GUIDE D'UTILISATION DU LOGICIEL emme/2

Gouvernement du Québec
Ministère des Transports
Direction générale du transport terrestre des personnes

GUIDE D'UTILISATION DU LOGICIEL emme/2

Jean-Pierre Primeau ing.
Directeur de projet

Novembre 1984

DIRECTEUR

Jean-Pierre Primeau, ing.

COORDONNATEUR

Michael Florian, C.R.T.

RÉDACTION

André Babin, C.R.T.

Linda James-Lefebvre, C.R.T.

RÉVISION

Pierre Tremblay, ing.

TRAITEMENT DE TEXTES

Nicole Audet

PRÉAMBULE

Ce document se veut un guide d'utilisation du logiciel emme/2 (1), tel qu'implanté sur l'ordinateur de la RAAQ pour le Ministère des transports du Québec (M.T.Q.). Il résume l'état de la situation, au moment de sa publication, des procédures d'accès et de création des banques de données relatives à emme/2 . Des mises-à-jour périodiques sont à prévoir, au fur à mesure qu'évolue l'environnement informatique dans lequel le logiciel est installé; il appartient au lecteur de s'assurer, auprès du Service des systèmes d'information du transport des personnes de la D.G.T.T.P.(2), que le présent document constitue la version la plus récente du guide d'utilisation.

Le logiciel est installé de façon à pouvoir être facilement utilisé par tous les gens du Ministère des transports. À cet effet, les procédures ont été paramétrisées pour permettre l'identification rapide du compte et du projet de l'utilisateur. De même, le logiciel exécutable de emme/2 réside sur des fichiers à accès libre pour tous les usagers.

On doit considérer comme prérequis à la lecture et à la bonne compréhension de ce document des notions solides du système TSO/SPF, ainsi qu'une connaissance adéquate du logiciel emme/2 . À cet effet, le lecteur pourra consulter le manuel de l'usager d' emme/2 (User's Manual), disponible à la D.G.T.T.P.

La D.G.T.T.P. tient à remercier la Direction des systèmes de gestion pour son support et sa précieuse collaboration tout au long de l'implantation du logiciel emme/2 .

(1) : Equilibre Multimodal-Multimodal Equilibrium, développé au Centre de Recherche sur les Transports de l'Université de Montréal, Coordonnateur: Michael Florian.

(2) D.G.T.T.P: Direction générale du transport terrestre des personnes.

liste des révisions

Numéro	Date	Détail
0	6 novembre 1984	Texte de base

TABLE DES MATIERES

TABLE DES MATIÈRES

PAGE

PAGE

PAGE

LISTE DES TABLEAUX

LISTE DES TABLEAUX

PAGE

LISTE DES TABLEAUX

SECTION 1

AUTORISATION D'ACCÈS À emme2

Bien que l'accès à l'utilisation de emme/2 ne fasse l'objet d'aucune protection spéciale, la D.G.T.T.P. suggère aux usagers éventuels de consulter les responsables de l'implantation de ce logiciel avant d'entreprendre des travaux comportant des opérations de création ou de destruction de banques de données. Ces gens pourront au besoin assister tout usager dans ce genre de tâche. Il est d'autre part entendu que l'accès aux programmes-sources de emme/2 est prohibé.

1.1 Conditions préalables

Pour qu'un usager ait accès au logiciel emme/2 , il suffit que son profil d'usager lui permette l'accès à un volume de disque et au logiciel "TSO".

Pour le reste du présent texte, on établira la convention suivante:

i) le symbole "préfixe" correspond au numéro de compte informatique de l'usager suivi du numéro de projet alloué à l'usager.
(exemple: A017.P0108)
(Voir la section 2.3.2)

ii) le symbole "ZZZ" correspond au numéro d'usager de l'utilisateur.
(exemple: logonid = nu188 alors ZZZ est 188)

1.2 Fichier de communication

Au début de chaque session de travail, emme/2 créera temporairement, pour la durée de cette session, le fichier partionné **préfixe.EMPROZZZ** , où **ZZZ** correspond au numéro de l'usager en liaison.

Ce fichier permettra la communication entre les divers modules d' emme/2 au cours de la session et est transparent à l'usager.

Il faut par contre faire attention de ne pas utiliser ce nom de fichier à d'autres fins, puisqu'il serait automatiquement détruit par emme/2 à l'initialisation de la prochaine session.

SECTION 2

ACCÈS À emme2, AVEC LA PROCÉDURE EMME2

Nous décrirons ici la procédure générale "EMME2" qui permet l'accès au logiciel lui-même. Cette procédure TSO peut et devrait normalement être appelée par des procédures plus spécifiques, préparées par l'usager pour son projet particulier, afin d'accélérer l'accès à emme/2 . En effet, la plupart des paramètres substituables seront invariables pour une banque donnée dans le contexte d'un projet donné, de sorte que l'appel à cette banque pourra se faire par une seule commande, comme nous le verrons à la section 3.4 plus loin.

2.1 **Appel de la procédure EMME2**

Afin de démarrer la procédure d'accès à emme/2 , l'usager doit, une fois la session TSO active, demander l'exécution de la procédure d'initialisation générale:

EXEC 'EMME2.EMPROC(EMSETUP)'

Cette initialisation sera valide tant que la session TSO durera, peu importe le nombre de fois où EMME2 est appelée.

À l'intérieur de la session TSO donc, à chaque fois que l'on voudrait accéder à une banque de données sous emme/2 , on exécute la procédure EMME2 en entrant sur une seule ligne:

i) pour une banque existante:
EMME2 USERPREF(préfixe) VOLSER(volume)
BANK(nomdebanque) BANKTOT(btot) BANKSP(bsp)
BANKINCR(bincr)

ii) Pour une banque qu'on désire créer:
EMME2 NNN(001) USERPREF(préfixe) VOLSER(volume)
BANKDISP(NEW) BANK(nomdebanque) BANKTOP(btot) BANKSP(bsp)
BANKINCR(bincr) BATCHLIB(fichierpartitionnébatch)

Nous décrirons plus loin chacun de ces paramètres.

2.2 Contenu de la procédure EMME2

Le tableau 2-1 reproduit intégralement le contenu de la procédure générale EMME2 qui sert à appeler le logiciel emme/2 . La procédure est abondamment commentée et contient des valeurs par défaut pour la plupart des paramètres substituables.

tableau 2-1

procédure générale EMME 2

```
PROC /*EMME2*/ 0 NNN(002)-
     USERPREF('A017.P0108') VOLSER(MTQ302)-
     BANK() BANKDISP(OLD) BANKTOT(380) BANKSP(300) BANKINCR(40) MAP()-
     PUNCHSP(0)  PRINTSP(1)  PLOTSP(0)-
     BATCHLIB() D002(D002IN) D141(D141IN) D201(D201IN) D202(D202IN)-
     D211(D211IN) D221(D221IN) D231(D231IN)-
     D301(D301IN) D311(D311IN) D411(D411IN)-
     USERPROC('A017.P0108.EMPROCX')-
     EMPREF(EMME2) EMLIB(EMLIB) EMOBJ(EMOBJ) EMLOAD(EMLOAD)
CONTROL MAIN /*LIST    SYMLIST
/* PROCEDURE TO EXECUTE EMME2
/*     84 06 08  LGJL
/* CALLING SEQUENCE:
/*     EMME2
/*
/* OPTIONAL PARAMETERS ARE SPECIFIED AS PARAMETER(VALUE) FOLLOWING
/* EMME2   (SEE EXAMPLES IN THE PROC STATEMENT)   IN THE PARAMETER
/* DESCRIPTIONS, ZZZ INDICATES THE USER NUMBER TAKEN FROM THE USER
/* ID NUZZZ, XXX.YYY INDICATES THE USER DSNAME PREFIX TAKEN FROM
/* PARAMETER USERPREF, EMME2 INDICATES THE SYSTEM PREFIX TAKEN FROM
/* PARAMTER EMPREF.
/*
/* OPTIONAL PARAMETERS:
/*     NNN       MODULE NUMBER     3 DIGITS. USE ONLY WHEN CREATING A
/*                                 NEW DATA BASE ( NNN(001) ).
/*     USERPREF USER PREFIX        USERPREF('XXX.YYY') USED TO FORM FIRST
/*                                 PART OF DSNAME OF ALL FILES EXCEPT
/*                                 EMME/2 SYSTEM FILES.
/*                                 E.G. XXX.YYY.EMPRTZZZ
/*     VOLSER   USER VOLUME        VOLUME SERIAL USED FOR ALL FILES EXCEPT
/*              SERIAL             EMME/2 SYSTEM FILES.
/*     BANK     DATA BANK FILE     BANK(XXXXXX) USED TO FORM DATA BANK
/*                                 FILE NAME:  XXX.YYY.XXXXXX
/*     BANKDISP DISP OF BANK       DATA BANK FILE MUST BE ALLOCATED BY
/*                                 USING BANKDISP(NEW) IN THIS PROC.
/*     BANKTOT  BANK SPACE         NUMBER OF BLOCKS (4628) FOR BANK
/*     BANKSP   PRIMARY SPACE      NUMBER OF BLOCKS (4628) PRIMARY
/*                                 ALLOCATION FOR BANK
/*     BANKINCR SECONDARY SP       NUMBER OF BLOCKS (4628) SECONDARY
/*                                 ALLOCATION FOR BANK
/*                                 NOTE: BANKTOT MUST BE LESS OR EQUAL TO
/*                                 BANKSP + 15*BANKINCR
/*     PUNCHSP  PUNCH SPACE        NUMBER OF BLOCKS (9040) FOR PRIMARY
/*                                 ALLOCATION FOR PUNCH FILE
/*                                 XXX.YYY.EMPCHZZZ, DELETED
/*                                 AND ALLOCATED AS NEW. SECONDARY ALLOC
/*                                 IS 5.
/*                                 PUNCHSP(0) WILL ALLOCATE A DUMMY
/*                                 FILE.
/*     PRINTSP  PRINTER SPACE      NUMBER OF BLOCKS (9044) FOR PRIMARY
/*                                 ALLOCATION OF EMME/2 PRINTER FILE
/*                                 XXX.YYY.EMPRTZZZ, DELETED AND
/*                                 ALLOCATED AS NEW. SECONDARY ALLOC 5.
/*                                 PRINTSP(0) WILL ALLOCATE A DUMMY
/*                                 FILE.
/*     PLOTSP   PLOT SPACE         NUMBER OF BLOCKS (8919) FOR PRIMARY
/*                                 ALLOCATION OF EMME/2 PLOT FILE
/*                                 XXX.YYY.EMPLTZZZ, DELETED AND
/*                                 ALLOCATED AS NEW. SECONDARY ALLOC 5.
/*                                 PLOTSP(0) WILL ALLOCATE A DUMMY FILE
/*     BATCHLIB BATCH INPUT  NAME OF PARTITIONED FILE CONTAINING
/*              LIBRARY      BATCH INPUT DATA
/*     D002     BATCH INPUT  MEMBER NAME OF BATCH INPUT TO MODULE
/*                           002 (DEVICE TABLE)
/*     D141 ETC              MEMBER NAME OF BATCH INPUT TO MODULE
/*                           141 ETC  (201,202,211,221,231,301,311,
/*                           411)
/*     EMPREF   EMME2 PREFIX PREFIX USED TO FORM FIRST PART
/*                           OF THE DSNAME OF EMME/2 SYSTEM FILES.
/*     EMLIB    SUBROUTINE LIB  EMLIB(XXX) USED TO FORM EMME/2
/*                           SUBROUTINE LIBRARY NAME: EMME2.XXX
/*     EMOBJ    COMPILED     EMOBJ(XXX) USED TO FORM EMME/2 COMPILED
/*              MODULES      MODULE LIBRARY NAME: EMME2.XXX
/*     EMLOAD   EXECUTABLE   EMLOAD(XXX) USED TO FORM EXECUTABLE
/*                MODULES      MODULE LIBRARY NAME: EMME2.XXX
/*
/*
/* EMME/2 USES THE FOLLOWING UNIT NUMBERS:
/*
/*    1  DATA BANK
/*    3  BATCH INPUT FILE
/*    4  PUNCH FILE
/*    5  DIALOG INPUT (NORMALLY FROM TERMINAL)
/*    6  DIALOG OUTPUT  (NORMALLY TO TERMINAL)
/*    7  PRINTER FILE
/*    8  PLOT FILE
/*   10  ERROR MESSAGE FILE
/*   98  BATCH MODE OR INTERACTIVE MODE INDICATOR FILE
/*   99  NEXT MODULE FILE
/*
/* EMME/2 SYSTEM FILES ARE:
/*       EMLIB   SUBROUTINE LIBRARY
/*       EMOBJ   COMPILED EMME/2 MODULE LIBRARY
/*       EMLOAD  LOAD MODULE LIBRARY
/*       EMPROC  TSO PROCEDURE LIBRARY
/*
IF &BANKTOT GT &BANKSP + 15 * &BANKINCR THEN DO
   WRITE *** BANKTOT(&BANKTOT) GREATER THAN BANKSP + 15 * BANKINCR
   EXIT
   END
SET USER = &SUBSTR(3:5,&SYSUID)
/*
/*   ATTACH PROCEDURE LIBRARY FOR THIS USER
/*
FREE ATTR(NEXTDCB) FILE(FT99F001)
ATTRIB NEXTDCB RECFM(F B) LRECL(80) BLKSIZE(2960) DSORG(PO)
DEL '&USERPREF..EMPRO&USER'
ALLOCATE FILE(FT99F001) DA('&USERPREF..EMPRO&USER') NEW-
   VOLUME(&VOLSER) UNIT(3380) SPACE(1,1) TRACKS DIR(1) USING(NEXTDCB)
FREE FILE(FT99F001)
FREE FILE(SYSPROC)
ALLOCATE FILE(SYSPROC) SHR-
   DA('SYS1.SYSPROC' '&EMPREF..EMPROC' '&USERPREF..EMPRO&USER')
/*
/*   FREE FILES AND ATTRIBUTE LISTS (DCBS)
/*
FREE ATTR(BANKDCB) FILE(FT01F001)
FREE ATTR(TKIDCB) FILE(FT05F001 FT06F001)
FREE ATTR(LDADCB) FILE(FT04F001)
FREE ATTR(PRINTDCB) FILE(FT07F001)
FREE ATTR(PLOTDCB) FILE(FT08F001)
FREE FILE(FT10F001 FT98F001)
ATTRIB BANKDCB RECFM(F) LRECL(4628) BLKSIZE(4628) DSORG(PS) BUFNO(1)
ATTRIB TKIDCB RECFM(F A) LRECL(132) BLKSIZE(132)
ATTRIB LDADCB RECFM(F B) LRECL(80) BLKSIZE(9040)
ATTRIB PRINTDCB RECFM(F B) LRECL(133) BLKSIZE(9044)
ATTRIB PLOTDCB RECFM(V B S) LRECL(182) BLKSIZE(8919)
/*
/*  DATA BANK
/*
ALLOCATE FILE(FT01F001) DA('&USERPREF..&BANK') &BANKDISP-
     VOLUME(&VOLSER)-
     UNIT(3380) SPACE(&BANKSP,&BANKINCR) BLKSIZE(4628) USING(BANKDCB)
/*
/*  DIALOG INPUT
/*
ALLOCATE FILE(FT05F001) DA(*) USING(TKIDCB)
/*
/*  DIALOG OUTPUT
/*
ALLOCATE FILE(FT06F001) DA(*) USING(TKIDCB)
/*
/*  PRINTER FILE
/*
IF &PRINTSP NE 0 THEN DO
   DEL '&USERPREF..EMPRT&USER'
   ALLOCATE FILE(FT07F001) DA('&USERPREF..EMPRT&USER') NEW-
      VOLUME(&VOLSER) UNIT(3380) SPACE(&PRINTSP,5) BLKSIZE(9044)
   FREE FILE(FT07F001)
   ALLOCATE FILE(FT07F001) DA('&USERPREF..EMPRT&USER') MOD-
      BLKSIZE(9044) USING(PRINTDCB) RELEASE
   END
ELSE-
     ALLOCATE FILE(FT07F001) DSNAME('NULLFILE') BLKSIZE(9044)-
        USING(PRINTDCB)
/*
/*  ERROR FILE
/*
ALLOCATE FILE(FT10F001) DATASET(*)
/*
/*  FILE TO COMMUNICATE TO NEXT MODULE
/*
ALLOCATE FILE(FT99F001) DA('&USERPREF..EMPRO&USER(NEXT)') OLD
/*
/*  BATCH OUTPUT OR PUNCH FILE
/*
IF &PUNCHSP GT 0 THEN DO
     DEL '&USERPREF..EMPCH&USER'
     ALLOC FILE(FT04F001) DATASET('&USERPREF..EMPCH&USER')-
     NEW KEEP CATALOG VOLUME(&VOLSER)-
     UNIT(3380) SPACE(&PUNCHSP,5) BLKSIZE(9040)
     FREE FILE(FT04F001)
     ALLOC FILE(FT04F001) DATASET('&USERPREF..EMPCH&USER')-
     MOD BLKSIZE(9040) USING(LDADCB) RELEASE
     END
ELSE-
     ALLOCATE FILE(FT04F001) DUMMY BLKSIZE(9040) USING(LDADCB)
```

