
C   R(i,2,K) - ( dry-bark / dry-wt )
C   RMBH - ratios at MBH
C   RMBH(1) - ( dry-wt / fresh-wt )
C   RMBH(2) - ( dry-bark / dry-wt )
C   IDB - debug information
C   IDB(1) = i if R(i,1,K) > 1, 0 otherwise
C   IDB(2) = K if R(i,1,K) > 1, 0 otherwise
C   RDB - debug information
C   RDB = R(i,1,K) if R(i,1,K) > 1, 0 otherwise
C   Note assumption of only 1 error per tree
C
C   INTEGER N(4)
C   REAL MBH(5), DISKS(5,5,4), RMBH(2), R(5,2,4)
C   INTEGER IDB(2)
C   REAL RDB
C
C   REAL DOB, HT, FW, DWW, DWB, BUF(20,2)
C   INTEGER I, J, K, II
C
C   Initialize
C   IDB(1) = 0
C   IDB(2) = 0
C   RDB = 0.0
C   DO 10, K = 1, 4
C      DO 10, J = 1, 5
C         DO 10, I = N(K), 5
C            IF ( I .NE. N(K) ) THEN
C               DISKS(I,J,K) = 0.0
C            IF ( J .LE. 2 ) THEN
C               R(I,J,K) = 0.0
C            ENDIF
C         ENDIF
C      CONTINUE
C
C   Calculate R values
C
C   FW = MBH(3)
C   DWW = MBH(4)
C   DWB = MBH(5)
C   IF ( FW .EQ. 0.0 .OR. DWW .EQ. 0.0
C       .OR. DWB .EQ. 0.0 ) THEN
C      RMBH(1) = 0.0
C   ELSE
C      RMBH(1) = ( DWW + DWB ) / FW
C      IF ( RMBH(1) .GE. 1.0 ) RMBH(1) = 0.0
C   ENDIF
C   IF ( DWW .EQ. 0.0 .OR. DWB .EQ. 0.0 ) THEN
C      RMBH(2) = 0.0
C   ELSE
C      RMBH(2) = DWB / ( DWW + DWB )
C   ENDIF
C   DO 20, K = 1, 4
C      DO 20, I = 1, N(K)
C         FW = DISKS(I,1,K)
C         DWW = DISKS(I,4,K)
C         DWB = DISKS(I,5,K)
C         IF ( FW .EQ. 0.0 .OR. DWW .EQ. 0.0
C             .OR. DWB .EQ. 0.0 ) THEN
C            R(I,1,K) = 0.0
C         ELSE
C            R(I,1,K) = ( DWW + DWB ) / FW
C            IF ( R(I,1,K) .GE. 1.0 ) THEN
C               IDB(1) = I
C               IDB(2) = K
C            ENDIF
C         ENDIF
C      CONTINUE
C
C   END

```

```

      RDB = R(I,1,K)
      R(I,1,K) = 0.0
    ENDIF
  ENDIF
  IF ( DWW .EQ. 0.0 .OR. DWB .EQ. 0.0 ) THEN
    R(I,2,K) = 0.0
  ELSE
    R(I,2,K) = DWB / ( DWW + DWB )
  ENDIF
20  CONTINUE
C
C Estimate missing R's
C
DO 35, J = 1, 2
  II = 0
  DO 30, K = 1, 4
    DO 30, I = 1, N(K)
      II = II + 1
      BUF(II,J) = R(I,J,K)
30  CONTINUE
35  CONTINUE
  DO 50, J = 1, 2
    DO 50, I = 2, II - 1
      IF ( BUF(I,J) .EQ. 0.0 .AND.
X        BUF(I-1,J) .NE. 0.0 .AND.
X        BUF(I+1,J) .NE. 0.0 )
X      BUF(I,J) = ( BUF(I-1,J) + BUF(I+1,J) ) / 2
50  CONTINUE
  DO 60, J = 1, 2
    II = 0
    DO 65, K = 1, 4
      DO 65, I = 1, N(K)
        II = II + 1
        IF ( R(I,J,K) .EQ. 0.0 )
X          R(I,J,K) = BUF(II,J)
60  CONTINUE
65  CONTINUE
  RETURN
END

```

Subroutines STEM and NSTEM

The computations for the dry weight of bark and wood are handled separately for stem components and non-stem components.

Stem Components

These include the merchantable stem, the non-merchantable stem (≥ 2 cm dob), the stem top (< 2 cm dob), and the stump.

Subroutine STEM is used for calculating the desired biomass information by using the volume-weighted ratios R_i and r_i for the merchantable stem, the non-merchantable stem, and the top stem as defined by the user: The subroutine STEM also calculates the biomass information for the stump by using the simple unweighted ratios R_i and r_i .

The volume of merchantable and nonmerchantable stem sections is calculated by the Smalian formula:

$$V_i = \frac{\pi}{8} (DOB_i^2 + DOB_{i+1}^2)(H_{i+1} - H_i)$$

where (DOB_i, H_i) and (DOB_{i+1}, H_{i+1}) are the tree diameter outside bark and tree height at each end of the stem section. The summation $V = \sum V_i$ is the total volume of a specified tree component.

The volume-weighted ratios R_i and r_i are obtained as follows:

$$R_i = \sum \left[\left(\frac{R_i + R_{i+1}}{2} \right) V_i \right] / V$$

$$r_i = \sum \left[\left(\frac{r_i + r_{i+1}}{2} \right) V_i \right] / V$$

The estimate for the total dry weight (DW) of a tree component and its partition into dry weight of bark (DWB) and dry weight of wood (DWW) is computed from the total fresh weight (FW) as

$$DW = R_i (FW)$$

$$DWB = r_i (DW), \text{ and } DWW = DW - DWB$$

A listing of the STEM subroutine is given below.

```

SUBROUTINE STEMIN,STUMP,MBH,DHT,TH,RSTUMP,
X           RMBH,R,RAV,VOL,COMP,RC)
C
C This subroutine calculates dry weights and volumes
C of stem components of tree
C
C Parameters
C
C   N       - number of disks
C   N(1,j) - 2 * number of sample disks in jth
C             component
C   N(2,j) - number of sample disks used by subroutine
C             ( returned )
C
C   STUMP   - top stump sample disk
C   STUMP(1) - diameter over bark of top stump disk
C   STUMP(2) - height
C
C   MBH     - mbh disk
C   MBH(1) - diameter over bark of mbh disk
C   MBH(2) - height
C
C   DHT     - heights and diameters of sample disks
C   DHT(i,1) - i'th diameter over bark for kth component
C   DHT(i,2) - i'th height for kth component
C
C   TH      - total height of tree
C
C   RSTUMP  - ratios for stump
C   RSTUMP(1) - ( dry-wt / fresh-wt ) for top stump
C               sample disk
C   RSTUMP(2) - ( dry-bark / dry-wt )
C
C   RMBH   - ratios for mbh disk
C   RMBH(1) - ( dry-wt / fresh-wt )
C   RMBH(2) - ( dry-bark / dry-wt )
C
C   R      - ratios
C   R(i,1,K) - i'th disk ( dry-wt / fresh-wt ) for
C             kth component
C   R(i,2,K) - i'th disk ( dry-bark / dry-wt ) for
C             kth component
C
C   RAV    - average R
C   RAV(i,1,K) - average of R(i,1,K) and R(i+1,1,K)
C   RAV(i,2,K) - average of R(i,2,K) and R(i+1,2,K)
C
C   VOL    - volume
C   VOL(i,1) volume of section between i'th and
C             i+1'th disks on kth component
C
C   COMP   - weights and volumes of tree components
C   COMP(1,K) - fresh weight of kth tree component
C   COMP(2,K) - dry weight wood of kth tree component
C   COMP(3,K) - dry weight bark of kth tree component
C   COMP(4,K) - volume of kth tree component
C
C   RC     - return code
C   RC(K) - return code for kth component
C           = 0 if dry weights calculated
C           = 1 if component is missing
C           = 2 if insufficient data for component
C
C   INTEGER N(2,3)
C   REAL STUMP(2), MBH(2), DHT(7,2,3), TH
C   REAL RSTUMP(2), RMBH(2), R(7,2,3), RAV(6,2,3)
C   REAL VOL(6,3), COMP(4,3)
C   INTEGER RC(3)
C
C   REAL VP, CH
C
C Do merchantable stem
C
C   DHT(1,1,1) = STUMP(1)           : get top disk from stump
C   DHT(1,2,1) = STUMP(2)
C   R(1,1,1) = RSTUMP(1)
C   R(1,2,1) = RSTUMP(2)
C
C   DHT(2,1,1) = MBH(1)            : get disk taken at MBH
C   DHT(2,2,1) = MBH(2)
C   R(2,1,1) = RMBH(1)
C   R(2,2,1) = RMBH(2)
C
C   IF ( DHT(1,1,1) .LE. 0.1 .AND.
X     DHT(1,1,1) .EQ. 0.0 ) THEN
C
C     tree is a small tree
C
C     DO 10, I = 1, 7
C       R(I,1,1) = 0.0
C       R(I,2,1) = 0.0
C       IF ( I .GE. 7 ) GO TO 10
C       RAV(I,1,1) = 0.0
C       RAV(I,2,1) = 0.0
C       VOL(I,1) = 0.0
C
C     CONTINUE
C     COMP(2,1) = 0.0
C     COMP(3,1) = 0.0
10

```