tableau 2-1 (suite)

```
/*
/*  PLOT FILE
/*
IF &PLOTSP GT 0 THEN DO
     DEL '&USERPREF..EMPLT&USER'
     ALLOC FILE(FT08F001) DATASET('&USERPREF..EMPLT&USER')-
     NEW KEEP CATALOG VOLUME(&VOLSER)-
     UNIT(3380) SPACE(&PLOTSP,5) BLKSIZE(8918)
     FREE FILE(FT08F001)
     ALLOC FILE(FT08F001) DATASET('&USERPREF..EMPLT&USER')-
     MOD BLKSIZE(8918) USING(PLOTDCB) RELEASE
     END
ELSE-
     ALLOCATE FILE(FT08F001) DUMMY BLKSIZE(8918) USING(PLOTDCB)
/*
/* SET TO INTERACTIVE MODE
/*
ALLOC FILE(FT98F001) SPACE(1) BLKSIZE(9040) USING(LDADCB)
OPENFILE FT98F001 OUTPUT
SET &FT98F001 = 0      /* 0= INTERACTIVE
PUTFILE FT98F001
CLOSFILE FT98F001
/*
/* SET TO CORRECT BDISK ROUTINE IF NEW DATA BANK
/*
SET &BDISK = &STR( )
IF &BANKDISP EQ OLD THEN GOTO L1
SET &DSIZE = 2700
IF &BANKTOT LT 2700 THEN SET &DSIZE = 2600
IF &BANKTOT LT 2600 THEN SET &DSIZE = 1800
IF &BANKTOT LT 1800 THEN SET &DSIZE = 380
IF &DSIZE NE 2700 THEN SET &BDISK = ,'&EMPREF..&EMLIB(BDSK&DSIZE)'
/*
/* INITIALIZE LIBS USED INDICATOR
/*
L1:SET LIBS = &STR(N)
/*
/* INITIALIZE MODULE NUMBER
/*
SET MODNUM = &NNN
SET ZEROS = &STR(0000)
/*
/* LOOP TO EXECUTE EMME/2 MODULES AS SPECIFIED ON MENU.
/*
/* LINK AND CALL ARE ONLY USED FOR A NEW DATA BANK TO USE CORRECT
/* BDISK ROUTINE
/*
DO WHILE &MODNUM GE 000 AND &MODNUM LT 999
   SET &MODTEX = &SUBSTR(1:&EVAL(4-&LENGTH(&MODNUM)),&ZEROS)&MODNUM
   SET &MODTEX = &SUBSTR(2:4,&MODTEX)
/*
/* SET UP ANY BATCH FILES SPECIFIED
/*
SET FT03 = &STR(N)
IF &LENGTH(&BATCHLIB) EQ 0 THEN GOTO L3
IF &MODTEX EQ 002 AND &LENGTH(&D002) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D002)') SHR
     GOTO L2
     END
IF &MODTEX EQ 141 AND &LENGTH(&D141) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D141)') SHR
     GOTO L2
     END
IF &MODTEX EQ 201 AND &LENGTH(&D201) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D201)') SHR
     GOTO L2
     END
IF &MODTEX EQ 202 AND &LENGTH(&D202) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D202)') SHR
     GOTO L2
     END
IF &MODTEX EQ 211 AND &LENGTH(&D211) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D211)') SHR
     GOTO L2
     END
IF &MODTEX EQ 221 AND &LENGTH(&D221) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D221)') SHR
     GOTO L2
     END
IF &MODTEX EQ 231 AND &LENGTH(&D231) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D231)') SHR
     GOTO L2
     END
IF &MODTEX EQ 301 AND &LENGTH(&D301) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D301)') SHR
     GOTO L2
     END
IF &MODTEX EQ 311 AND &LENGTH(&D311) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D311)') SHR
     GOTO L2
     END
IF &MODTEX EQ 411 AND &LENGTH(&D411) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D411)') SHR
     GOTO L2
     END
GOTO L3
L2:SET FT03 = &STR(Y)
L3:IF &BDISK EQ &STR( ) THEN DO
/*   LINK '&EMPREF..&EMOBJ(E&MODTEX)' &MAP SIZE(999000)-
/*     LIB('FORT.FORTLIB','&EMPREF..&EMLIB','TKTCS.LOADLIB',-
/*     'SYS1.MAINTLIB')-
/*     LOAD('&EMPREF..EMTEMP(MOD)')
/*   CALL '&EMPREF..EMTEMP(MOD)'
/*   SET LIBS = &STR(Y)
     CALL '&EMPREF..&EMLOAD(E&MODTEX)'
     SET CCODE = &LASTCC
     END
   ELSE DO
     LOAD ('&EMPREF..&EMOBJ(E&MODTEX)' &BDISK) -
       &MAP SIZE(999000)-
       LIB('FORT.FORTLIB','&EMPREF..&EMLIB','TKTCS.LOADLIB',-
       'SYS1.MAINTLIB')
     SET CCODE = &LASTCC
     SET &BDISK = &STR( )
     SET LIBS = &STR(Y)
     END
   IF &FT03 EQ &STR(Y) THEN FREE FILE(FT03F001)
   IF &CCODE NE 000 THEN GOTO SETMOD
   %NEXT
   SET CCODE = &LASTCC
   SETMOD: SET MODNUM = &CCODE - 3000
   END
END
/*
/* FREE FILES
/*
%EMEND EMPREF(&EMPREF) LIBS(&LIBS) EMLIB(&EMLIB) EMOBJ(&EMOBJ)-
   USERPROC(&USERPROC)
DEL 'AUSERPREF..EMPRO&USER'
END
```

2.3 Les paramètres de la procédure EMME2

Les paramètres substituables les plus importants pour l'usager sont les suivants:

NNN(nnn)
USERPREF(préfixe)
VOLSER(volume)
BANK(nomdebanque)
BANKDISP(disp)
BANKTOT(btot)
BANKSP(bsp)
BANKINCR(bincr)
PUNCHSP(psp)
PRINTSP(prtsp)
PLOTSP(pltsp)
BATCHLIB(fichierpartitionnébatch)
D002(d002)
D141(d141)
D201(d201)
D202(d202)
D211(d211)
D221(d221)
D231(d231)
D301(d301)
D311(d311)
D411(d411)
USERPROC('préfixe.fichierpartitionnéproc')

Seulement deux de ceux-ci n'ont pas de valeur par défaut attribuée. Le nom de la banque de données désirée (BANK) doit être mentionné obligatoirement. Le nom du fichier partionné contenant les données de base, à lire en lot, (BATCHLIB) est requis seulement si on veut lire, au cours de la session, des données en lot pour constituer une banque de données emme2 , ou encore générer un scénario à l'intérieur d'une banque existante.

2.3.1 Le paramètre NNN(nnn)

Ce paramètre permet d'accéder directement au module n.nn de emme/2 sans passer par le logo de emme/2 , ni son menu principal.

On déconseille l'utilisation de ce paramètre puisque la mise-à-jour du livre de bord ("logbook") (i.e. la mise-à-jour de la date avec les initiales de l'usager) et l'impression dans ce "logbook" de l'utilisation du module n.nn ne sont pas faites. Pour plus de détails concernant le "logbook", consulter le module 1.21 de emme/2 .

Pour la création d'une nouvelle banque de données emme/2 , il est nécessaire de faire l'appel suivant:

NNN(001)

Si NNN n'est pas appelé, alors par défaut:
nnn = 002, ce qui amène une session normale passant par l'identification du logiciel suivi du menu principal.

2.3.2 Le paramètre USERPREF(préfixe)

Ce paramètre permet d'accéder aux différents fichiers qu'utilise l'usager en rapport avec le logiciel. Il représente la première partie du "dsname" des fichiers utilisés, de sorte que emm92 reconnaîtra les fichiers:

préfixe.EMPROZZZ
préfixe.nomdebanque
préfixe.fichierpartitionnébatch
préfixe.empchZZZ
préfixe.emprtZZZ
préfixe.empltZZZ
préfixe.emproZZZ

Si USERPREF n'est pas appelé, alors par défaut on a préfixe = **A017.P0108** , qui correspond au compte attribué au Service des systèmes d'information de la D.G.T.T.P.

Pour le fichier partitionné contenant la ou les procédures d'appel à emm92 on peut avoir un "préfixe" différent, tel que donné par le paramètre USERPROC:
préfixe.fichierpartitionnéproc
(voir section 2.3.10).

2.3.3 Le paramètre VOLSER(volume)

Ce paramètre indique sur quel disque se situent les fichiers de l'usager.

Par défaut,
volume = MTQ302 (D.G.T.T.P.)

2.3.4 Les paramètres BANK(nomdebanque) et BANKDISP(disp)

Le paramètre BANK est obligatoire.
Pour une banque existante, il permet d'accéder à la banque **préfixe.nomdebanque** ;

Lors de la création d'une nouvelle banque, le fichier **préfixe.nomdebanque** est créé à l'intérieur de emm/2 et catalogué automatiquement.

BANKDISP contrôle la disposition de ce fichier par le système.

disp = OLD permet d'accéder à la banque existante **préfixe.nomdebanque**

disp = NEW entraîne la création d'une nouvelle banque **préfixe.nomdebanque**

Si BANKDISP n'est pas donné, alors par défaut: disp = OLD.

2.3.5 Les paramètres BANKTOT(btot), BANKSP(bsp) et BANKINCR(bincr)

Ces paramètres sont reliés à la dimension de la banque de données "nomdebanque".

2.3.5.1 Évaluation des dimensions d'une banque

La création d'une banque emm/2 requiert la détermination des paramètres suivants:

i) paramètres fixés par le système

MMODE: nombre maximum de modes de transport (différent du nombre maximum de modes par lien); fixé à 31.

MVEH: nombre maximum de véhicules (types de véhicules en transport en commun); fixé à 50.

MLINE: nombre maximum de lignes de transport en commun; fixé à 500

MDEML: espace, en nombre maximum de mots, attribué à chaque ensemble de démarcations; fixé à 3000.

MLOGB: espace, en nombre maximum de mots, attribué au livre de bord ("LOGBOOK"); fixé à 4000.

ii) Paramètres à fixer par l'usager pour une banque spécifique

MSCEN: nombre maximum de scénarios.
(entre 1 et 20)

MCENT: nombre maximum de centroïdes.
(entre 1 et 750)

MNODE: nombre maximum de noeuds, incluant les centroïdes.
(entre 1 et 8000)

MLINK: nombre maximum de liens.
(entre 1 et 20 000)

MTURN: nombre maximum de mots pour les virages des intersections déclarées. Chaque noeud déclaré "éclaté" occupe environ [nombre de liens entrants au noeud * nombre de liens sortants du noeud] mots.
(entre 1 et 4 000)

MLSEG: nombre maximum de segments pour l'ensemble des lignes de transport en commun.
(entre 1 et 30 000)

MMAT: nombre maximum de matrices de chaque type.
(entre 1 et 99)

MFUNC: nombre maximum de fonctions par classe.
(entre 1 et 99)

MOPER: nombre maximum d'opérations, par classe de fonctions.
(entre 1 et 2 000)

iii) autres paramètres

Ces paramètres n'ont pas d'incidence sur la taille en espace-disque de la banque.

MGROUP: nombre maximum de groupes (pour les zones); fixé à 1 000.

MLTYP: nombre maximum de types de liens; fixé à 999.

La formule suivante permet maintenant de calculer le nombre de mots requis pour une banque de données emme2 , en fonction des paramètres variables:

```
mmots = (MMAT * MCENT * MCENT) +
   max(MNODE,MLINK,MLSEG,MTURN,MCENT*MCENT)+
   (((2*MMAT)+26) * MCENT)+
   (13 * MSCEN * MLINK)+
   (9 * MSCEN * MNODE)+
   (6 * MSCEN * MLSEG)+
   (13723 * MSCEN)+
   (5 * MSCEN * MTURN)+
   (12 * MOPER)+
   (6 * MFUNC)+
   (101 * MMAT)+
   35139.
```

On peut ensuite calculer "bbloc", le nombre de blocs nécessaires pour la banque, à partir de la formule suivante(1):

```
bbloc = INT [(4*mmots/4628)+40]
```

On trouvera à la section 3.2 un exemple complet de calcul.

---

(1) [1 mot = 4 bytes] et [1 bloc = 4628 bytes]

2.3.5.2 Allocation de l'espace-disque

Une fois déterminé l'espace-disque requis pour la banque (bbloc), il reste à l'allouer au fichier "nomdebanque" à travers la procédure EMME2.

Des contraintes reliées au compilateur FORTRAN IV font qu'on ne peut utiliser ici que certaines valeurs fixées pour "btot", qui correspond au nombre de blocs (de 4628 bytes) attribués à la banque. Quatre valeurs ont été définies comme "permises" pour l'instant; elles peuvent par contre être changées par les gens responsables de l'entretien du logiciel emme2 , si nécessaire.
On retiendra donc:

btot = 380 si 0 < bbloc < = 380
= 1800 si 380 < bbloc < = 1800
= 2600 si 1800 < bbloc < = 2600
= 2700 si 2600 < bbloc < = 2700

Si bbloc > 2700, alors on doit recompiler la routine BDISK du logiciel et modifier la procédure EMME2 en conséquence, ceci étant du ressort des gens de l'entretien du logiciel.

Maintenant que btot est connu, il suffit de déterminer le nombre de blocs en espace primaire (bsp) et en espace secondaire (bincr), de telle sorte que

btot < = (bsp + 15 * bincr);

sous contrainte évidemment que cet espace soit disponible sur l'unité de disque identifiée par VOLSER.

La procédure EMME2 allouera ainsi bsp blocs de 4628 bytes en espace primaire et bincr blocs à chacune des 15 extensions secondaires.

Une fois la banque créée, il faut quand même fournir, à chaque appel par le procédure EMME2, les valeurs de btot, bsp et bincr. Par défaut, btot = 380, bsp = 300 et bincr = 40.

2.3.6 Le paramètre PUNCHSP(psp)

Ce paramètre sert à créer un fichier général d'images-cartes, utilisé au cours d'une session pour copier des données originant de la banque de donnée sur laquelle on travaille. Ce fichier pourra à son tour être plus tard lu en lot, pour générer une partie d'une autre banque, ou être utilisé par un programme quelconque de l'usager.

Au début d'une session, EMME2 attribue "psp" blocs de 9040 bytes au fichier **préfixe.EMPCHZZZ** , où ZZZ est le numéro de l'usager ayant établi cette session. Si le fichier existe déjà, découlant de la session précédente, il sera détruit et réinitialisé.

On devra donc, lorsqu'on veut reproduire en fichiers d'images-cartes le contenu d'une banque, refaire pour chaque type de données de la banque une session distincte (lignes de démarcation, matrices, lignes t.c., etc).

À la fin de chacune de ces sessions, on copiera le contenu du fichier **préfixe.EMPCHZZZ** sous forme d'un membre d'un fichier partitionné (via l'option 3.3 de SPF). Il faut en effet, pour lire des données en lot avec emme2 , qu'elles soient groupées comme membres d'un fichier partitionné (voir paramètre BATCHLIB).

Les modules de emme2 reconnaissent l'action de copier des données en images-cartes par la sélection du verbe "PUNCH".

Il est permis de spécifier une valeur nulle (psp = 0), ce qui créera un fichier bidon au lieu du fichier **préfixe.EMPCHZZZ** .

Si psp est non nul, EMME2 alloue psp blocs de 9040 bytes en espace primaire et utilise un incrément de 5 blocs pour les extensions secondaires. Il faudra donc s'assurer que le fichier **préfixe.EMPCHZZZ** comporte un espace suffisant pour recevoir le contenu à "puncher".

Par défaut, psp = 5

2.3.7 Le paramètre PRINTSP (prtsp)

Ce paramètre contrôle l'impression des rapports générés au cours d'une session emme/2 . Les modules de emme/2 reconnaissent l'action d'envoyer sur le fichier d'impression un rapport par le verbe "PRINT", le support choisi ("DEVICE") devant alors être "PRINTER".

Au début d'une session, EMME2 attribue "prtsp" blocs de 9044 bytes au fichier **préfixe.EMPRTZZZ** , où ZZZ est le numéro de l'usager ayant établi cette session. Si nécessaire, des extensions de 5 blocs seront allouées. Si le fichier existe déjà, découlant de la session précédente, il sera détruit et réinitialisé.

Il est permis de spécifier une valeur nulle (prst = 0), ce qui créera un fichier bidon au lieu de **préfixe.EMPRTZZZ** .

L'usager doit s'assurer que l'espace qu'il requiert est disponible et suffisant pour recevoir le contenu en images-lignes des rapports produits par emme/2 .

On peut envoyer le contenu du fichier **préfixe.EMPRTZZZ** sur imprimante, en utilisant l'utilitaire IEBGENER D'IBM.

Un exemple de cette procédure peut être consulté au fichier **EMME2.JCLLIB(PRINTERA)** , reproduit au tableau 2-2. Chaque usager est invité à se créer un membre de ce genre, avec ses propres paramètres d'usager, afin d'envoyer directement le fichier à l'impression (MSGCLASS = A). Il faut cependant ajuster la classe d'exécution en fonction du nombre de lignes à imprimer (20 000 lignes maximum en classe "A", sinon en classe "B"). Notons qu'on peut aussi employer l'utilitaire "3.6" de TSO/SPF pour arriver aux mêmes fins.

tableau 2-2

programme d'impression général

```
//J7PRINTA JOB (XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX),
//  'SIZZZ',CLASS=A,MSGCLASS=A,NOTIFY=NUZZZ,MSGLEVEL=(0,0)
/*ROUTE PRINT  RMT07    D= USER NAME
//*T  PRINTERA
//ETAPE02 EXEC PGM=IEBGENER,COND=EVEN
//SYSUT1 DD DSN=PREFIXE.EMPRTZZZ,DISP=(OLD,PASS)
//SYSUT2 DD SYSOUT=(A),
//       DCB=(RECFM=FBA,LRECL=133,BLKSIZE=19019,OPTCD=)
//SYSPRINT  DD DUMMY
//SYSIN DD DUMMY
//
```

2.3.8 Le paramètre PLOTSP(pltsp)

Ce paramètre sert à conserver le contenu d'un graphique emme/2 , comme une séquence de caractères de contrôle, de paramètres et d'adresses (x,y), conformément au protocole graphique de Tektronix. Les modules de emme/2 reconnaissent l'action d'envoyer sur fichier le contenu d'un graphique par la commande graphique "P".

EMME2 alloue, au début d'une session emme/2 , pltsp blocs de 8918 bytes au fichier **préfixe.EMPLTZZZ** , où **ZZZ** est le numéro de l'usager ayant établi cette session. Si nécessaire, des extensions secondaires de 5 blocs seront allouées. Si le fichier existe déjà, il sera détruit et réinitialisé par EMME2.

Il est aussi permis de spécifier une valeur nulle, afin de définir un fichier bidon.

Par défaut, pltsp = 0.

Pour envoyer sur l'écran Tektronix un fichier graphique, on utilise la procédure

PLOTPAGE
montrée au tableau 2-3.

On effectue donc, à l'écran Tektronix,

EXEC 'EMME2.EMPROC(EMSETUP)'
PLOTPAGE

pour visionner l'image-fichier **"préfixe.EMPLTZZZ"** .

tableau 2-3

procédure de rappel d'un traçage

```
PROC /*PLOTPAGE*/ 0 NAME() NU(0)-
     USERPREF('A017.P0108')-
     MAP() EMPREF(EMME2) EMLIB(EMLIB) EMOBJ(EMOBJ) EMLOAD(EMLOAD)
CONTROL MAIN   LIST    SYMLIST
/* PROCEDURE TO DISPLAY A PLOT FILE
/*      84 02 17  LGJL
/* CALLING SEQUENCE:
/*     PLOTPAGE
/*
/* OPTIONAL PARAMETERS ARE SPECIFIED AS PARAMETER(VALUE) FOLLOWING
/* PLOTPAGE (SEE EXAMPLES IN THE PROC STATEMENT)   IN THE PARAMETER
/* DESCRIPTIONS ZZZ INDICATES THE USER NUMBER TAKEN FROM THE USER
/* ID NUZZZ
/*
/* OPTIONAL PARAMETERS:
/*     NAME      NAME OF PLOT FILE (CASE: ANY PLOT FILE DIFFE-
/*               RENT OF  EMPLTZZZ)
/*               THIS PARAMETER HAS PRIORITY ON PARAMETER NU
/*     NU        USER NUMBER    3 DIGITS.
/*               ALLOW ANY ZZZ IN PLOT FILE  EMPLTZZZ
/*               BY DEFAULT IT IS THE PRESENT LOGON ID
/*               (N.B.: NAME HAS PRIORITY ON NU)
/*     USERPREF,EMPREF - SEE PROCEDURE EMME2
/*
SET USER = &SUBSTR(3:5,&SYSUID)
IF &NU NE 0 THEN SET &USER = &NU
SET PFILE = &STR(EMPLT)&USER
IF &LENGTH(&NAME) NE 0 THEN SET &PFILE = &NAME
/*   FREE FILES AND ATTRIBUTE LISTS (DCBS)
FREE ATTR(PLOTDCB TKDCB) FILE(FT08I001 FT05F001 FT06F001)
ATTRIB TKDCB RECFM(F A) LRECL(132) BLKSIZE(132)
ATTRIB PLOTDCB RECFM(V B S) LRECL(182) BLKSIZE(19019)
/*  DIALOG INPUT  */
ALLOCATE FILE(FT05F001) DA(*) USING(TKDCB)
/*  DIALOG OUTPUT  */
ALLOCATE FILE(FT06F001) DA(*) USING(TKDCB)
/*  PLOT FILE  */
ALLOC FILE(FT08F001) DATASET('&USERPREF..&PFILE')-
OLD USING(PLOTDCB)
SET LIBS = &STR(N)
/*
/*   LINK '&EMPREF..&EMOBJ(E901)' &MAP SIZE(999000)-
/*     LIB('FORT.FORTLIB','&EMPREF..&EMLIB','TKTCS.LOADLIB',-
/*     'SYS1.MAINTLIB')-
/*     LOAD('&USERPREF..EMTEMP(MOD)')
/*   SET LIBS = &STR(Y)
/*   CALL '&USERPREF..EMTEMP(MOD)'
/*
     CALL '&EMPREF..&EMLOAD(E901)'
/* FREE FILES */
FREE ATTR(TKDCB PLOTDCB) FILE(FT05F001 FT06F001 FT08F001)
IF &LIBS EQ &STR(Y) THEN DO
   FREE DA('&EMPREF..&EMOBJ' '&EMPREF..&EMLIB' 'TKTCS.LOADLIB'-
   'FORT.FORTLIB' 'SYS1.MAINTLIB')
   END
END
```

Pour visionner le fichier **préfixe.EMPLTXXX** , où XXX est un autre usager, on ajoute le paramètre NU(XXX):

PLOTPAGE NU(XXX)

Pour un fichier quelconque, on ferait

PLOTPAGE NAME("préfixe.filename") .

Un programme d'interface existe aussi pour interpréter le fichier **"préfixe.EMPLTZZZ"** et générer une sortie graphique sur table traceuse CALCOMP, à Québec.

Un exemple de ce programme peut être consulté au fichier **EMME2.JCLLIB(CALCOMP)** , reproduit au tableau 2-4. Chaque usager peut se créer sa copie du programme, en changeant les paramètres préfixe et ZZZ partout. La carte "JOB" doit aussi être adaptée à l'usager.

tableau 2-4

procédure de traçage sur Calcomp

```
//J7CALCOM  JOB  (XXXXXXXXXXXXXXXXXXXXXXXXXXX,
//  XXXXXXXXXXXXXXXXXX),'SIZZZ',CLASS=A,NOTIFY=NUZZZ,MSGCLASS=X
//*
//*  LE COMMENTAIRE SUIVANT NE S'ADRESSE PAS A L'UTILISATEUR
//*      DU FICHIER  CALCOMP   (SEULEMENT A LA PERSONNE RES-
//*      PONSABLE DE L'ENTRETIEN DU SOURCE)
//*
//*  LE PROGRAMME SOURCE DE L'EMULATEUR DU TEKTRONIX POUR CALCOMP
//*      EST SOUS LE CONTROLE DE FRANCOIS MONGRAIN
//*      A ETE REECRIT ET ADAPTE POUR LA D.G.T.T.P. DU M.T.Q.
//*          PAR ANDRE BABIN
//*      ET PROVIENT ORIGINAIREMENT DU C.R.T. DE UNIVERSITE DE MONTREAL
//*
//INIT  EXEC  PGM=STG803,
//    PARM='S1=B,S2=R,S3=V,S4=N,T=METRE'
//STEPLIB  DD  DSN=A041.MODLIB,DISP=SHR
//OS01  DD  SYSOUT=X,DCB=BLKSIZE=133
//SYSUDUMP  DD  DUMMY
//OS05  DD  SYSOUT=X,DCB=BLKSIZE=133
//TRACEUR  DD  DSN=&&A002,DISP=(MOD,PASS),
//   UNIT=PUBLIC,DCB=BLKSIZE=500,SPACE=(TRK,(20,5))
//EMUL  EXEC  PGM=ETM95600
//STEPLIB  DD  DSN=A017.MODLIB,DISP=SHR
//SYSUDUMP  DD  DUMMY
//*
//*    YPRINT ET XPRINT SONT DEUX PARAMETRES MODIFIABLES PAR L'USAGER
//*       VALEURS EXPRIMEES EN CENTIMETRES,  YPRINT REPRESENTE LA
//*       LARGUEUR DU GRAPHIQUE SUR CALCOMP (MAXIMUM PHYSIQUE ETANT
//*       DE 80) ET XPRINT REPRESENTE LA LONGUEUR DU GRAPHIQUE (SI
//*       CETTE VALEUR EST 0 ALORS XPRINT EST CALCULE PAR LE PROGRAMME
//*       DE TELLE FACON QUE LE RATIO  X SUR Y DE L'ECRAN TEKTRONIX
//*       EST PRESERVE)
//*    N.B.:  IL Y A TRANSFORMATION DU GRAPHIQUE TEKTRONIX EN GRA-
//*           PHIQUE CALCOMP EN TENANT COMPTE DU NOUVEAU RATIO
//*           FORME PAR XPRINT SUR YPRINT,  SI ON VEUT OBTENIR LE
//*           GRAPHIQUE IL FAUT S'ASSURER QUE LE RATIO XPRINT SUR
//*           YPRINT SOIT LE MEME QUE LE RATIO DU GRAPHIQUE INITIAL
//*
//FT05F001  DD  *
 &PARAM    YPRINT=80.,XPRINT=0.   &END
/*
//FT06F001  DD  SYSOUT=X
//FT08F001  DD  DSN=PREFIXE.EMPLTZZZ,DISP=SHR
//TRACEUR  DD  DSN=&&A002,DISP=(MOD,PASS),
//    UNIT=PUBLIC,DCB=BLKSIZE=500,SPACE=(TRK,(20,5))
//DESSINE  EXEC  PGM=STG804,COND=(0,NE,EMUL)
//STEPLIB  DD  DSN=A041.MODLIB,DISP=SHR
//SYSUDUMP  DD  DUMMY
//OD02  DD  DSN=A041.P0192.D02,DISP=SHR
//TRACEUR  DD  DSN=&&A002,DISP=(OLD,DELETE),
//    UNIT=PUBLIC,DCB=BLKSIZE=500
//OS06  DD  SYSOUT=X,DCB=BLKSIZE=133
```

2.3.9 Le paramètre BATCHLIB(fichierpartitionnébatch) et ses paramètres associés

Le paramètre BATCHLIB sert à transmettre au logiciel le nom du fichier partitionné (librairie) dont les membres contiennent les données à lire en lot pour générer tout ou partie d'une banque.

Ce fichier existant portera le nom:

préfixe.fichierpartitionnébatch et ses membres seront constitués d'images-cartes de 80 colonnes.

Il est obligatoire de donner une "valeur" au paramètre BATCHLIB, aucune n'étant prévue par défaut, lorsqu'on désire accéder à un des modules suivants (voir tableau ci-dessous).

Une fois identifié le fichier partitionné, il faut transmettre le nom de chacun des membres où les différents modules de EMME2 pourront lire les données en lot.

Le paramètre BATCHLIB est donc associé aux paramètres suivants dans la procédure EMME2:

Paramètre	Module correspondant	Valeur par défaut
D002(d002)	Création de banque	D002IN
D141(d141)	1.41	D141IN
D201(d201)	2.01	D201IN
D211(d211)	2.11	D211IN
D221(d221)	2.21	D221IN
D231(d231)	2.31	D231IN
D301(d301)	3.01	D301IN
D311(d311)	3.11	D311IN
D411(d411)	4.11	D411IN

Lorsque dans une session emme2 on accède à un module de lecture en lot ("BATCH ENTRY"), ou à l'option de lecture en lot à l'intérieur du module n.nn, alors il faut que le membre
préfixe.fichierpartitionnebatch(dnnn) existe et que son nom soit transmis à EMME2.

Le contenu des membres doit respecter les définitions suivantes:

Membre	Contenu
d002	la table descriptive des différents types de terminaux utilisables pour une session emme2, graphiques ou non, et des imprimantes disponibles ("DEVICE TABLE") (voir manuel pour la description). Ce membre peut être obtenu en copiant **EMME2.DATA(D002IN)** ; voir tableau 2-5.
d141	la table des lignes de démarcations.
d201	la table des modes.
d202	la table des véhicules (pour les lignes de transport en commun).
d211	la table des noeuds suivie de la table des liens (constituant le réseau de base).
d221	la table des lignes de transport en commun.
d231	la table des virages pénalisés (définissant les noeuds qui sont considérés comme intersections éclatées).

tableau 2-5

table des terminaux

```
TEKTRONIX 4116
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
27,59,27,12,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
27,58,27,12,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
27,58,27,12,32,32,32,32,32,32,32,32,32,32,32,32
07,27,26,32,32,32,32,32,32,32,32,32,32,32,32,32
27,59,27,12,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
13,10,7,62
0,5,525819,5,56,0,1000,1000,1500
IBM PC-XT (EMULATION TEKTRONIX)
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
27,59,27,12,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
27,58,27,12,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
27,58,27,12,32,32,32,32,32,32,32,32,32,32,32,32
07,27,26,32,32,32,32,32,32,32,32,32,32,32,32,32
27,59,27,12,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
13,10,7,62
0,5,525579,5,56,0,1000,1000,1500
TEKTRONIX 4116 + GRAPH. CALCOMP
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
27,59,27,12,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
27,58,27,12,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
27,58,27,12,32,32,32,32,32,32,32,32,32,32,32,32
07,27,26,32,32,32,32,32,32,32,32,32,32,32,32,32
27,59,27,12,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
13,10,7,62
0,5,527867,5,56,0,1000,1000,1500
TERMINAL NON-GRAPHIQUE
12,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
38,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
12,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
12,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
78,79,32,71,82,65,80,72,73,67,83,32,79,78,32,84
72,73,83,32,84,69,82,77,73,78,65,76,33,33,32,32
13,37,47,76
0,1,0,0,24,4800,0,0,0

IMPRIMANTE LOURDE 60 LIGNES/PAGE
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
241,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32
0,2,0,0,60,0,0,0,0
IMPRIMANTE LOURDE 45 LIGNES/PAGE
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
241,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32
0,2,0,0,45,0,0,0,0
TEKTRONIX 4663
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
27,12,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32
32,32,32,32
6,4,2811,5,0,0,0,0,0
```

d301 la table des ensembles de groupes (le système de zones est divisé en groupes).

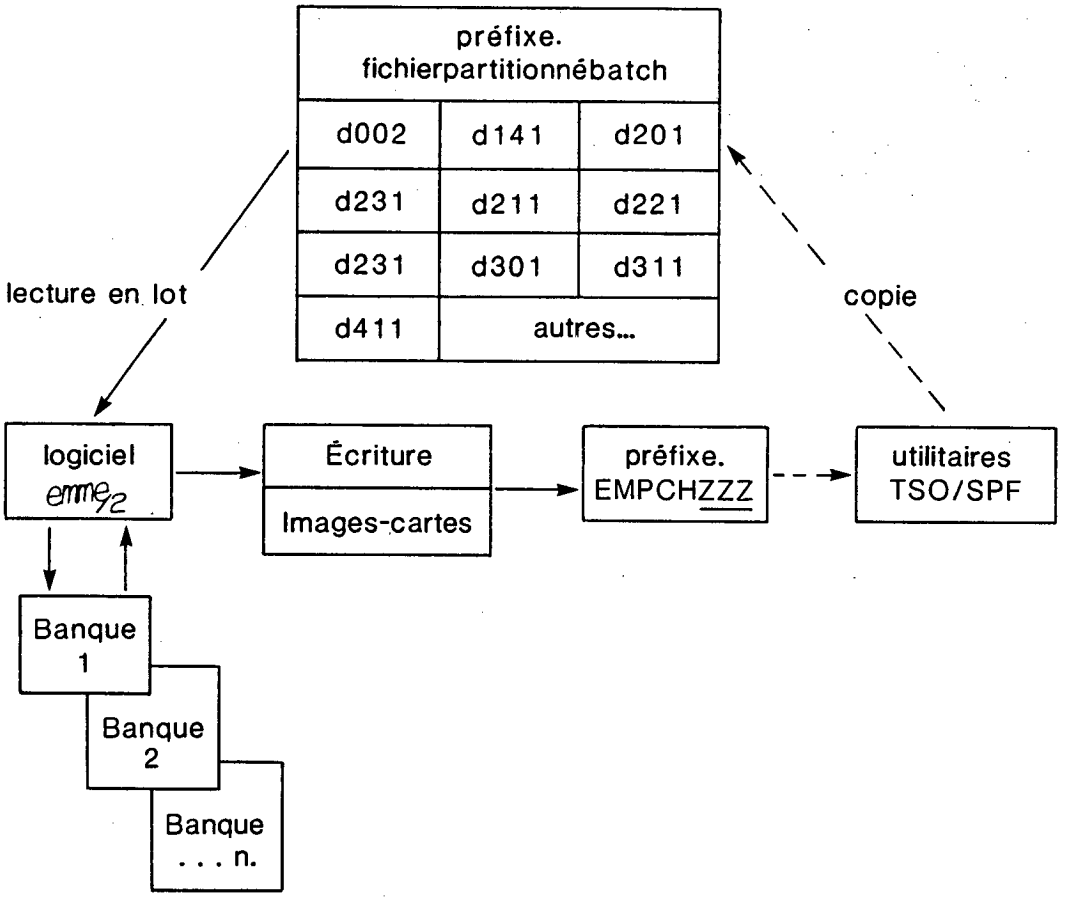
d311 la table des matrices.

d411 la table des fonctions.

Les manipulations reliées à ce fichier sont schématisées au tableau 2-6.

tableau 2-6

schéma des manipulations de données



2.3.10 Le paramètre USERPROC(fichierpartitionnéproc)

Ce paramètre sert à désigner le fichier partitionné qui contient la ou les procédures appelant la procédure EMME2 pour une ou des banques de données particulières. On notera que le nom doit apparaître au complet et que le préfixe peut être différent de celui utilisé à travers ce guide (voir section 2.3.2). Règle générale, le "préfixe" sera celui désigné par le paramètre USERPREF (section 2.3.2)

par défaut,

fichierpartitionnéproc est

A017.P0108.EMPROCX

SECTION 3

CRÉATION D'UNE BANQUE DE DONNÉES

On présentera ici le déroulement complet du processus de création d'une banque de données, incluant la création de procédures d'accès spécifiques. Un exemple complet suivra à la section 4.

3.1 **Étape 1: Autorisation d'accès**

S'assurer que le code d'usager affecté au projet permette l'accès au logiciel et qu'un espace-disque suffisant soit disponible (voir section 1).

3.2 **Étape 2: Identification du fichier de données en lot**

Si la création d'une banque de données se fait dans le cadre d'un nouveau projet, avec utilisation de données à générer, il devra y avoir eu création d'un fichier partitionné avec longueur des enregistrements à 80 colonnes. Ce fichier est différent du fichier partitionné des procédures d'accès (section 3.4).