```

COMP(4,1) = 0.0
RC(1,1) = 1
ELSE
  IF ( MBH(1) .LE. 0.1 ) THEN
    MGH not in merchantable stem
    DHT(2,1,1) = 0.0
    DHT(2,2,1) = 0.0
  ENDIF
  CALL VDRYWB(IN(1,1),DHT(1,1,1),R(1,1,1))
  RAV(1,1,1),VOL(1,1),COMP(1,1),RC(1)
ENDIF

Do nonmerchantable stem
IF ( N(2,1) .EQ. 0 ) N(2,1) = N(1,1)
DHT(1,1,2) = DHT(N(2,1),1,1)      ! get highest disk in
DHT(1,2,2) = DHT(N(2,1),2,1)      ! merchantable stem
R(1,1,2) = R(N(2,1),1,1)
R(1,2,2) = R(N(2,1),2,1)
IF ( DHT(1,1,2) .EQ. 0.0 .OR.
     DHT(1,2,2) .EQ. 0.0 ) THEN
  DHT(1,1,2) = STUMP(1)           ! use stump if small
  DHT(1,2,2) = STUMP(2)           ! tree
  R(1,1,2) = RSTUMP(1)
  R(1,2,2) = RSTUMP(2)
ENDIF
DHT(2,1,2) = MBH(1)
DHT(2,2,2) = MBH(2)
R(2,1,2) = RMBH(1)
R(2,2,2) = RMBH(2)
IF ( MBH(1) .GT. 0.1 .OR. MBH(1) .LT. 0.02 ) THEN
  MBH not in nonmerchantable stem
  DHT(2,1,2) = 0.0
  DHT(2,2,2) = 0.0
ENDIF
CALL VDRYWB(N(1,2),DHT(1,1,2),R(1,1,2))
  RAV(1,1,2),VOL(1,2),COMP(1,2),RC(2)