Chacun des membres de ce fichier représentera une table de données à lire en lot pour construire la banque de données (voir le paramètre BATCHLIB, section 2.3.9).

3.3 **Étape 3: Dimensions de la banque**

On évalue les dimensions de la banque de données à créer, tel qu'expliqué à la section 2.3.5.1 et on détermine ensuite la valeur des paramètres BANKTOT, BANKSP et BANKINCR de la procédure EMME2.

3.4 **Étape 4: Création d'une procédure spécifique d'accès à la banque**

Pour l'utilisation répétée d'une banque de données particulière sous emme2 , il faudra toujours associer des valeurs invariables

à la plupart des paramètres de la procédure générale d'accès EMME2. Il peut donc devenir très intéressant de se créer des procédures "privées", spécifiques aux banques de données qui nous concernent, qui nous permettront d'y accéder plus rapidement et efficacement.

Le processus est le suivant:

I. Identification ou création d'un fichier partitionné qui contiendra les procédures d'accès à une ou plusieurs banques de données. On peut employer l'option 3.2 de SPF, pour connaître les caractéristiques du fichier **EMME2.EMPROC** ; le fichier créé est désigné comme: **fichierpartitionnéproc** .

II. Copier le membre **EMME2.EMPROC(SETUP)** de ce fichier dans votre fichier partitionné, en lui donnant le nom **EMSETUP** , à l'aide de l'option 3.3 de SPF, (voir tableau 3-1).

III. Modifier le membre **(EMSETUP)** du nouveau fichier, en insérant le nom de ce fichier partitionné au paramètre USERPROC de EMSETUP (deuxième ligne). Le nom passé devra être identique à celui passé au paramètre USERPROC d'une procédure appelant EMME2 et qui sera aussi membre (comme **EMSETUP**) de ce fichier partitionné.

tableau 3-1
procédure d'initialisation d'une session

```
PROC /*SETUP*/ 0 EMPREF(EMME2)-
    USERPROC('PREFIXE.NOMDEFICHIERPARTITIONNEPROC')
/*CONTROL LIST SYMLIST
/*
/*T   EMSETUP  84 06 08  LGJL
/*
/*    PROCEDURE TO - ATTACH EMME/2 PROCEDURE LIBRARY EMPROC FOR TSO
/*                 - SET TERMINAL DEFINITION (BACKSPACE,LINESIZE)
/*
/*    IN THE PARAMETER DESCRIPTIONS, EMME2 INDIACTES THE SYSTEM PREFIX
/*    TAKEN FORM PARAMETER EMPREF.
/*
/*    PARAMETERS:
/*        EMPREF   PREFIX USED TO FORM THE FIRST PART OF THE
/*                 DSNAME OF EMME/2 SYSTEM FILES
/*        USERPROC USER PROCEDURE LIBRARY NAME
/*
FREE FILE(SYSPROC)
ALLOCATE FILE(SYSPROC) DA('SYS1.SYSPROC'-
   'ISP.ISPPROC' 'SYS2.ISPPROC' 'SYS2.MTO.CLIST'-
   '&EMPREF..EMPROC' '&USERPROC.') SHR
%TKSTART
END
```

IV. Pour créer chacune des procédures d'accès aux banques de données, procédures qui feront appel à la procédure générale EMME2, on se basera sur une procédure existante qu'on copiera;

Copier le membre **EMME2.EMPROC(EXEM)** (voir tableau 3-2) dans votre nouveau fichier partitionné, en lui donnant le nom que vous voulez donner à votre procédure spécifique, à l'aide de l'option 3.3 de SPF; on obtient ainsi le membre **préfixe.fichierpartionnéproc(procéduredaccès)** .

V. Editer dans ce nouveau membre les changements requis au niveau des paramètres substituables (huit premières lignes).

tableau 3-2

exemple de procédure d'appel spécifique d'EMME 2

```
PROC /*EXEM*/ 0  USERPREF('PREFIXE')  VOLSER(VOLUME)-
   BANK(NOMDEBANQUE) BANKDISP(OLD) BANKTOT(BTOT)-
   BANKSP(BSP)  BANKINCR(BINCR)-
   PUNCHSP(0) PRINTSP(0)  PLOTSP(0)-
   BATCHLIB(FICHIERPARTITIONNEBATCH) D002(D002IN) D141(D141)-
   D201(D201) D202(D202) D211(D211) D221(D221) D231(D231)-
   D301(D301) D311(D311) D411(D411)-
   USERPROC('PREFIXE.FICHIERPARTITIONNEPROC')
/*
/*  CONTRAINTE:  BTOT  PLUS GRAND QUE   BSP + ( 15 * BINCR )
/*
/*  CONVENTION:  FICHIERPARTITIONNEPROC  DEVRAIT ETRE DE NOM =
/*                 EMPROCXX   OU  XX CORRESPOND A DEUX LETTRES OU
/*                            CHIFFRES ( S'ASSURER QU'IL N'EXISTE
/*                            PAS DEJA)
/*
CONTROL MAIN /*LIST SYMLIST
SET &NNN = &STR(002)
IF &BANKDISP EQ &STR(NEW) THEN SET &NNN = &STR(001)
EMME2 NNN(&NNN) USERPREF(&USERPREF) VOLSER(&VOLSER)-
     BANK(&BANK) BANKDISP(&BANKDISP) BANKTOT(&BANKTOT)-
     BANKSP(&BANKSP)  BANKINCR(&BANKINCR)-
     PUNCHSP(&PUNCHSP) PRINTSP(&PRINTSP) PLOTSP(&PLOTSP)-
     BATCHLIB(&BATCHLIB)-
     D002(&D002) D141(&D141) D201(&D201) D202(&D202) D211(&D211)-
     D221(&D221) D231(&D231) D301(&D301) D311(&D311) D411(&D411)-
     USERPROC(&USERPROC)
END
```

3.5 Étape 5: Création de la banque

Il suffit maintenant d'être en session TSO et d'exécuter pour débuter cette session emme/2 :

- **EXEC 'préfixe.fichierpartitionnéproc(EMSETUP)'**
- **procéduredaccès BANKDISP(NEW)**

Après quoi débutera la session emme/2 ; on aura alors à répondre au dialogue pour donner les différentes dimensions ainsi qu'initialiser le premier scénario.

On peut alors poursuivre la session en débutant la lecture des données.

3.6 Étape 6: Lecture des données dans la banque

Si on n'est pas actuellement en session emme/2 , il faut y entrer:

- **EXEC 'préfixe.fichierpartitionnéproc(EMSETUP)'**
- **procéduredaccès**

On lit maintenant les données du fichier partitionné de l'étape 2 pour remplir la banque nouvellement créée, ou encore on les génère interactivement durant la session emme/2 .

Il y a deux types de données: celles reliées à un scénario et qui sont hiérarchisées, et les autres données non reliées à un scénario et qui servent à l'ensemble des scénarios de la banque.

3.6.1 Données d'un scénario

Il s'agit de s'assurer que le scénario courant est bien celui sur lequel on veut entrer les données; pour ce faire, au niveau du menu principal des modules, on entre la commande:

S = n

où n est le numéro du scénario désiré.
Exemple: MODULE : S = 1000

Voici l'ordre dans lequel on doit lire les données pour remplir un scénario, en passant par les modules de lecture en lot correspondants:

1- module 2.01 (modes)
2- module 2.02 (véhicules)
3- module 2.11 (réseau de base)
4a- module 2.31 (virages pénalisés)
4b- module 2.21 (lignes T.C.)

Les deux derniers modules peuvent indifféremment être appelés l'un avant l'autre, car il n'y a pas de relation hiérarchique entre les deux types de données.

Si on veut relire dans un scénario des données en lot (à cause d'une mise-à-jour par exemple), alors il suffit dans le scénario en question de détruire les données qui sont de niveau hiérarchique plus bas, puis de relire les données en question ainsi que celles qui ont été détruites.

Exemple: relire le réseau de base

1° mettre le scénario en question comme scénario courant (si ce n'est pas déjà fait);

2° détruire les lignes T.C.
(module 2.22 en initialisant la table transit);

3° détruire les virages
(module 2.31 initialiser les virages)

4° relire le réseau de base
(module 2.11)

5° relire les virages
(module 2.31)

6° relire les lignes T.C.
(module 2.21)

3.6.2 Données générales de la banque

Il n'y a pas d'ordre pour la lecture des données suivantes, à l'exception des matrices et des groupes pour lesquels il doit exister un scénario avec la table des noeuds lue (le réseau de base comprend la table des noeuds suivie de la table des liens) dans la banque. Les données qui suivent peuvent être lues à partir de n'importe lequel scénario, puisqu'elles n'affectent pas les scénarios.

module 1.41 (démarcations)
module 3.01 (groupes)
module 3.11 (matrices)
module 4.11 (fonctions)

Il n'est pas nécessaire d'avoir, dans la banque de données, toutes les données énumérées précédemment pour pouvoir utiliser le logiciel emme/2 sur la banque en question. Par contre chaque module requiert certains types de données lorsqu'on y accède.

Il doit y avoir consistence entre la liste des centroïdes du scénario courant et les listes des centroïdes sous lesquels on a traité les groupes ou matrices, lorsqu'on veut traiter ces mêmes groupes ou matrices.

SECTION 4

SECTION 4

EXEMPLE DE CRÉATION D'UNE BANQUE DE DONNÉES

4.1 Autorisation d'accès

L'usager vérifie qu'il est autorisé à effectuer les manipulations qui suivront et s'assure qu'il consommera de façon "justifiable" les ressources en espace-disque qu'il gèlera.

4.2 Fichier de données en lot

Nous utiliserons simplement pour cette démonstration, les données pour la région de Montréal, déjà contenues dans le fichier partitionné **A017.P0108.EMMTL** ; ce fichier contient des membres dont les noms sont mnémotechniques et qui deviendront des paramètres substitués pour l'appel de EMME2.

Le tableau 4-1 montre l'index de ce fichier de données, avec ses principales caractéristiques.

tableau 4-1

index d'un fichier de données typique

PROJECT: A017
LIBRARY: P0108
TYPE: EMMTL

DATE: 84/10/19
TIME: 11:33
PAGE: 001

GENERAL DATA:
VOLUME SERIAL: MTQ302
DEVICE TYPE: 3380
ORGANIZATION: PO
CREATION DATE: 84/10/16

GENERAL DATA:
RECORD FORMAT: FB
RECORD LENGTH: 80
BLOCK SIZE: 9,040
1ST EXTENT SIZE: 9
SECONDARY QUAN: 2

CURRENT ALLOCATION:
31 CYLINDERS
12 EXTENTS
20 DIRECTORY BLOCKS

CURRENT UTILIZATION:
31 CYLINDERS
12 EXTENTS
5 DIRECTORY BLOCKS
27 MEMBERS

MEMBER NAME	VERS.MOD LEVEL	CREATION DATE	DATE AND TIME LAST MODIFIED	CURRENT NO. LINES	INITIAL NO. LINES	MODIFIED NO. LINES	USER ID
AUTO	01.00	84/10/19	84/10/19 11:23	5557	5557	0	NU188
DEMVI	01.00	84/10/19	84/10/19 11:23	1924	1924	0	NU188
DEMZ7	01.00	84/10/19	84/10/19 11:24	3988	3988	0	NU188
D002IN	01.08	83/11/11	84/09/19 19:29	91	104	0	NU157
FCT	01.02	84/02/21	84/08/22 09:33	22	18	0	NU104
FCTSIMUL	01.01	84/08/15	84/08/15 11:33	27	27	0	NU752
FCT40M	01.01	84/08/17	84/08/20 10:20	27	27	0	NU141
FCT40P	01.05	84/02/21	84/08/22 08:32	29	18	0	NU141
GRP	01.00	84/10/19	84/10/19 11:24	136	136	0	NU188
MATFM	01.00	84/08/22	84/08/22 12:15	5557	5557	0	NU104
MOD	01.01	83/09/22	83/09/26 10:23	16	16	0	NU552
NET	01.02	84/09/13	84/09/21 20:31	26662	26663	0	NU141
NETFUTUR	01.00	84/10/19	84/10/19 11:26	26744	26744	0	NU188
NETOBSRR	01.00	84/10/19	84/10/19 11:27	26654	26654	0	NU188
NETTC	01.00	84/10/01	84/10/01 14:16	26664	26664	0	NU188
NETTRT	01.00	84/04/13	84/04/13 10:33	1	1	0	NU552
NET9999	01.00	84/10/19	84/10/19 11:29	26658	26658	0	NU188
PEN	01.00	84/10/19	84/10/19 11:29	204	204	0	NU188
TEST	01.01	84/01/18	84/01/18 12:54	21	21	0	NU552
TRANS	01.00	84/10/19	84/10/19 11:30	3605	3605	0	NU188
TRANS1	01.00	84/04/06	84/04/06 11:06	265	265	0	NU552
TRL	01.81	84/01/04	84/10/02 10:47	1030	676	0	NU141
TRLF2	01.01	84/07/27	84/07/27 14:03	1029	1029	0	NU104
TRLX	01.00	84/09/21	84/09/21 19:39	932	932	0	NU141
TRL9999	01.00	84/10/19	84/10/19 11:30	1030	1030	0	NU188
VEH	01.00	84/10/19	84/10/19 11:30	13	13	0	NU188
VEH40	01.00	84/10/19	84/10/19 11:31	13	13	0	NU188
MAXIMUMS:	01.81	84/10/19	84/10/19 11:31	26,744	26,744	0	
TOTALS:				158,899	158,544	0	

END OF MEMBER LIST

4.3 Dimensions de la banque

Pour les données sur la région de Montréal, on a considéré la construction d'une banque de données emme2 (le nom de la banque est MTLBANK1), possédant les dimensions suivantes:

paramètre variable	valeur
MSCEN	2
MCENT	700
MNODE	7000
MLINK	20000
MTURN	1000
MLSEG	21000
MMAT	3
MFUNC	30
MOPER	600

par le formule, voir section 2.3.5.1.
mmots = 2960266

donc
bbloc = entier de [(2558.57) + 40]

bbloc = 2598

Dans les circonstances, on retiendra une valeur arrondie de 2600 pour le paramètre BANKTOT de la procédure EMME2.

Supposons qu'on constate qu'il reste un maximum de 155 pistes contiguës sur le disque, en faisant appel à l'utilitaire 3,7 de SPF; on peut donc associer 1200 blocs au paramètre BANKSP, puisque 8 blocs correspondent à une piste. En posant BANKINCR = 96, on s'assure d'un espace total suffisant
([1200 + 15 * 96] > 2600).

4.4 Procédure spécifique d'accès

I) on crée le fichier partitionné **A017.P0108.EMPROCUS** , où **"US"** pourraient être les initiales de l'usager.

II) on copie **EMME2.EMPROC(SETUP)**

dans

A017.P0108.EMPROCUS(EMSETUP)

III) on modifie une ligne du membre **A017.P0108.EMPROCUS(EMSETUP)** pour obtenir sur cette ligne, le membre: **USERPROC('A017.P0108.EMPROCUS')** , tel que montré au tableau 4-2.

tableau 4-2

procédure "EMSETUP"

```
PROC /*EMSETUP*/ 0 EMPREF(EMME2) USERPROC('A017.P0108.EMPROCUS')
/*CONTROL LIST SYMLIST
/*
/*T   EMSETUP  84 06 08  LGJL
/*
/*    PROCEDURE TO - ATTACH EMME/2 PROCEDURE LIBRARY EMPROC FOR TSO
/*                 - SET TERMINAL DEFINITION (BACKSPACE,LINESIZE)
/*
/*    IN THE PARAMETER DESCRIPTIONS, EMME2 INDIACTES THE SYSTEM PREFIX
/*    TAKEN FORM PARAMETER EMPREF.
/*
/*    PARAMETERS:
/*        EMPREF   PREFIX USED TO FORM THE FIRST PART OF THE
/*                 DSNAME OF EMME/2 SYSTEM FILES
/*        USERPROC USER PROCEDURE LIBRARY NAME
/*
FREE FILE(SYSPROC)
ALLOCATE FILE(SYSPROC) DA('SYS1.SYSPROC'-
   'ISP.ISPPROC' 'SYS2.ISPPROC' 'SYS2.MTQ.CLIST'-
   '&EMPREF..EMPROC' '&USERPROC.') SHR
%TKSTART
END
```

IV) on copie

EMME2.EMPROC(EXEM)

dans

A017.P0108.EMPROCUS(MTLTC)

Ceci aura créé la procédure "MTLTC" qui servira à appeler la banque de données "transport collectif" pour la région de Montréal, (voir tableau 4-3).

V) on édite les changements pour les paramètres à substituer, en leur donnant la valeur par défaut appropriée. On notera ici que le paramètre NNN est devenu transparent à l'usager.

tableau 4-3

procédure "MTLTC"

```
PROC /*MTLTC*/ 0  USERPREF('A017.P0108')  VOLSER(MTQ302)-
   BANK(BANKMTL1) BANKDISP(OLD) BANKTOT(2600)-
   BANKSP(1200)  BANKINCR(96)-
   PUNCHSP(0) PRINTSP(0)  PLOTSP(0)-
   BATCHLIB(EMMTL) D002(D002IN) D141(DEMZ7)-
   D201(MOD) D202(VEH) D211(NETTC) D221(TRL) D231(PEN)-
   D301(GRP) D311(TRANS) D411(FCT)-
   USERPROC('A017.P0108.EMPROCUS')
/*
/*  CONTRAINTE:  BTOT  PLUS GRAND QUE   BSP + ( 15 * BINCR )
/*
/*  CONVENTION:  FICHIERPARTITIONNEPROC  DEVRAIT ETRE DE NOM =
/*                 EMPROCXX   OU  XX CORRESPOND A DEUX LETTRES OU
/*                            CHIFFRES ( S'ASSURER QU'IL N'EXISTE
/*                            PAS DEJA)
/*
CONTROL MAIN /*LIST SYMLIST
SET &NNN = &STR(002)
IF &BANKDISP EQ &STR(NEW) THEN SET &NNN = &STR(001)
EMME2 NNN(&NNN) USERPREF(&USERPREF) VOLSER(&VOLSER)-
     BANK(&BANK) BANKDISP(&BANKDISP) BANKTOT(&BANKTOT)-
     BANKSP(&BANKSP)  BANKINCR(&BANKINCR)-
     PUNCHSP(&PUNCHSP) PRINTSP(&PRINTSP) PLOTSP(&PLOTSP)-
     BATCHLIB(&BATCHLIB)-
     D002(&D002) D141(&D141) D201(&D201) D202(&D202) D211(&D211)-
     D221(&D221) D231(&D231) D301(&D301) D311(&D311) D411(&D411)-
     USERPROC(&USERPROC)
END
```

NOTE : Cet exemple ne doit en aucun cas être exécuté tel quel par un nouvel usager. Il est donné ici à titre indicatif, puisque la banque BANKMTL1 existe déjà.