Do nonmerchantable top
DHT(1,1,3) = DHT(N(2,2),1,2)      ! get highest disk in
DHT(1,2,3) = DHT(N(2,2),2,2)      ! nonmerchantable stem
R(1,1,3) = R(N(2,2),1,2)
R(1,2,3) = R(N(2,2),2,2)
DHT(2,1,3) = 0.0
DHT(2,2,3) = 0.0
CALL VDRYWB(N(1,3),DHT(1,1,3),R(1,1,3))
  RAV(1,1,3),VOL(1,3),COMP(1,3),RC(3)
IF ( RC(3) .EQ. 0 ) THEN
  CH = TH - DHT(1,2,3),2,3)
  IF ( CH .GT. 0 ) THEN
    VP = 0.392699 * CH * DHT(N(2,3),1,3) ** 2
    COMP(4,3) = COMP(4,3) + VP
  ENDIF
ENDIF
RETURN
END
SUBROUTINE VDRYWB(N,DHT,R,RAV,VOL,COMP,RC)
C
  INTEGER N(2)
  REAL DHT(7,2), R(7,2), RAV(6,2), VOL(6), COMP(4)
  INTEGER RC
C
  INTEGER I, J
  REAL TEMP, RV(2)
  LOGICAL FLAG
C
  Initialize
C
  N(1) = MOD(N(1),7)
  IF ( N(1) .EQ. 0 ) N(1) = 7
  N(2) = N(1)
  DO 10, I = 1, 6
    RAV(I,1) = 0.0
    RAV(I,2) = 0.0
    VOL(I) = 0.0
  10 CONTINUE
  DO 20, I = N(1), 7
    IF ( I .EQ. N(1) ) GOTD 20
    DHT(I,1) = 0.0
    DHT(I,2) = 0.0
    R(I,1) = 0.0
    R(I,2) = 0.0
  20 CONTINUE
  IF ( DHT(1,1) .NE. 0.0 .AND. DHT(N(1),1) .NE. 0.0 .AND.
       DHT(1,2) .NE. 0.0 .AND. DHT(N(1),2) .NE. 0.0 .AND.
       N(1) .GE. 2 ) THEN
    Sufficient data
    RC = 0
    Sort disks by heights, throw out bad disks
    DO 30, I = 1, N(1) - 1
      DO 30, J = 1, N(1) - 1
        IF ( (DHT(J,2) > DHT(J+1,2)) .AND.
             (DHT(J+1,2) .NE. 0.0) .OR.
             (DHT(J,2) .EQ. 0.0) .OR.
             (DHT(J+1,2) .EQ. 0.0) ) THEN
          DO 25, K = 1, 2
            TEMP = DHT(J,K)
            DHT(J,K) = DHT(J+1,K)
            DHT(J+1,K) = TEMP
          TEMP = R(J,K)
          R(J,K) = R(J+1,K)
          R(J+1,K) = TEMP
        CONTINUE
      ENDIF
    CONTINUE
    DO 40, I = 1, N(1)
      IF ( DHT(I,1) .EQ. 0.0 .OR. DHT(I,2) .EQ. 0.0 ) THEN
        N(2) = N(2) - 1
      ENDIF
    CONTINUE
    DO 50, I = N(2), N(1)
      IF ( I .EQ. N(2) ) GOTD 50
      DHT(I,1) = 0.0
      DHT(I,2) = 0.0
    CONTINUE
    R(1,1) = 0.0
    R(1,2) = 0.0
    CONTINUE
    IF ( N(2) .GE. 2 ) THEN
      Enough good disks, estimate any missing R's
      DO 60, I = 2, N(2)
        DO 60, J = 1, 2
          IF ( R(I,J) .EQ. 0.0 ) R(I,J) = R(I,J)
          IF ( R(N(2),J) .EQ. 0.0 )
            R(N(2),J) = R(N(2)-1,J)
      60 CONTINUE
      DO 70, I = 2, N(2) - 1
        DO 70, J = 1, 2
          IF ( R(I,J) .EQ. 0.0 )
            R(I,J) = R(I-1,J) + R(I+1,J)
          IF ( R(I-1,J) .NE. 0.0 .AND.
               R(I+1,J) .NE. 0.0 ) THEN
            R(I,J) = R(I,J) / 2
          ENDIF
        CONTINUE
        FLAG = .TRUE.
        DO 80, I = 1, N(2)
          IF ( R(I,1) .EQ. 0.0 ) FLAG = .FALSE.
          IF ( R(I,2) .EQ. 0.0 ) FLAG = .FALSE.
        80 CONTINUE
        IF ( FLAG ) THEN
          Enough R data
          Calculate VDL, RAV for pairs of disks
          Calculate total volume of part and
          volume-weighted RV ratios for entire part
          COMP(4) = 0.0
          RV(1) = 0.0
          RV(2) = 0.0
          DO 90, I = 1, N(2) - 1
            RAV(I,1) = ( R(I,1) + R(I+1,1) ) / 2
            RAV(I,2) = ( R(I,2) + R(I+1,2) ) / 2
            VOL(I) = 0.392699 *
              ( DHT(I-1,1)**2 + DHT(I,1)**2 +
                ( DHT(I-1,2) - DHT(I,2) ) *
                ( DHT(I-1,1) + DHT(I,1) ) )
            COMP(4) = COMP(4) + VOL(I)
            RV(1) = RV(1) + RAV(I,1) * VOL(I)
            RV(2) = RV(2) + RAV(I,2) * VOL(I)
          90 CONTINUE
          RV(1) = RV(1) / COMP(4)
          RV(2) = RV(2) / COMP(4)
          COMP(3) = COMP(1) * RV(1) + RV(2)
          COMP(2) = COMP(1) * RV(1) - COMP(3)
        ELSE
          Insufficient data
          COMP(2) = 0.0
          COMP(3) = 0.0
          COMP(4) = 0.0
          RC = 2
        ENDIF
      ELSE
        Insufficient data
        COMP(2) = 0.0
        COMP(3) = 0.0
        COMP(4) = 0.0
        RC = 2
      ENDIF
    ELSE
      Insufficient data
      COMP(2) = 0.0
      COMP(3) = 0.0
      COMP(4) = 0.0
      RC = 2
    ENDIF
  RETURN
END

```

Nonstem Components

These include the large live branches, the small live branches including foliage, the dead branches, and the cones.

Subroutine NSTEM computes dry weights of bark and wood for live branches, dry weight of wood for dead branches, and dry weight of cones. Live branches are grouped into branches ≥ 2 cm dbh and branches < 2 cm dbh. Size limits are arbitrary and can be adjusted for any other specification. Small branches can be further split into groups of dry weight of bark and wood and groups of dry weight of foliage.

Computations are similar to those described under the STEM subroutine; the main difference is that no weighting is done for nonstem components and only the simple ratio averages of R_i and r_i are used in computing biomass weights.

A listing of this subroutine is given below.

```

SUBROUTINE NSTEMIN(SW,R,PART)
This subroutine calculates dry weights
of nonstem components of the tree. There
is no volume estimation for these
components.

Nonstem components:
Large live branches
Small live branches + foliage
Dead branches
Cones

For large and small live branches
the dry weight of wood and dry weight of bark
or needles are calculated. For dead branches
and cones only total dry weight is calculated.

Parameters
N - number of samples
N(1) - number of samples for stump
    1st tree component
N(2) - number of samples for large
    branches + 2nd tree component
N(3) - . . . small branches + 3rd tree component
N(4) - . . . dead branches + 4th tree component
N(5) - . . . cones ( 5th tree component )

SW - sample weights
SW(i,1,k) - i-th sample fresh-wt for kth
    tree component
SW(i,2,k) - i-th sample dry-wt wood or
    sample dry-wt for kth tree component
SW(i,3,k) - i-th sample dry-wt bark for
    kth tree component ( not always used )

R - ratios
R(i,1,k) - i-th sample + dry-wt / fresh-wt
    for kth tree component
R(i,2,k) - i-th sample + dry-bark / dry-wt
    for kth tree component

PART - weights of tree components
PART(1,k) fresh weight of kth tree component
PART(2,k) dry weight wood or dry weight
    of kth tree component
PART(3,k) dry weight bark of kth tree
    component or unused

Subroutines called:
DRYWB
DRYW

INTEGER N(5)
REAL SW(5,3), R(5,2,5), PART(3,5)

Do stump
CALL DRYWB(N(1),SW(1,1,1),R(1,1,1),PART(1,1))

Do large branches
CALL DRYWB(N(2),SW(1,1,2),R(1,1,2),PART(1,2))

Do small branches
CALL DRYWB(N(3),SW(1,1,3),R(1,1,3),PART(1,3))

Do dead branches
CALL DRYWB(N(4),SW(1,1,4),R(1,1,4),PART(1,4))

Do cones
CALL DRYWB(N(5),SW(1,1,5),R(1,1,5),PART(1,5))
RETURN
END
SUBROUTINE DRYWIN(SW,R,PART)
INTEGER N
REAL SW(5,3), R(5,2), PART(3)

INTEGER I, J, NR
REAL RAV(2)