4.5 Création de la banque

EXEC 'A017.P0108.EMPROCUS(EMSETUP)'
MTLTC BANKDISP(NEW)

4.6 Lecture des données dans la banque ou utilisation générale

Si on n'y est pas déjà, on entre dans la banque en tapant simplement:
EXEC 'A017.P0108.EMPROCUS(EMSETUP)'
MTLTC

Le reste est strictement de la "cuisine" emm92 , et l'usager peut alors se référer au manuel de l'usager pour toute question d'application.

APPENDICE "A"

APPENDICE "A"

DESCRIPTION DES FICHIERS

A.1 Fichiers généraux

Plusieurs fichiers ci-dessous auront dans leur contenu des noms symboliques. L'usager habituellement recopiera ces fichiers dans sa librairie et en conséquent changera ces noms symboliques pour les noms actuels reliés à son projet spécifique (exemple: nom de la banque).

EMME2.EMPROC(EMME2)

Ce fichier est la procédure standard d'appel au logiciel emme/2 . En effet toute procédure de l'usager pour un projet donné accédant à emme/2 , fait appel à la procédure EMME2.

EMME2.EMPROC(EMSETUP)

Ce fichier est la procédure à exécuter en T.S.O. pour pouvoir accéder directement aux autres procédures du fichier partitionné EMPROC.

EMME2.EMPROC(EXEM)

Ce fichier contient la procédure d'exemple avec noms symboliques permettant d'appeler la procédure EMME2 pour un projet donné.

EMME2.EMPROC(PLOTPAGE)

Ce fichier est la procédure permettant de générer des graphiques emme/2 sur un écran de terminal Tektronix à partir d'un fichier graphique emme/2 (plot file).

EMME2.EMPROC(SETUP)

Ce fichier contient la procédure avec noms symboliques permettant, lorsqu'éxécutée en T.S.O., de pouvoir accéder directement aux autres procédures du fichier partitionné de l'usager. Normalement ce fichier est recopié dans le fichier partitionné de l'usager avec comme nom EMSETUP.

EMME2.JCLLIB(CALCOMP)

Ce fichier contient la procédure JCL avec noms symboliques, permettant d'envoyer sur le traceur CALCOMP, les graphiques provenant d'un fichier graphique emme/2 (plot file).

EMME2.JCLLIB(PRINTERA)

Ce fichier est la procédure JCL permettant d'envoyer à l'imprimante le fichier d'impression emme/2 **(préfixe.EMPRTZZZ)** dont le nombre de lignes est plus petit que 20 000 (CLASS=A).

EMME2.JCLLIB(PRINTERB)

Ce fichier est la procédure JCL identique à PRINTERA, à utiliser lorsque le nombre de lignes du fichier d'impression emme/2 est plus grand que 20 000 (maximum 50 000) (CLASS=B).

EMME2.JCLLIB(RUN521)

Ce fichier est la procédure JCL permettant d'exécuter le module 5.21 de emme/2 en lot pour une banque donnée emme/2 (simulation sur le réseau routier).

EMME2.JCLLIB(RUN531)

Ce fichier est la procédure JCL permettant d'exécuter le module 5.31 de emme/2 en lot pour une banque donnée emme/2 (simulation sur le réseau de transport en commun).

EMME2.DATA(DOO2IN)

Ce fichier contient la table des données relatives à la description des différentes composantes sur les terminaux utilisés à la D.G.T.T.P. (service des Systèmes d'information).

A.2 Fichiers particuliers à la D.G.T.T.P., à Montréal

A017.P0108.EMPROCUS

Le fichier partitionné qui contient les procédures d'appel à emme2 pour les projets sur le territoire de la région métropolitaine.

EMSETUP — Membre provenant du fichier **EMME2.EMPROC(SETUP)** .

MTLTC — Membre qui est la procédure d'accès à emme2 pour la banque de Montréal relative au transport en commun.

MTLRR — Membre qui est procédure d'accès à emme2 pour la banque de Montréal relative au réseau routier.

LAVAL8 — Membre qui est la procédure d'accès à emme2 pour la banque de Laval relative à 8 secteurs.

LAVAL35 — Membre qui est la procédure d'accès à emme2 pour la banque de Laval relative à 35 secteurs.

LAVAL105 — Membre qui est la procédure d'accès à emme2 pour la banque de Laval relative à 105 secteurs.

RIVESD — Membre qui est la procédure d'accès à emme2 pour la banque de la Rive-Sud.

A017.P0108.EMJCLMTL

PRINTERA
PRINTERB — membres qui ont été recopiés du fichier partitionné EMME2.JCLLIB et qui ont été modifiés en fonction des besoins de la D.G.T.T.P.

RUN521 membre recopié de **EMME2.JCLLIB** et modifié pour la banque de Montréal sur le réseau routier: **A017.P0108.BANKMTL1** .

RUN531 membre recopié de **EMME2.JCLLIB** et modifié pour la banque de Montréal sur le réseau de transport en commun:
A017.P0108.BANKMTL2

FTTRL membre qui permet de transformer le fichier de données sur les lignes T.C.
A017.P0108.EMMTL(TRL)
où la fréquence de chaque ligne est augmentée de 2; résultat sur
A017.P0108.EMMTL(TRLF2)

TCVCNET membre qui permet de transformer le fichier du réseau de base
A017.P0108.EMMTL(NET)
où certaines modifications, comme éliminer le mode auto, sont faites pour obtenir un réseau orienté sur le transport en commun
A017.P0108.EMMTL(NETTC)

LECTTRAN membre qui permet de créer la matrice des déplacements en transport en commun emme/2 à partir de la matrice UTPS correspondante.

LECTAUTO membre qui permet de créer la matrice des véhicules pour le réseau routier emme/2 à partir de la matrice UTPS correspondante.

LECTNET membre qui permet de lire le réseau routier de la banque emme/2 pour la région de Montréal pour fin de comparaison avec les comptages (versus volumes simulés).

FTEXSP membre qui permet d'extraire les liens du réseau routier dont les vitesses simulées sont plus petites qu'une valeur donnée.

PUNCHISO membre qui permet de lire une matrice de la banque emme2 réseau routier et de la transformer en matrice UTPS pour fin de création du graphique des isochrones (sur les temps simulés).

CALCOMP membre recopié de **EMME2.JCLLIB** et modifié en conséquence pour le projet de la région de Montréal.

RUNCHAIN membre qui permet de faire deux simulations sur une même banque emme2 (transport en commun) grâce au chaînage des jobs.

A017.P0108.EMMTL

contient l'ensemble des tables de données emme2 nécessaires pour les banques emme2 réseau routier et transport en commun de la région de Montréal.

A017.P0108.EMLVL

contient l'ensemble des tables de données emme2 nécessaires pour les banques emme2 du projet sur la région de Laval.

A017.P0108.EMRVS

contient l'ensemble des tables de données emme2 nécessaires pour les banques emme2 du projet de la Rive-Sud.

APPENDICE "B"

APPENDICE "B"

DESCRIPTION DES BANQUES DE LA D.G.T.T.P., MONTRÉAL

Afin de mieux orienter l'usager, cette section présentera les quatres principales banques de données emme2 déjà créées et utilisées pour les études courantes à la D.G.T.T.P.

1: Réseau de transport collectif de la région de Montréal;

2: Réseau routier de la région de Montréal;

3: Plan de transport de la Rive-Sud;

4: Plan de transport CTL-Laval.

B.1 Réseau de transport collectif de la région de Montréal

B.1.1 La procédure d'accès:

Cette banque a été construite afin de donner à chacune des lignes de T.C. sa propre vitesse commerciale, plutôt qu'une référence à des vitesses "automobiles" sur le réseau routier.

Banque: BANKMTL1

Procédure d'appel: MTLTC (décrite au tableau 4-3)

Fichier: **A017.P0108.EMPROCUS**

Données en lot: **A017.P0108.EMMTL**

Dimensions: 699 zones
2 scénarios
3 matrices
les autres dimensions sont telles que montrées en exemple à la section 4 du guide.

La procédure MTLTC contient par défaut l'appel aux membres suivants:

MOD	modes
VEH	véhicules
NETTC	réseau de base
TRLF2	lignes T.C.
DEMZ7	démarcations
FCT	fonctions
TRANS	matrices des déplacements T.C.

où

TRANS vient de l'exécution du membre **A017.P0108.CDCLASSE(MBUILD02)** suivi de l'exécution du membre **A017.P0108.EMJCLMTL(LECTTRAN)** dont la sortie est le fichier TRANS

NETTC vient de l'exécution du membre **A017.P0108.EMJCLMTL(TCVCNET)** qui lit le fichier NET et le transforme en fichier NETTC

où

- le mode "a" est enlevé sur la plupart des liens;
- la donnée d'usager #3 sur les noeuds correspond au temps de transfert;
- le mode "p" est ajouté à certains liens.

TRLF2 vient du fichier TRL, où les fréquences des lignes de transport collectif sont ajustées, de telle sorte que le temps maximum d'attente pour une ligne soit de 10 minutes. Obtenu par l'exécution de **A017.P0108.EMJCLMTL(FTTRL)** .

Éventuellement, ces changements se feront directement par emme72 , de façon transparente à l'usager.

B.1.2 Préparation de la simulation

Étape 1: s'assurer que l'accès aux fonctions "transit" est permis pour les lignes T.C. circulant sur des liens ayant le mode auto (mode "a");

La démarche suivante doit être entreprise toutes les fois que le réseau de base ou bien les lignes T.C. sont relus dans le scénario où on veut simuler le transport en commun, avec les vitesses commerciales.

Mettre les temps auto en valeurs positives, sur le scénario en question, à l'aide de

module 1.11
select: 2 = fill data
enter: file, record =
17,1 (pour scénario 1000)
17,2 (pour scénario 2000)
enter: from word, to word, new value
= 1,20000,1.0
= /
quitter le module

Note: on change IF,IR,1W1,1W2 =
17,ISCEN,1,200000
Voir manuel emme/2 .

Note: Cette étape est temporaire, en attendant une nouvelle version de l'affectation transport en commun.

Étape 2: S'assurer que

- les fréquences des lignes T.C. sont celles ajustées pour un temps maximum d'attente par ligne de 10 minutes (fichier TRLF2);

- la troisième donnée d'usager des noeuds contient le temps de transfert.

- normalement ces données sont déjà incluses dans la lecture du réseau de base.

Étape 3: Préparation de l'affectation, à l'aide du module 5.11:

- matrice de demande: MF1
- matrices de temps obtenues: MF2 (scénario 1000) et MF3 (scénario 2000)
- facteur global de temps d'attente: 0,5
- poids unitaires pour les composantes de temps

Étape 4: À l'extérieur du logiciel,
lancer l'exécution du membre
A017.P0108.EMJCLMTL(RUN531)
en s'assurant que ce membre ait été modifié en conséquence, ait suffisamment de temps d'alloué et que la bonne banque est appelée BANKMTL1

Note: On peut simuler interactivement, à l'intérieur du logiciel emme2 , une paire O/D en créant une matrice de zéros sauf pour la paire en question et en appelant directement 5.31 à l'intérieur de emme2 .

B.2 Réseau routier de la région de Montréal

Banque: BANKMTL2

Procédure: MTLRR

Fichier: **A017.P0108.EMPROCUS**

Données: **A017.P0108.EMMTL**

Dimensions: 699 zones
2 scénarios
3 matrices
4000 mots pour les virages déclarés;
les autres dimensions sont telles que montrées à la section 3.7.3.

tableau B-1

procédure "MTLRR"

```
PROC /*MTLRR*/ 0  USERPREF('A017.P0108')  VOLSER(MTQ302)-
   BANK(BANKMTL2) BANKDISP(OLD) BANKTOT(2700)-
   BANKSP(2400)  BANKINCR(20)-
   PUNCHSP(0) PRINTSP(0)  PLOTSP(0)-
   BATCHLIB(EMMTL) D002(D002IN) D141(DEMZ7)-
   D201(MOD) D202(VEH40) D211(NET) D221(TRL) D231(PEN)-
   D301(GRP) D311(AUTO) D411(FCT40P)-
   USERPROC('A017.P0108.EMPROCUS')
/*
/*  CONTRAINTE:  BTOT  PLUS GRAND QUE   BSP + ( 15 * BINCR )
/*
/*  CONVENTION:  FICHIERPARTITIONNEPROC  DEVRAIT ETRE DE NOM =
/*                 EMPROC%%   OU  %% CORRESPOND A DEUX LETTRES OU
/*                            CHIFFRES ( S'ASSURER QU'IL N'EXISTE
/*                            PAS DEJA)
/*
CONTROL MAIN /*LIST SYMLIST
SET &NNN = &STR(002)
IF &BANKDISP EQ &STR(NEW) THEN SET &NNN = &STR(001)
EMME2 NNN(&NNN) USERPREF(&USERPREF) VOLSER(&VOLSER)-
     BANK(&BANK) BANKDISP(&BANKDISP) BANKTOT(&BANKTOT)-
     BANKSP(&BANKSP)  BANKINCR(&BANKINCR)-
     PUNCHSP(&PUNCHSP) PRINTSP(&PRINTSP) PLOTSP(&PLOTSP)-
     BATCHLIB(&BATCHLIB)-
     D002(&D002) D141(&D141) D201(&D201) D202(&D202) D211(&D211)-
     D221(&D221) D231(&D231) D301(&D301) D311(&D311) D411(&D411)-
     USERPROC(&USERPROC)
END
```

La procédure MTLRR contient par défaut l'appel aux membres suivants:

MOD	modes
VEH40	véhicules
NET	réseau de base
PEN	virages
TRL	lignes T.C.
DEMZ7	démarcations (700 zones)
FCT40P	fonctions
AUTO	matrice des déplacements auto

Le contenu de ces membres se caractérise ainsi:

VEH40 vient du fichier VEH
où l'équivalent en auto est multiplié par 2.5
pour donner VEH40

FCT40P vient de FCT40
dont on a réduit les puissances des fonctions V/D pour donner FCT40P

FCT40 vient de FCT
dont on a transformé VOLAU par VOLAU * 40%
pour donner FCT40P

AUTO vient de l'exécution du membre
A017.P108.CDCLASSE(MBUILD01)
suivi de l'exécution du membre
A017.P0108.EMJCLMTL(LECTAUTO)
dont la sortie est le fichier AUTO

NET viennent de l'exécution du membre
PEN **A017.P0108.RESEAU(FTRESO)**

DEMZ7 vient de l'exécution du membre
A017.P0108.RESEAU(FTEMMEZ7)

B.3 Plan de transport de la Rive-Sud

Afin de traiter l'analyse des lignes de désir dans le cadre de cette étude, on a généré une banque de données spéciale, dédiée uniquement à l'examen des matrices de demande. Les seuls noeuds définis ici sont les centroïdes(1), tandis qu'un ensemble complet de lignes de démarcation permet de tracer toutes les zones d'analyse.

Banque: BANKRVS

Procédure: RIVESD

Fichier: **A017.P0108.EMPROCUS**

Données: **A017.P0108.EMRVS**

Dimensions: 36 zones
1 scénario
50 matrices

tableau B-2
procédure "RIVESD"

```
PROC /*RIVESD*/ 0  USERPREF('A017,P0108')  VOLSER(MTQ302)-
   BANK(BANKRVS) BANKDISP(OLD) BANKTOT(380)-
   BANKSP(230)  BANKINCR(10)-
   PUNCHSP(0) PRINTSP(0)  PLOTSP(0)-
   BATCHLIB(EMRVS) D002(D002IN) D141(DEM)-
   D201(MOD) D202(VEH) D211(NET) D221(TRL) D231(PEN)-
   D301(GRP) D311(MAT) D411(FCT)-
   USERPROC('A017.P0108.EMPROCUS')
/*
/*  CONTRAINTE:  BTOT  PLUS GRAND QUE   BSP + ( 15 * BINCR )
/*
/*  CONVENTION:  FICHIERPARTITIONNEPROC  DEVRAIT ETRE DE NOM =
/*                 EMPROCXX   OU  XX CORRESPOND A DEUX LETTRES OU
/*                            CHIFFRES ( S'ASSURER QU'IL N'EXISTE
/*                            PAS DEJA)
/*
CONTROL MAIN /*LIST SYMLIST
SET &NNN = &STR(002)
IF &BANKDISP EQ &STR(NEW) THEN SET &NNN = &STR(001)
EMME2 NNN(&NNN) USERPREF(&USERPREF) VOLSER(&VOLSER)-
     BANK(&BANK) BANKDISP(&BANKDISP) BANKTOT(&BANKTOT)-
     BANKSP(&BANKSP)  BANKINCR(&BANKINCR)-
     PUNCHSP(&PUNCHSP) PRINTSP(&PRINTSP) PLOTSP(&PLOTSP)-
     BATCHLIB(&BATCHLIB)-
     D002(&D002) D141(&D141) D201(&D201) D202(&D202) D211(&D211)-
     D221(&D221) D231(&D231) D301(&D301) D311(&D311) D411(&D411)-
     USERPROC(&USERPROC)
END
```

(1) En réalité quelques liens "bidons" ont été définis, aux fins d'utilisation des commandes graphiques qui sont axées sur la manipulation d'un réseau.