Calculate R
NR = N
DO 20, I = 1, N
  IF ( SW(I,1) .EQ. 0.0 .OR. SW(I,2) .EQ. 0.0
      .OR. SW(I,3) .EQ. 0.0 ) THEN
    R(I,1) = 0.0
    R(I,2) = 0.0
    NR = NR - 1
  ELSE
    R(I,1) = ( SW(I,2) + SW(I,3) ) / SW(I,1)
    R(I,2) = SW(I,3) / ( SW(I,2) + SW(I,3) )
    IF ( R(I,1) .GE. 1.0 ) THEN
      R(I,1) = 0.0
      R(I,2) = 0.0
      NR = NR - 1
    ENDIF
  ENDIF
CONTINUE
Calculate average R
RAV(1) = 0.0
RAV(2) = 0.0
IF ( NR .NE. 0 ) THEN
  DO 40, J = 1, 2
    DO 40, I = 1, N
      RAV(J) = RAV(J) + R(I,J)
    CONTINUE
    RAV(1) = RAV(1) / NR
    RAV(2) = RAV(2) / NR
  ENDIF
END

```

```

ENDIF
Calculate dry weights
PART(3) = PART(1) * RAV(1) * RAV(2)
PART(2) = PART(1) * RAV(1) - PART(3)
RETURN
END
SUBROUTINE DRYWIN(SW,R,PART)
INTEGER N
REAL SW(5,2), R(5), PART(2)
INTEGER I, J, NR
REAL RAV
Calculate R
NR = N
DO 20, I = 1, N
  IF ( SW(I,1) .EQ. 0.0 .OR.
      SW(I,2) .EQ. 0.0 ) THEN
    R(I) = 0.0
    NR = NR - 1
  ELSE
    R(I) = SW(I,2) / SW(I,1)
    IF ( R(I) .GE. 1.0 ) THEN
      R(I) = 0.0
      NR = NR - 1
    ENDIF
  ENDIF
CONTINUE
Calculate average R
RAV = 0.0
IF ( NR .NE. 0 ) THEN
  DO 40, I = 1, N
    RAV = RAV + R(I)
  CONTINUE
  RAV = RAV / NR
ENDIF
Calculate dry weights
PART(2) = PART(1) * RAV
RETURN
END

```

REFERENCES

Alemdag, I.S. 1980. Manual of data collection and processing for the development of forest biomass relationships. Environ. Can., Can. For. Serv. Petawawa Nat. For. Inst. Chalk River, Ontario. Inf. Rep. PI-X-4.

Singh, T. 1982. Biomass equations for ten major tree species of the prairie provinces. Environ. Can., Can. For. Serv., North. For. Res. Cent. Edmonton, Alberta. Inf. Rep. NOR-X-242.

ACKNOWLEDGMENTS

The assistance of N. Leenders, C. Martin, and S. Lux in the formulation, writing, and testing of the preliminary programs is gratefully acknowledged.

T. Singh
D. Campbell
November 1983

APPENDIX 1

Listed below are sample programs used for driving the subroutines RATIOS, STEM, and NSTEM at NoFRC.

C C This program drives the subroutine RATIOS

```

INTEGER TREEN, N(4), IDB(2), I, J, K, II
REAL MBH(5), DISCS(5,5,4), RMBH(2), R(5,2,4), RDB
DATA N / 1,4,3,1 /
OPEN (UNIT=1,FILE='B10B1.DAT',RECORDSIZE=410,STATUS='OLD')
OPEN (UNIT=2,FILE='BIGR81.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=3,FILE='TMP1.TMP1',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=4,FILE='SMALLRB1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
CONTINUE
READ (1,400,END=999) TREEN, (DISCS(I,J,1), J=1,5),
  (MBH(I,J), J=1,5), (DISCS(I,J,2), J=1,5), (I=1,4)
  (DISCS(I,J,3), J=1,5), (I=1,3)
  (DISCS(I,J,4), J=1,5)
CALL RATIOSIN(MBH,DISCS,RMBH,R,IDB,RDB)
IF (IDB(1,1).NE.0) THEN
  II = 0
  DO 30, K = 1, IDB(2)
    II = II + N(K)
30  CONTINUE
  II = II - IDB(1,1) - N(IDB(2))
  IF (IDB(2).GT.1) II = II + 1
  WRITE (3,127) TREEN, DISCS(IDB(1,1),4, IDB(2)),
    DISCS(IDB(1,1),5, IDB(2)), DISCS(IDB(1,1),3, IDB(2)),
    II, RDB
ENDIF
WRITE (2,410) TREEN,R(1,1,1,1),RMBH(1),
  (R(1,1,2), I = 1, 4),
  (R(1,1,3), I = 1, 3 ), (R(1,1,4)
  WRITE (4,410) TREEN,R(1,2,1),RMBH(2),
  (R(1,2,2), I = 1, 4 ),
  (R(1,2,3), I = 1, 3 ), (R(1,2,4)
GOTO 1
999 CONTINUE
127 FORMAT (1X,I3,3(2X,F5.1),2X,I3,2X,F7.4)
400 FORMAT (13,77,4(F4.3,F4.2,3F5.1),2(F3.3,F4.2,3F5.1),
  X (F3.3,F4.2,2F5.1,F4.1,-3(F3.3,F4.2,3F4.1))
410 FORMAT (I3,10(2X,F7.6))
420 FORMAT (1X,13,5(2(X,F6.5),F8.4))
430 FORMAT (2X,13, ' is a small tree.')
432 FORMAT (2X,13, ' has insufficient data.')
440 FORMAT (1X,13,12(1X,F6.5),3F11.5)
441 FORMAT (1X,13,84X,3F11.5)
450 FORMAT (1X,13,4(2(2X,F6.5),F8.4))
460 FORMAT (1X,13,10(1X,F6.5),3F11.5)
465 FORMAT (1X,13,10F12.4)
470 FORMAT (1X,13,2(2X,F6.5))
480 FORMAT (1X,13,3F11.5)
STOP
END

```

C C This program drives the subroutine STEM

```

INTEGER TREEN, N(2,3)
REAL TH, STUMP(2), MBH(2), DHT(7,2,3)
REAL RSTUMP(2), RMBH(2), R(7,2,3)
REAL RAV(6,2,3), VOL(6,3), COMP(4,3)
INTEGER RC(3)

C INTEGER I, J, K
C
C Initialize
C
C DATA N/6,0,5,0,3,0/
C
C Open files
C
OPEN (UNIT=1,FILE='SMALLRB1.TEJ',STATUS='OLD')
OPEN (UNIT=2,FILE='BIGR81.TEJ',STATUS='OLD')
OPEN (UNIT=3,FILE='B10B1.DAT',RECORDSIZE=410,
      STATUS='OLD')
OPEN (UNIT=4,FILE='TMP1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=7,FILE='RES1B1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=8,FILE='TMP2B1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=9,FILE='RES1B1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=10,FILE='RES1B1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=11,FILE='TMP3B1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=12,FILE='RES1B1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')

C Read and process data until end of file
C
CONTINUE
READ (3,400,END=999) TREEN, TH, (COMP(I,K), K = 1, 3),
  STUMP, MBH, (DHT(I,J,1), J = 1, 2 ), I = 3, 6

```