B.4 Plan de transport CTL-Laval

Cette étude a nécessité la construction de trois banques, correspondant à des systèmes de 8, 35 et 105 zones d'analyse. Ces banques sont dédiées uniquement à l'examen des matrices de demande, à l'aide de lignes de désir tracées sur un fond représentant les découpages des zones d'analyse.

Banque: BANKLVL1

Procédure: LAVAL35

Fichier: **A017.P0108.EMPROCUS**

Données: **A017.P0108.EMLVL**

Dimensions: 35 zones
1 scénario
99 matrices

tableau B-3

procédure "LAVAL35"

```
PROC 0 LDI(*) LDO(*) LERO(*) MAP()-
   BANK(BANKLVL1) BANKDISP(OLD) BANKTOT(380) BANKSP(230)-
   BANKINCR(10)-
   PUNCHSP(0) PRINTSP(10)  PLOTSP(0)-
   BATCHLIB(EMLVL) D002(D002IN) D141(DEM35)-
   D201(MOD) D202(VEH) D211(NET35) D221(TRL) D231(PEN)-
   D301(GRP) D311(MAT35) D411(FCT)
CONTROL MAIN /*LIST SYMLIST
SET &NNN = &STR(002)
IF &BANKDISP EQ &STR(NEW) THEN SET &NNN = &STR(001)
SET MAPSTR = &STR()
SET BLIBSTR = &STR()
IF &LENGTH(&MAP) NE 0 THEN SET &MAPSTR =MAP(&MAP)
IF &LENGTH(&BATCHLIB) NE 0 THEN SET &BLIBSTR =BATCHLIB(&BATCHLIB)
EMME2 NNN(&NNN) &MAPSTR &BLIBSTR-
     BANK(&BANK) BANKDISP(&BANKDISP) BANKTOT(&BANKTOT) BANKSP(&BANKSP)-
     BANKINCR(&BANKINCR)-
     PRINTSP(&PRINTSP) PUNCHSP(&PUNCHSP) PLOTSP(&PLOTSP)-
     D002(&D002) D141(&D141) D201(&D201) D202(&D202) D211(&D211)-
     D221(&D221) D231(&D231) D301(&D301) D311(&D311) D411(&D411)
END
```

Banque: BANKLVL2

Procédure: LAVAL8

Fichier: **A017.P0108.EMPROCUS**

Données: **A017.P0108.EMLVL**

Dimensions: 8 zones

1 scénario

99 matrices

tableau B-4

procédure "LAVAL8"

```
PROC /*LAVAL8*/ 0  USERPREF('A017.P0108')  VOLSER(MTQ302)-
   BANK(BANKLVL2) BANKDISP(OLD) BANKTOT(380)-
   BANKSP(230)  BANKINCR(10)-
   PUNCHSP(0) PRINTSP(0)  PLOTSP(0)-
   BATCHLIB(EMLVL) D002(D002IN) D141(DEM8)-
   D201(MOD) D202(VEH) D211(NET8) D221(TRL) D231(PEN)-
   D301(GRP) D311(MAT8) D411(FCT)-
   USERPROC('A017.P0108.EMPROCUS')
/*
/*  CONTRAINTE:  BTOT  PLUS GRAND QUE   BSP + ( 15 * BINCR )
/*
/*  CONVENTION:  FICHIERPARTITIONNEPROC  DEVRAIT ETRE DE NOM =
/*                 EMPROC%%   OU  %% CORRESPOND A DEUX LETTRES OU
/*                            CHIFFRES ( S'ASSURER QU'IL N'EXISTE
/*                            PAS DEJA)
/*
CONTROL MAIN /*LIST SYMLIST
SET &NNN = &STR(002)
IF &BANKDISP EQ &STR(NEW) THEN SET &NNN = &STR(001)
EMME2 NNN(&NNN) USERPREF(&USERPREF) VOLSER(&VOLSER)-
     BANK(&BANK) BANKDISP(&BANKDISP) BANKTOT(&BANKTOT)-
     BANKSP(&BANKSP)  BANKINCR(&BANKINCR)-
     PUNCHSP(&PUNCHSP) PRINTSP(&PRINTSP) PLOTSP(&PLOTSP)-
     BATCHLIB(&BATCHLIB)-
     D002(&D002) D141(&D141) D201(&D201) D202(&D202) D211(&D211)-
     D221(&D221) D231(&D231) D301(&D301) D311(&D311) D411(&D411)-
     USERPROC(&USERPROC)
END
```

Banque: BANKLVL5

Procédure: LAVAL105

Fichier: **A017.P0108.EMPROCUS**

Données: **A017.P0108.EMLVL**

Dimensions: 105 zones

1 scénario

99 matrices

tableau B-5

procédure "LAVAL105"

```
PROC 0 LDI(*) LDO(*) LERO(*) MAP()-
   BANK(BANKLVL5) BANKDISP(OLD) BANKTOT(380) BANKSP(230)-
   BANKINCR(10)-
   PUNCHSP(0) PRINTSP(10)  PLOTSP(10)-
   BATCHLIB(EMLVL) D002(D002IN) D141(DEM105)-
   D201(MOD) D202(VEH) D211(NET105) D221(TRL) D231(PEN)-
   D301(GRP) D311(MAT105) D411(FCT)
CONTROL MAIN /*LIST SYMLIST
SET &NNN = &STR(002)
IF &BANKDISP EQ &STR(NEW) THEN SET &NNN = &STR(001)
SET MAPSTR = &STR()
SET BLIBSTR = &STR()
IF &LENGTH(&MAP) NE 0 THEN SET &MAPSTR =MAP(&MAP)
IF &LENGTH(&BATCHLIB) NE 0 THEN SET &BLIBSTR =BATCHLIB(&BATCHLIB)
EMME2 NNN(&NNN) &MAPSTR &BLIBSTR-
     BANK(&BANK) BANKDISP(&BANKDISP) BANKTOT(&BANKTOT) BANKSP(&BANKSP)-
     BANKINCR(&BANKINCR)-
     PRINTSP(&PRINTSP) PUNCHSP(&PUNCHSP) PLOTSP(&PLOTSP)-
     D002(&D002) D141(&D141) D201(&D201) D202(&D202) D211(&D211)-
     D221(&D221) D231(&D231) D301(&D301) D311(&D311) D411(&D411)
END
```

APPENDICE "C"

APPENDICE "C"

MODIFICATION AUX VALEURS PERMISES POUR LE PARAMÈTRE "BANKTOT"

Le processus de modification aux valeurs permises de BANKTOT ne doit être exécuté que par le responsable de l'entretien du logiciel emme2 , sur demande d'un usager.

Soit "btot", la nouvelle valeur permise à ajouter parmi les valeurs possibles du paramètres BANKTOT de la procédure EMME2 (voir section 2.3.3.), et pour fin d'exemple, posons btot = 3000 blocs.

À l'intérieur d'une session TSO, on exécutera les tâches suivantes:

1- EXEC **'EMME2.EMPROC(EMSETUP)'**
MKBDISK2 btot
Exemple: MKBDISK2 3000

2- édition du fichier
A017.P0108.EMCRTLIB(JCLBDISK)
substituer la nouvelle valeur de btot dans BDSKbtot pour les énoncés FORT.SYSIN et LKED.SYSLMOD:

tableau C-1

procédure "JCLBDISK"

```
//J7JCLBDI  JOB  (R07,'A017,0108,952,01,D',1,1,0,0010),
//   'SI552 L.JAMES',NOTIFY=NU552,CLASS=A,MSGCLASS=X,MSGLEVEL=(2,0)
/*ROUTE  PRINT RMT07    D=LINDA JAMES
//*
//*T  JCLBDISK  84 06 01  LGJL
//*
//*   COMPILE BDISK ROUTINE FOR A REQUIRED DATA BANK SIZE.
//*   THIS JOB IS RUN AFTER EXECUTING PROCEDURE EMPROC(MKBDISK2)
//*   TO CREATE THE SOURCE FILE IN LJ.EMMOD
//*
//JCLSELIB EXEC FORG1CL,CPARM='LOAD,NOSOURCE',PARM.LKED=NCAL
//FORT.SYSIN DD DSNAME=EMME2.LJ.EMMOD(BDSK1800),DISP=SHR
//LKED.SYSLMOD DD DSN=EMME2.EMLIB(BDSK1800),DISP=OLD
/*
```

3- soumettre en lot
A017.P0108.EMCRTLIB(JCLBDISK)
attendre le résultat

4- édition de la procédure
EMME2.EMPROC(EMME2)

a) substituer pour
SET &DSIZE = btot

b) et ajouter la ligne suivante:
IF &BANKTOT LT btot THEN SET &DSIZE = btot'
où btot' est la prochaine plus grande valeur (en faisant abstraction de btot) possible comme paramètre de BANKTOT

Exemple: on change pour
SET &DSIZE = 3000
et ajoute immédiatement après
IF &BANKTOT LT 3000 THEN SET &DSIZE = 2700

c) et on modifie aussi une ligne pour
IF &DSIZE NE btot THEN SET

&BDISK = ,'&EMPREF..&EMLIB(BDSK&DSIZE)'
Voir tableau C-2

tableau C-2

exemple de modification à la procédure EMME 2

```
PROC /*EMME2*/ 0 NNN(002)-
     USERPREF('A017.P0108') VOLSER(MTQ302)-
     BANK() BANKDISP(OLD) BANKTOT(380) BANKSP(300) BANKINCR(40) MAP()-
     PUNCHSP(0)  PRINTSP(1)  PLOTSP(0)-
     BATCHLIB() D002(D002IN) D141(D141IN) D201(D201IN) D202(D202IN)-
     D211(D211IN) D221(D221IN) D231(D231IN)-
     D301(D301IN) D311(D311IN) D411(D411IN)-
     USERPROC('A017.P0108.EMPROCX')-
     EMPREF(EMME2) EMLIB(EMLIB) EMOBJ(EMOBJ) EMLOAD(EMLOAD)
CONTROL MAIN /*LIST    SYMLIST
/* PROCEDURE TO EXECUTE EMME2
/*      84 06 08  L.G.M.
/* CALLING SEQUENCE:
/*      EMME2
/*
/* OPTIONAL PARAMETERS ARE SPECIFIED AS PARAMETER(VALUE) FOLLOWING
/* EMME2   (SEE EXAMPLES IN THE PROC STATEMENT)   IN THE PARAMETER
/* DESCRIPTIONS, ZZZ INDICATES THE USER NUMBER TAKEN FROM THE USER
/* ID NUZZZ, XXX.YYY INDICATES THE USER DSNAME PREFIX TAKEN FROM
/* PARAMETER USERPREF, EMME2 INDICATES THE SYSTEM PREFIX TAKEN FROM
/* PARAMTER EMPREF.
/*
/* OPTIONAL PARAMETERS:
/*      NNN       MODULE NUMBER     3 DIGITS. USE ONLY WHEN CREATING A
/*                                  NEW DATA BASE ( NNN(001) ).
/*      USERPREF USER PREFIX        USERPREF('XXX.YYY') USED TO FORM FIRST
/*                                  PART OF DSNAME OF ALL FILES EXCEPT
/*                                  EMME/2 SYSTEM FILES.
/*                                  E.G. XXX.YYY.EMPRTZZZ
/*      VOLSER    USER VOLUME       VOLUME SERIAL USED FOR ALL FILES EXCEPT
/*                SERIAL            EMME/2 SYSTEM FILES.
/*      BANK      DATA BANK FILE    BANK(XXXXXX) USED TO FORM DATA BANK
/*                                  FILE NAME:  XXX.YYY.XXXXXX
/*      BANKDISP DISP OF BANK       DATA BANK FILE MUST BE ALLOCATED BY
/*                                  USING BANKDISP(NEW) IN THIS PROC.
/*      BANKTOT   BANK SPACE        NUMBER OF BLOCKS (4628) FOR BANK
/*      BANKSP    PRIMARY SPACE     NUMBER OF BLOCKS (4628) PRIMARY
/*                                  ALLOCATION FOR BANK
/*      BANKINCR SECONDARY SP       NUMBER OF BLOCKS (4628) SECONDARY
/*                                  ALLOCATION FOR BANK
/*                                  NOTE: BANKTOT MUST BE LESS OR EQUAL TO
/*                                  BANKSP + 15*BANKINCR
/*      PUNCHSP   PUNCH SPACE       NUMBER OF BLOCKS (9040) FOR PRIMARY
/*                                  ALLOCATION FOR PUNCH FILE
/*                                  XXX.YYY.EMPCHZZZ, DELETED
/*                                  AND ALLOCATED AS NEW. SECONDARY ALLOC
/*                                  IS 5.
/*                                  PUNCHSP(0) WILL ALLOCATE A DUMMY
/*                                  FILE.
/*      PRINTSP   PRINTER SPACE     NUMBER OF BLOCKS (9044) FOR PRIMARY
/*                                  ALLOCATION OF EMME/2 PRINTER FILE
/*                                  XXX.YYY.EMPRTZZZ, DELETED AND
/*                                  ALLOCATED AS NEW. SECONDARY ALLOC 5.
/*                                  PRINTSP(0) WILL ALLOCATE A DUMMY
/*                                  FILE.
/*      PLOTSP    PLOT SPACE        NUMBER OF BLOCKS (8919) FOR PRIMARY
/*                                  ALLOCATION OF EMME/2 PLOT FILE
/*                                  XXX.YYY.EMPLTZZZ, DELETED AND
/*                                  ALLOCATED AS NEW. SECONDARY ALLOC 5.
/*                                  PLOTSP(0) WILL ALLOCATE A DUMMY FILE
/*      BATCHLIB BATCH INPUT  NAME OF PARTITIONED FILE CONTAINING
/*               LIBRARY      BATCH INPUT DATA
/*      D002     BATCH INPUT  MEMBER NAME OF BATCH INPUT TO MODULE
/*                            002 (DEVICE TABLE)
/*      D141 ETC              MEMBER NAME OF BATCH INPUT TO MODULE
/*                            141 ETC  (201,202,211,221,231,301,311,
/*                            411)
/*      EMPREF   EMME2 PREFIX PREFIX USED TO FORM FIRST PART
/*                            OF THE DSNAME OF EMME/2 SYSTEM FILES.
/*      EMLIB    SUBROUTINE LIB EMLIB(XXX) USED TO FORM EMME/2
/*                            SUBROUTINE LIBRARY NAME: EMME2.XXX
/*      EMOBJ    COMPILED     EMOBJ(XXX) USED TO FORM EMME/2 COMPILED
/*               MODULES      MODULE LIBRARY NAME: EMME2.XXX
/*      EMLOAD   EXECUTABLE   EMLOAD(XXX) USED TO FORM EXECUTABLE
/*                 MODULES      MODULE LIBRARY NAME: EMME2.XXX
/*
/*
/* EMME/2 USES THE FOLLOWING UNIT NUMBERS:
/*
/*    1  DATA BANK
/*    3  BATCH INPUT FILE
/*    4  PUNCH FILE
/*    5  DIALOG INPUT (NORMALLY FROM TERMINAL)
/*    6  DIALOG OUTPUT  (NORMALLY TO TERMINAL)
/*    7  PRINTER FILE
/*    8  PLOT FILE
/*   10  ERROR MESSAGE FILE
/*   98  BATCH MODE OR INTERACTIVE MODE INDICATOR FILE
/*   99  NEXT MODULE FILE
/*
/* EMME/2 SYSTEM FILES ARE:
/*       EMLIB   SUBROUTINE LIBRARY
/*       EMOBJ   COMPILED EMME/2 MODULE LIBRARY
/*       EMLOAD  LOAD MODULE LIBRARY
/*       EMPROC  TSO PROCEDURE LIBRARY
/*
IF &BANKTOT GT &BANKSP + 15 * &BANKINCR THEN DO
   WRITE *** BANKTOT(&BANKTOT) GREATER THAN BANKSP + 15 * BANKINCR
   EXIT
   END
SET USER = &SUBSTR(3:5,&SYSUID)
/*
/*   ATTACH PROCEDURE LIBRARY FOR THIS USER
/*
FREE ATTR(NEXTDCB) FILE(FT99F001)
ATTRIB NEXTDCB RECFM(F B) LRECL(80) BLKSIZE(2960) DSORG(PO)
DEL '&USERPREF..EMPRO&USER'
ALLOCATE FILE(FT99F001) DA('&USERPREF..EMPRO&USER') NEW-
   VOLUME(&VOLSER) UNIT(3380) SPACE(1,1) TRACKS DIR(1) USING(NEXTDCB)
FREE FILE(FT99F001)
FREE FILE(SYSPROC)
ALLOCATE FILE(SYSPROC) SHR-
   DA('SYS1.SYSPROC' '&EMPREF..EMPROC' '&USERPREF..EMPRO&USER')
/*
/*   FREE FILES AND ATTRIBUTE LISTS (DCBS)
/*
FREE ATTR(BANKDCB) FILE(FT01F001)
FREE ATTR(TKDCB) FILE(FT05F001 FT06F001)
FREE ATTR(LDADCB) FILE(FT04F001)
FREE ATTR(PRINTDCB) FILE(FT07F001)
FREE ATTR(PLOTDCB) FILE(FT08F001)
FREE FILE(FT10F001 FT98F001)
ATTRIB BANKDCB RECFM(F) LRECL(4628) BLKSIZE(4628) DSORG(PS) BUFNO(1)
ATTRIB TKDCB RECFM(F A) LRECL(132) BLKSIZE(132)
ATTRIB LDADCB RECFM(F B) LRECL(80) BLKSIZE(9040)
ATTRIB PRINTDCB RECFM(F B) LRECL(133) BLKSIZE(9044)
ATTRIB PLOTDCB RECFM(V B S) LRECL(182) BLKSIZE(8919)
/*
/*  DATA BANK
/*
ALLOCATE FILE(FT01F001) DA('&USERPREF..&BANK') &BANKDISP-
     VOLUME(&VOLSER)-
     UNIT(3380) SPACE(&BANKSP,&BANKINCR) BLKSIZE(4628) USING(BANKDCB)
/*
/*  DIALOG INPUT
/*
ALLOCATE FILE(FT05F001) DA(*) USING(TKDCB)
/*
/*  DIALOG OUTPUT
/*
ALLOCATE FILE(FT06F001) DA(*) USING(TKDCB)
/*
/*  PRINTER FILE
/*
IF &PRINTSP NE 0 THEN DO
   DEL '&USERPREF..EMPRT&USER'
   ALLOCATE FILE(FT07F001) DA('&USERPREF..EMPRT&USER') NEW-
      VOLUME(&VOLSER) UNIT(3380) SPACE(&PRINTSP,5) BLKSIZE(9044)
   FREE FILE(FT07F001)
   ALLOCATE FILE(FT07F001) DA('&USERPREF..EMPRT&USER') MOD-
      BLKSIZE(9044) USING(PRINTDCB) RELEASE
   END
ELSE-
     ALLOCATE FILE(FT07F001) DSNAME('NULLFILE') BLKSIZE(9044)-
        USING(PRINTDCB)
/*
/*  ERROR FILE
/*
ALLOCATE FILE(FT10F001) DATASET(*)
/*
/*  FILE TO COMMUNICATE TO NEXT MODULE
/*
ALLOCATE FILE(FT99F001) DA('&USERPREF..EMPRO&USER(NEXT)') OLD
/*
/*  BATCH OUTPUT OR PUNCH FILE
/*
IF &PUNCHSP GT 0 THEN DO
     DEL '&USERPREF..EMPCH&USER'
     ALLOC FILE(FT04F001) DATASET('&USERPREF..EMPCH&USER')-
     NEW KEEP CATALOG VOLUME(&VOLSER)-
     UNIT(3380) SPACE(&PUNCHSP,5) BLKSIZE(9040)
     FREE FILE(FT04F001)
     ALLOC FILE(FT04F001) DATASET('&USERPREF..EMPCH&USER')-
     MOD BLKSIZE(9040) USING(LDADCB) RELEASE
     END
ELSE-
     ALLOCATE FILE(FT04F001) DUMMY BLKSIZE(9040) USING(LDADCB)
```

tableau C-2 (suite)

```
/*
/*  PLOT FILE
/*
IF &PLOTSP GT 0 THEN DO
     DEL '&USERPREF..EMPLT&USER'
     ALLOC FILE(FT08F001) DATASET('&USERPREF..EMPLT&USER')-
     NEW KEEP CATALOG VOLUME(&VOLSER)-
     UNIT(3380) SPACE(&PLOTSP,5) BLKSIZE(8918)
     FREE FILE(FT08F001)
     ALLOC FILE(FT08F001) DATASET('&USERPREF..EMPLT&USER')-
     MOD BLKSIZE(8918) USING(PLOTDCB) RELEASE
     END
ELSE-
     ALLOCATE FILE(FT08F001) DUMMY BLKSIZE(8918) USING(PLOTDCB)
/*
/* SET TO INTERACTIVE MODE
/*
ALLOC FILE(FT98F001) SPACE(1) BLKSIZE(9040) USING(LDADCB)
OPENFILE FT98F001 OUTPUT
SET &FT98F001 = 0      /* 0= INTERACTIVE
PUTFILE FT98F001
CLOSFILE FT98F001
/*
/* SET TO CORRECT BDISK ROUTINE IF NEW DATA BANK
/*
SET &BDISK = &STR( )
IF &BANKDISP EQ OLD THEN GOTO L1
SET &DSIZE = 3000
IF &BANKTOT LT 3000 THEN SET &DSIZE = 3000
IF &BANKTOT LT 2700 THEN SET &DSIZE = 2600
IF &BANKTOT LT 2600 THEN SET &DSIZE = 1800
IF &BANKTOT LT 1800 THEN SET &DSIZE = 390
IF &DSIZE NE 3000 THEN SET &BDISK = ,'&EMPREF..&EMLIB(BDSK&DSIZE)'
/*
/* INITIALIZE LIBS USED INDICATOR
/*
L1:SET LIBS = &STR(N)
/*
/* INITIALIZE MODULE NUMBER
/*
SET MODNUM = &000
SET ZEROS = &STR(0000)
/*
/* LOOP TO EXECUTE EMME/2 MODULES AS SPECIFIED ON MENU.
/*
/* LINK AND CALL ARE ONLY USED FOR A NEW DATA BANK TO USE CORRECT
/* BDISK ROUTINE
/*
DO WHILE &MODNUM GE 000 AND &MODNUM LT 999
   SET &MODTEX = &SUBSTR(1:&EVAL(4-&LENGTH(&MODNUM)),&ZEROS)&MODNUM
   SET &MODTEX = &SUBSTR(2:4,&MODTEX)
/*
/* SET UP ANY BATCH FILES SPECIFIED
/*
SET FT03 = &STR(N)
IF &LENGTH(&BATCHLIB) EQ 0 THEN GOTO L3
IF &MODTEX EQ 002 AND &LENGTH(&D002) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D002)') SHR
     GOTO L2
     END
IF &MODTEX EQ 141 AND &LENGTH(&D141) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D141)') SHR
     GOTO L2
     END
IF &MODTEX EQ 201 AND &LENGTH(&D201) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D201)') SHR
     GOTO L2
     END
IF &MODTEX EQ 202 AND &LENGTH(&D202) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D202)') SHR
     GOTO L2
     END
IF &MODTEX EQ 211 AND &LENGTH(&D211) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D211)') SHR
     GOTO L2
     END
IF &MODTEX EQ 221 AND &LENGTH(&D221) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D221)') SHR
     GOTO L2
     END
IF &MODTEX EQ 231 AND &LENGTH(&D231) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D231)') SHR
     GOTO L2
     END
IF &MODTEX EQ 301 AND &LENGTH(&D301) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D301)') SHR
     GOTO L2
     END
IF &MODTEX EQ 311 AND &LENGTH(&D311) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D311)') SHR
     GOTO L2
     END
IF &MODTEX EQ 411 AND &LENGTH(&D411) NE 0 THEN DO
     ALLOC FILE(FT03F001) DA('&USERPREF..&BATCHLIB(&D411)') SHR
     GOTO L2
     END
GOTO L3
L2:SET FT03 = &STR(Y)
L3:IF &BDISK EQ &STR( ) THEN DO
/*   LINK '&EMPREF..&EMOBJ(E&MODTEX)' &MAP SIZE(999000)-
/*     LIB('FORT.FORTLIB','&EMPREF..&EMLIB','TKTCS.LOADLIB',-
/*      'SYS1.MAINTLIB')-
/*     LOAD('&EMPREF..EMTEMP(MOD)')
/*   CALL '&EMPREF..EMTEMP(MOD)'
/*   SET LIBS = &STR(Y)
     CALL '&EMPREF..&EMLOAD(E&MODTEX)'
     SET CCODE = &LASTCC
     END
   ELSE DO
     LOAD ('&EMPREF..&EMOBJ(E&MODTEX)' &BDISK) -
       &MAP SIZE(999000)-
       LIB('FORT.FORTLIB','&EMPREF..&EMLIB','TKTCS.LOADLIB',-
        'SYS1.MAINTLIB')
     SET CCODE = &LASTCC
     SET &BDISK = &STR( )
     SET LIBS = &STR(Y)
     END
   IF &FT03 EQ &STR(Y) THEN FREE FILE(FT03F001)
   IF &CCODE NE 000 THEN GOTO SETMOD
   %NEXT
     SET CCODE = &LASTCC
     SETMOD: SET MODNUM = &CCODE - 3000
     END
END
/*
/* FREE FILES
/*
%EMEND EMPREF(&EMPREF) LIBS(&LIBS) EMLIB(&EMLIB) EMOBJ(&EMOBJ)-
   USERPROC(&USERPROC)
DEL '&USERPREF..EMPRO&USER'
END
```

5- édition du fichier
EMME2.LJ.EMMOD(BASIC) ;
dans la routine BDISK,
on modifie une ligne pour obtenir

DEFINE FILE 1(btot,1157,U,NEXREC)

Exemple:
DEFINE FILE 1(3000,1157,U,NEXREC).

6- édition du fichier
A017.P0108.EMCRTLIB(JCLSUB1)
pour s'assurer que partout dans ce fichier on a
EMME2.LJ.EMMOD(BASIC)

7- soumettre
A017.P0108.EMCRTLIB(JCLSUB1)
et attendre le résultat

8- soumettre
A017.P0108.EMCRTLIB(JCLLOAD)
et attendre le résultat

9- la soumission précédente a créé
EMME.EMLOAD2
qu'on rebaptisera **EMME2.EMLOAD:**
DELETE **EMME2.EMLOAD**
RENAME **EMME2.EMLOAD2 AS**
EMME2.EMLOAD
(on peut utiliser l'option 3.2 de SPF).

APPENDICE "D"

APPENDICE "D"

INTERFACES - MATRICES UTPS / emmg2

D.1 **Programme MATUTEM**

But: Convertir une matrice UTPS en une matrice emme2

Données à fournir:

a) Les données suivantes sont lues en "stream imput", c'est-à-dire à partir de l'unité 5:

   # le numéro de matrice emme2 qui sera générée (MFi)
     format (I1)

   # le nom de la matrice emme2 générée (6 caractères)
     format (6A1)

   # la description de la matrice emme2 générée (40 caractères)
     format (40A1)

b) L'unité sur laquelle on lit la matrice UTPS dans le DATA du début du programme sous la variable NIN (NIN = 8 par défaut) et également la description de cette unité à la fin du programme (ligne de JCL).

c) L'unité sur laquelle on écrit la matrice emme2 dans le DATA du début du programme sous la variable NOUT (NOUT = 9 par défaut) et également la description de cette unité à la fin du programme (ligne de JCL).

   Il faut prévoir l'espace nécessaire pour un fichier d'enregistrements de 72 caractères où le nombre d'enregistrements est environ le nombre de paires O/D de valeur non nulle de la matrice UTPS, divisé par cinq.

Description:

Ce programme lit une matrice UTPS selon l'un des quatres formats possibles: compressé, 1-byte, 2-bytes ou 4-bytes. Puisque ce format est inclus comme donnée, dans la lecture de la matrice UTPS, le programme décode, automatiquement le format approprié et agit en conséquence.

Une restriction est imposée quant à la question du "table number", c'est-à-dire du nombre de matrices à lire. Cet attribut qui est également inclus comme donnée des enregistrements lues de la matrice UTPS doit être égal à 1.

Le fichier de sortie contenant la matrice emm92 résultante est composé d'enregistrements de 72 caractères, car cette matrice est codée sous forme de caractères.

Toutes les paires O/D de valeur non nulle provenant de la matrice UTPS, sont écrites dans la matrice emm92 .

D.2 Programme MATEMUT

But: Convertir une matrice emme2 en une matrice UTPS.

Données à fournir:

a) Les données suivantes sont lues en "stream imput", c'est-à-dire à partir de l'unité 5:

le format dans lequel on veut créer la matrice UTPS soit

- COMP format compressé
- 1BYT format 1-byte
- 2BYT format 2-bytes
- 4BYT format 4-bytes

format (1A4)

le nombre de centroïdes qui formeront la matrice UTPS (variable NCEN)
format (1I4)

le facteur multiplicatif qui appliqué à chaque valeur O/D de la matrice emme2 donne la valeur O/D à sauver sur la matrice UTPS. Cette dernière valeur est arrondie et entière (par défaut 1)
format (1I4)

une chaîne de caractères fixe qui sert à la manipulation de caractères à l'intérieur du programme;
format (10A1)
cette chaîne doit être '0123456789.'

b) L'unité sur laquelle on lit la matrice emme2 dans le DATA du début du programme sous la variable NIN (NIN = 8 par défaut) et également la description de cette unité à la fin du programme (ligne de JCL).

Il faut prévoir que ce fichier a des enregistrements de 72 caractères et qu'il contient comme les deux premiers enregistrements, des enregistrements bidons (habituellement ces enregistrements contiennent la description de la matrice emme2 , telle que sortant du logiciel emme2).

c) L'unité sur laquelle on écrit la matrice UTPS dans le DATA du début du programme sous la variable NOUT (NOUT = 9 par défaut) et également la description de cette unité à la fin du programme (ligne de JCL).

Il faut prévoir l'espace nécessaire pour la matrice UTPS (dépendant du format demandé)

soit N = nombre de paires O/D de la matrice emme2 à lire
(nombre d'enregistrements - 2)/5

La longueur maximale des enregistrements du fichier résultant en mots de 32 bits doit être de LRECL mots (par défaut 400) provenant du programme.
Alors le lrecl du JCL est

LRECL * 4 + 1 en bytes

le blksize du JCL est

LRECL * 4 + 8 en bytes

Le nombre d'enregistrement maximum pour le format 4-bytes est

NCEN * (int (NCEN/LRECL) + 1)

Pour le format 2-bytes est

NCEN * (int (NCEN/2*LRECL) + 1)

Pour le format 1-byte est

NCEN * (int (NCEN/4*LRECL) + 1)

Pour le format compressé est

NCEN + int(N - NCEN/LRECL) + 1 - N/NCEN

Description:

Ce programme lit une matrice emme2 ayant comme deux premiers enregistrements sa description.

Les paires O/D, n'apparaissant pas dans la matrice emme2 , auront leur valeur à zéro. De plus les valeurs des paires O/D de la matrice emme2 seront multipliées par le facteur multiplicatif et arrondies à l'unité près, puisque ce sont des valeurs entières qui seront écrites dans la matrice UTPS.

La longueur maximale des enregistrements de la matrice UTPS est donnée par la variable LRECL (en mots de 32 bits). Par défaut cette variable est à 400, mais on peut utiliser toute autre valeur en changeant le DATA correspondant au début du programme.

Dans le cas des formats 1-byte et 2-bytes, les valeurs résultantes des paires O/D sont testées pour voir à la restriction de non-négativité et à ce que ces valeurs soient plus petites ou égales à 255 (format 1-byte) ou à 32 767 (format 2-bytes), si cete restriction n'est pas respectée alors un message est imprimé et les valeurs sont mises à 255 (format 1-byte) ou à 32 767 (format 2-bytes).

Enfin si il y a des paires O/D pour lesquelles la destination est plus grande que NCEN alors ces paires sont ignorées et un message est imprimé pour cette destination hors limite.

D.3 Matrices de temps routier pour isochrones

Il y a deux façons de procéder pour créer des matrices de temps routiers emme/2 dans un format compatible aux programmes de traçage d'isochrones développés à la D.G.T.T.P.

La première est, de loin, beaucoup plus facile que l'autre, mais ne peut s'exécuter qu'en classes de nuit (M ou N) sur l'ordinateur central. L'autre façon peut fonctionner en classe "C", de jour, mais nécessite temporairement beaucoup d'espace-disque.

D.3.1 Méthode "Nuit"

Le tableau D-1 montre un exemple du programme à soumettre pour fabriquer, de nuit, la matrice des temps routiers emme/2 , sous format UTPS.

L'usager est invité à se faire une copie du fichier **A017.P0108.EMJCLMTL(PUNCHISO)**

L'usager doit s'assurer que le nom du fichier qu'il veut créer n'existe pas déjà. Par défaut, ce fichier portera le nom **A017.P0108.MAT.EMMEUTPS.ISO** . S'il existe et qu'on n'en a plus besoin, on le détruit dans l'étape "01" du travail. L'usager est entièrement libre de donner le nom qu'il veut au fichier de sortie.