```

  ( DHT(3,J,2), J = 1, 2 ), I = 3, 5 )
  READ (2,410) RSTUMP(1), RMBH(1), (R(1,1,1), I=3,6),
  (R(1,1,2), I=3,5), (R(3,1,3)
  READ (1,410) RSTUMP(2), RMBH(2), (R(1,2,1), I=3,6),
  (R(1,2,2), I=3,5), (R(3,2,3)
  CALL STEMINT(STUMP,MBH,DHT,TH,RSTUMP,RMBH,R,RAV,
  VOL,COMP,RC)
  IF (RC(1).EQ.0) THEN
    WRITE (4,430) TREEN,(RAV(1,2,1),RAV(1,1,1),VOL(1,1), I=1,5)
    WRITE (7,440) TREEN,(R(1,2,1),I=1,6),(R(1,1,1),I=1,6),
    COMP(4,1), COMP(3,1), COMP(2,1)
  ELSE
    IF (RC(1).EQ.1) THEN
      WRITE (4,431) TREEN
      WRITE (7,441) TREEN,O,0,0,0,0,0
    ELSE
      WRITE (4,432) TREEN
      WRITE (7,432) TREEN
    ENDIF
  ENDIF
  IF (RC(2).EQ.0) THEN
    WRITE (9,450) TREEN,(RAV(1,2,2),RAV(1,1,2),VOL(1,2), I=1,4)
    WRITE (10,460) TREEN,(R(1,2,2),I=1,5),(R(1,1,2),I=1,5),
    COMP(4,2), COMP(3,2), COMP(2,2)
    WRITE (18,465) TREEN,( DHT(1,J,2), J = 1, 2 ), I = 1, 5 )
  ELSE
    WRITE (9,432) TREEN
    WRITE (10,432) TREEN
    WRITE (18,432) TREEN
  ENDIF
  IF (RC(3).EQ.0) THEN
    WRITE (11,470) TREEN,RAV(1,2,3),RAV(1,1,3)
    WRITE (12,480) TREEN,COMP(4,3),COMP(3,3),COMP(2,3)
  ELSE
    WRITE (11,432) TREEN
    WRITE (12,432) TREEN
  ENDIF
GOTO 1
999 CONTINUE
400 FORMAT ( 13,6X,F4.2,T42,F6.2,F5.2,F4.2,
  X (177,4(F4.3,F4.2,15X),2(F3.3,F4.2,15X),
  X (F3.3,F4.2,14X,3(F3.3,F4.2,12X))
410 FORMAT (3X,10(2X,F7.6))
430 FORMAT (1X,13,5(2(X,F6.5),F8.4))
431 FORMAT (2X,13, ' is a small tree.')
432 FORMAT (2X,13, ' has insufficient data.')
440 FORMAT (1X,13,12(1X,F6.5),3F11.5)
441 FORMAT (1X,13,84X,3F11.5)
450 FORMAT (1X,13,4(2(2X,F6.5),F8.4))
460 FORMAT (1X,13,10(1X,F6.5),3F11.5)
465 FORMAT (1X,13,10F12.4)
470 FORMAT (1X,13,2(2X,F6.5))
480 FORMAT (1X,13,3F11.5)
STOP
END

```

C C This program drives the subroutine NSTEM

```

INTEGER TREEN, N(5)
REAL SW(5,3,5), R(5,2,5), PART(3,5)
INTEGER I, J, K
DATA N / 1,3,3,3,1 /
OPEN (UNIT=1,FILE='B10B1.DAT',RECORDSIZE=410,
      STATUS='OLD')
OPEN (UNIT=2,FILE='RES1B1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=3,FILE='RESV81.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=4,FILE='RESV181.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=7,FILE='RESV1B1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
OPEN (UNIT=8,FILE='RES0WB1.TEJ',CARRIAGECONTROL='LIST',
      STATUS='NEW')
1  CONTINUE
  READ (1,400,END=999) TREEN, (PART(1,K), K = 1, 5),
  ( ( SW(I,J,1), J = 1, 3 ),
  ( ( SW(I,J,K), J = 1, 3 ), I = 1, 3 ), K = 2, 3 ),
  ( ( SW(I,J,4), J = 1, 2 ), I = 1, 3 ),
  ( ( SW(I,J,5), J = 1, 2 )
  CALL NSTEMIN(SW,R,PART)
  WRITE (2,600) TREEN,( ( R(1,1,2), I = 1, 3 ), J = 1, 2 ),
  (PART(3,2),PART(2,2)
  WRITE (3,600) TREEN,( ( R(1,1,3), I = 1, 3 ), J = 1, 2 ),
  (PART(3,3),PART(2,3)
  WRITE (4,610) TREEN,( ( R(1,1,4), I = 1, 3 ), PART(2,4)
  WRITE (7,620) TREEN, PART(2,5)
  WRITE (18,620) TREEN, PART(3,1), PART(2,1)
GOTO 1
999 CONTINUE
400 FORMAT (13,T37,F5.2,T57,4F5.2,T85,3F5.1,
  X (191,100X,100X,3(3F4.1),3(F5.1,2F4.1),
  X (T66,100X,100X,3(2F5.1),2F5.1)
600 FORMAT (1X,I3,6(2X,F6.5),2F11.5)
610 FORMAT (1X,I3,3(2X,F7.5),F11.5)
620 FORMAT (1X,I3,2F11.5)
STOP
END

```