Il est essentiel de vérifier que, dans la banque de donnée emme/2 en question, la "switch 4" est allumé ("ON"). Pour être certain, on fait, en session interactive au niveau du menu principal,

4 = ON

tableau D-1

programme "Nuit" pour conversion des matrices

```
//J7PUNCHI JOB (R07,'A017,0108,188,03,D',6,8,0,0010),
//  'SI104 A.BABIN',CLASS=N,NOTIFY=NU104,MSGCLASS=X,MSGLEVEL=(1,1)
//ETAPE00  EXEC  PGM=E314
//*
//*   SESSION EN BATCH DE EMME/2 POUR
//*     PUNCHER UNE MATRICE
//*
//STEPLIB DD DSN=EMME2.EMLOAD,DISP=SHR
//FT06F001 DD SYSOUT=*
//FT07F001 DD SYSOUT=*,DCB=(RECFM=FA,BLKSIZE=133)
//FT01F001 DD DSN=A017.P0108.BANKMTL2,DISP=OLD,DCB=(BUFNO=1)
//FT10F001 DD SYSOUT=*,DCB=(RECFM=FA,BLKSIZE=133)
//FT99F001 DD DUMMY                    /
//FT98F001 DD DUMMY
//FT04F001 DD DSN=&&TEMPEM,UNIT=PUBLIC,DISP=(,PASS),
//   SPACE=(TRK,(350,20),RLSE),DCB=(RECFM=FB,LRECL=80,BLKSIZE=18960)
//*
//*   S'ASSURER QUE  SWITCH 4  EST ON SUR
//*     LA BANQUE DE DONNEES EMME/2   (SINON IL FAUT DONNER
//*         UNE REPONSE DE PLUS DANS CE QUI SUIT, ET PAR
//*         CONSEQUENT ENTRAINE UNE ERREUR SI ON NE LE FAIT PAS)
//*
//FT05F001 DD *
4
2
6,1
4
N
3
MF2

N
N
N
Q
/*
//ETAPE01  EXEC   PGM=IEFBR14
//DD01 DD DSN=A017.P0108.MAT.EMMEUTPS.ISO,DISP=(OLD,DELETE)
//ETAPE02  EXEC  FORG1CLG
//FORT.SYSIN DD  *
C
C   PROGRAMME : MATEMUT  (MATRICES EMME/2  --)  MATRICES UTPS)
C
C   INTERFACE SUR LES MATRICES
C     MATRICES EMME/2 CONVERTIES EN MATRICES UTPS
C
C   EN ENTREE, ON FOURNIT:
C      (1A4) LE TYPE DES MATRICES UTPS A LA SORTIE
C            SOIT  COMP   FORMAT COMPRESSION
C                  1BYT   FORMAT EN 1-BYTE
C                  2BYT   FORMAT EN 2-BYTES
C                  4BYT   FORMAT EN 4-BYTES
C      (I4)  LE NOMBRE DE CENTROIDES
C      (I4)  UN FACTEUR MULTIPLICATIF S'APPLIQUANT A CHAQUE VALEUR O/D
C              (CAR LES VALEURS UTPS SONT SAUVEES COMME ENTIERS)
C      (13A1)  INSERTION DE LA CHAINE DE CARACTERES: '0123456789.*M'
C
C   RESTRICTION: ON LIT PLUSIEURS MATRICES A LA FOIS DE
C                EMME/2, C'EST DONC DIRE QUE TABLE NUMBER EST
C                MISE AU NUMERO DE LA MATRICE COURANTE LUE
C                DEBUTANT AVEC UN  ET INCREMENTANT DE UN.
C                (BITS 8-15 DE MOT 2 DES ENREGISTREMENTS DE UTPS)
C
C   S'IL Y A PLUS QU'UNE MATRICE EMME/2 A TRANSFORMER
C      ALORS IL FAUT SOUMETTRE PAR LA SUITE LE PROGRAMME SUIVANT:
C            A017.P0108.ABMATEM(MATPOSTX)
C      QUI PREND COMME FICHIER D'ENTREE, LE FICHIER DE SORTIE
C      DU PRESENT PROGRAMME, LE FICHIER INTERMEDIAIRE ETANT:
C            A017.P0108.MAT.INTER.POSTX
C   S'IL N'Y A QU'UNE MATRICE EMME/2 A TRANSFORMER
C      ALORS LE FICHIER DE SORTIE DU PRESENT PROGRAMME EST
C      LE RESULTAT FINAL  I.E. ON PEUT CHANGER
C            A017.P0108.MAT.INTER.POSTX
C      POUR LE FICHIER QU'ON VEUT SAUVER (VOIR POUR FT08F001)
C
      LOGICAL FIRST,LAST
      INTEGER LIG(72),ITYP,NCEN,NWORDS(1000)
      REAL VECT(1000)
      COMMON /NUM/ INUM(13)
      DATA JCOMP,I1BYT,I2BYT,I4BYT/'COMP','1BYT','2BYT','4BYT'/
      DATA NIN,NOUT/8,9/
C
C   NIN = UNITE D'ENTREE DE LA MATRICE EMME/2
C   NOUT = UNITE DE SORTIE DE LA MATRICE UTPS
C
      DATA LRECL/400/
C
C   LRECL : LONGUEUR DES RECORDS LOGIQUES EN MOTS DE 4 BYTES
C           EN FAIT CETTE LONGUEUR CORRESPOND A
C           LRECL*4 + 4  BYTES DU LRECL DU JCL POUR UNITE NOUT
C           LRECL*4 + 8  BYTES DU BLKSIZE DU JCL DE UNITE NOUT
C
C   PASSER LES DEUX PREMIERES LIGNES DE LA MATRICE EMME/2
C    (I.E. 'T MATRIX' ET LA LIGNE DE LA DESCRIPTION DE LA MATRICE)
C
      READ(NIN,10) LIG
      READ(NIN,10) LIG
   10 FORMAT(72A1)
C
C   LIRE LE TYPE DE MATRICE UTPS ET LE NOMBRE DE CENTROIDES
C      SUR L'UNITE 5 (STREAM INPUT)
C
      READ(5,20) ITYP
      READ(5,30) NCEN
   20 FORMAT(1A4)
   30 FORMAT(1I4)
      WRITE(6,25)
   25 FORMAT('1'//'     - -  M A T E M U T  - -'//)
      READ(5,30) IFACT
      FACT=FLOAT(IFACT)
      READ(5,35) INUM
   35 FORMAT(13A1)
      LAST=.FALSE.
      FIRST=.TRUE.
      ITABN=1
      ICOMP=0
      NTOT=0
C
C   ITABN = TABLE NUMBER (BITS 8-15 WORD 2 ON UTPS MATRIX RECORDS)
C   CONTINUATION RECORDS (BIT 0 WORD 2 ON UTPS MATRIX RECORDS, BIT ON)
C   ICOMP = FORMAT CODE (BITS 1-7 WORD 2 ON UTPS MATRIX RECORDS)
C
      IF(ITYP.EQ.JCOMP) ICOMP=0
      IF(ITYP.EQ.I1BYT) ICOMP=1
      IF(ITYP.EQ.I2BYT) ICOMP=2
      IF(ITYP.EQ.I4BYT) ICOMP=4
      IF(ICOMP.EQ.0) VMAX=999999.
      IF(ICOMP.EQ.1) VMAX=255
      IF(ICOMP.EQ.2) VMAX=32767
      IF(ICOMP.EQ.4) VMAX=999999.
      DO 40 I=1,1000
      VECT(I)=0.
   40 CONTINUE
C
C   IWORD = NOMBRE DE MOTS - 2  (ENREGISTREMENTS UTPS)
C
      IWORD=NCEN
      IF(ICOMP.EQ.1) IDIV=4
      IF(ICOMP.EQ.2) IDIV=2
      IF(ICOMP.EQ.4) IDIV=1
      IF(ICOMP.GE.1) IWORD = (NCEN-1)/IDIV + 1
      NLIG=0
   50 READ(NIN,10,END=500) LIG
      NLIG=NLIG+1
      IF(NLIG.LE.1000) WRITE(6,1111) LIG
 1111 FORMAT(' --->',72A1)
      I=0
   37 I=I+1
      IF(I.GT.10) GOTO 52
      IF(LIG(I).NE.INUM(13)) GOTO 37
      GOTO 50
   52 CALL SCANO(LIG,NO)
      IF(.NOT.FIRST) GOTO 53
      NORIG = NO
      FIRST = .FALSE.
   53 IF(NORIG.EQ.NO) GOTO 480
  490 NWORDS(2)=ICOMP*16777216 + ITABN*65536 + NORIG
      IF(ICOMP.NE.0) GOTO 100
      NB=0
      DO 70 I=1,NCEN
      IF(VECT(I).EQ.0.) GOTO 70
      IVAL1 = IFIX(VECT(I)*FACT+0.5)
   60 IVAL2=IVAL1
      IF(IVAL1.GT.262143) IVAL2=262143
      NB=NB+1
      NWORDS(NB+2)=I*262144+IVAL2
      NTOT=NTOT+IVAL2
      IF(IVAL1.LE.262143) GOTO 70
      IVAL1=IVAL1-262143
      GOTO 60
   70 CONTINUE
      NWORDS(1)=NB+1
      GOTO 400
C
  100 NWORDS(1)=IWORD+1
      IF(ICOMP.NE.4) GOTO 120
      DO 110 I=1,IWORD
```

tableau D-1 (suite)

```
      NWORDS(I+2)=IFIX(VECT(I)*FACT+0.5)
      NTOT=NTOT+NWORDS(I+2)
  110 CONTINUE
      GOTO 400
C
  120 IF(ICOMP.NE.2) GOTO 200
      DO 130 I=1,IWORD
      NW1=IFIX(VECT(I*2-1)*FACT+0.5)
      NW2=IFIX(VECT(I*2)*FACT+0.5)
      IF(NW1.GE.0.AND.NW1.LE.32767) GOTO 125
      II=I*2-1
      WRITE(6,122) NORIG,II,NW1
  122 FORMAT(//'  ATTENTION ORIGINE ',I4,' A DESTINATION ',I5,
     X    ' VALEUR ',I12,'  MISE A 32767')
      NW1=32767
  125 IF(NW2.GE.0.AND.NW2.LE.32767) GOTO 132
      II=2*I
      WRITE(6,122) NORIG,II,NW2
      NW2=32767
  132 CONTINUE
      NWORDS(I+2)=NW1*65536 + NW2
      NTOT=NTOT+NW1+NW2
  130 CONTINUE
      GOTO 400
C
  200 DO 300 I=1,IWORD
      NW1=IFIX(VECT(I*4-3)*FACT+0.5)
      NW2=IFIX(VECT(I*4-2)*FACT+0.5)
      NW3=IFIX(VECT(I*4-1)*FACT+0.5)
      NW4=IFIX(VECT(I*4)*FACT+0.5)
      IF(NW1.GE.0.AND.NW1.LE.255) GOTO 220
      II=I*4-3
      WRITE(6,225) NORIG,II,NW1
  225 FORMAT(//'  ATTENTION ORIGINE ',I5,' A DESTINATION ',I5,
     X     ' VALEUR ',I8,' MISE A 255')
      NW1=255
  220 IF(NW2.GE.0.AND.NW2.LE.255) GOTO 240
      II=I*4-2
      WRITE(6,225) NORIG,II,NW2
      NW2=255
  240 IF(NW3.GE.0.AND.NW3.LE.255) GOTO 260
      II=I*4-1
      WRITE(6,225) NORIG,II,NW3
      NW3=255
  260 IF(NW4.GE.0.AND.NW4.LE.255) GOTO 310
      II=I*4
      WRITE(6,225) NORIG,II,NW4
      NW4=255
  310 CONTINUE
      NWORDS(I+2)=NW1*16777216+NW2*65536+NW3*256+NW4
      NTOT=NTOT+NW1+NW2+NW3+NW4
  300 CONTINUE
C
C   ECRIRE LE OU LES RECORDS SUR UNITE DE SORTIE NOUT
C    AVEC LE BIT DE CONTINUATION A 1 SAUF LE DERNIER RECORD
C    S'IL Y A LIEU
C
  400 CONTINUE
      L=NWORDS(1)+1
      LP=2
      IF(L.LE.LRECL) GOTO 420
      LRECLM=LRECL-1
C
C   METTRE LE BIT 0 (BIT LE PLUS A GAUCHE) DU MOT 2 A UN
C   2 ** 31 = 2147483648 = 2147483647 + 1
C
      NWW2=NWORDS(2)+2147483647
      NWW2=NWW2+1
      WRITE(NOUT) LRECLM,NWW2,(NWORDS(I),I=3,LRECL)
      LP=LRECL
  410 IF(L-LP.LE.LRECL-2) GOTO 420
      I1=LP+1
      LP=LP+LRECL-2
      WRITE(NOUT) LRECLM,NWW2,(NWORDS(I),I=I1,LP)
      GOTO 410
  420 I1=LP+1
      LLP=L-LP+1
      WRITE(NOUT) LLP,NWORDS(2),(NWORDS(I),I=I1,L)
      DO 450 I=1,1000
      VECT(I)=0.
  450 CONTINUE
      IF(LAST) GOTO 510
      IF(NORIG.LE.NO) GOTO 470
      WRITE(6,460) ITABN,NTOT
  460 FORMAT('   UTPS MATRIX CREATED  TABLE NUMBER = ',I4/
     +      20X,'MATRIX TOTAL = ',I12)
      NTOT=0
      ITABN=ITABN+1
  470 NORIG=NO
C
  480 CALL SCAN(LIG,VECT,NCEN,VMAX)
      GOTO 50
C
  500 LAST=.TRUE.
      GOTO 490
C
  510 CONTINUE
      WRITE(6,460) ITABN,NTOT
      STOP
      END
      SUBROUTINE SCANO(L,N)
      INTEGER L(72)
      IL=0
      NO=0
   10 IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
      IF(INOS.EQ.-1) GOTO 10
   20 NO=NO*10+INOS
      IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
      IF(INOS.GE.0.AND.INOS.LE.9) GOTO 20
      N=NO
      RETURN
      END
      INTEGER FUNCTION INTC(LC)
      COMMON /NUM/ INUM(13)
      DO 10 I=1,12
      IF(LC.NE.INUM(I)) GOTO 10
      INTC=I-1
      RETURN
   10 CONTINUE
      INTC=-1
      RETURN
      END
      SUBROUTINE SCAN(L,V,NCEN,VMAX)
      REAL V(1)
      INTEGER L(72)
      IL=0
   10 IL=IL+1
      IF(IL.GT.72) RETURN
      IF(INTC(L(IL)).EQ.-1) GOTO 10
   20 IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
      IF(INOS.GE.0.AND.INOS.LE.9) GOTO 20
   30 IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
   35 ND=0
      IF(INOS.EQ.-1) GOTO 30
   40 ND=ND*10+INOS
      IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
      IF(INOS.GE.0.AND.INOS.LE.9) GOTO 40
   50 IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
      X=0.
      Y=0.
      YY=0.
      IF(INOS.EQ.-1) GOTO 50
      IF(INOS.NE.11) GOTO 58
      V(ND)=VMAX
   55 IL=IL+1
      INOS=INTC(L(IL))
      IF(INOS.EQ.11) GOTO 55
      GOTO 35
   58 IF(INOS.EQ.10) GOTO 70
   60 X=X*10.+FLOAT(INOS)
      IL=IL+1
      IF(IL.GT.72) GOTO 90
      INOS=INTC(L(IL))
      IF(INOS.GE.0.AND.INOS.LE.9) GOTO 60
      IF(INOS.NE.10) GOTO 90
   70 IL=IL+1
      IF(IL.GT.72) GOTO 80
      INOS=INTC(L(IL))
      IF(INOS.EQ.-1) GOTO 80
      Y=Y*10.+FLOAT(INOS)
      YY=YY-1.
      GOTO 70
   80 X=X+(Y*10.**YY)
   90 IF(ND.LE.0.OR.ND.GT.NCEN) GOTO 100
      V(ND)=X
      IF(IL.LT.72) GOTO 30
      RETURN
  100 WRITE(6,110) ND,NCEN,X
  110 FORMAT(/'  ATTENTION !!  DESTINATION ZONE = ',I6,
     +    '  (MAX. ZONE IS ',I6,')  WITH  VALUE = ',F10.3,/,
     +    20X,'IS SKIPPED')
      RETURN
      END
/*
//GO.SYSIN  DD  *
2BYT
 699
  10
0123456789.*M
/*
//GO.FT08F001  DD  DSN=&&TEMPEM,DISP=(OLD,DELETE)
//GO.FT09F001  DD  DSN=A017.P0108.MAT.EMMEUTPS.ISO,UNIT=3380,
//  VOL=SER=MTQ302,SPACE=(TRK,(10,3),RLSE),DISP=(,CATLG),
//  DCB=(RECFM=VBS,LRECL=1604,BLKSIZE=1608)
```

Après exécution du travail, la matrice des temps routiers, en format UTPS, existera comme table no.1, dans le fichier **A017.P0108.MAT.EMMEUTPS.ISO** .

D.3.2 Méthode "Jour"

Cette façon de faire demande temporairement environ 200 pistes sur l'unité de disques 3380.

Étape 1:

Faire un "Punch" de la matrice, à l'intérieur de la banque choisie, dans emme2 :
MTLRR PUNCHSP(1000)

Changer les paramètres du module 3.14:
dimension for in-line report: 6,1

Exécuter le punch de la matrice contenant les temps simulés, habituellement MF2.
Le résultat sera versé au fichier
A017.P0108.EMPCHZZZ

Étape 2:

En se servant du fichier
A017.P0108.ABMATEM(MATEMUT)
comme exemple (tableau D-2), s'en faire une copie avec les modifications suivantes:

a) Insérer ses propres cartes JOB, avec class = C.

b) À l'étape 00 du travail, donner s'il y a lieu le nom du fichier à détruire. Normalement, on détruira, s'il existe, le fichier
A017.P0108.MAT.EMMEUTPS.ISO .
si cette étape n'est pas nécessaire, l'éliminer.

c) À la fin du fichier-programme, il faut avoir

2BYT

699

10

d) S'assurer que le fichier d'entrée [FT08F001 de l'étape GO] porte le nom

A017.P0108.EMPCHZZZ

e) Donner au fichier de sortie le nom que vous voulez. Habituellement, on aurait

A017.P0108.MAT.EMMEUTPS.ISO .

La table 1 de ce fichier est la matrice des temps routiers emme2 , en format UTPS.

tableau D-2

programme "Jour" pour conversion des matrices

```
//J7MATEMU JOB (R00,'A017,0108,188,03,D',3,8,0,0010),
//  'SI552 A BABIN',CLASS=C,NOTIFY=NU188,MSGCLASS=X,MSGLEVEL=(1,1)
//ETAPE00  EXEC   PGM=IEF8R14
//DD01 DD DSN=A017.P0108.MAT.EMMEUTPS.ISO,DISP=(OLD,DELETE)
//ETAPE01  EXEC  FORG1CLG
//FORT.SYSIN DD  *
C
C   PROGRAMME : MATEMUT  (MATRICES EMME/2  --)  MATRICES UTPS)
C
C   INTERFACE SUR LES MATRICES
C     MATRICES EMME/2 CONVERTIES EN MATRICES UTPS
C
C   EN ENTREE, ON FOURNIT:
C      (1A4) LE TYPE DES MATRICES UTPS A LA SORTIE
C            SOIT  COMP   FORMAT COMPRESSION
C                  1BYT   FORMAT EN 1-BYTE
C                  2BYT   FORMAT EN 2-BYTES
C                  4BYT   FORMAT EN 4-BYTES
C      (I4)  LE NOMBRE DE CENTROIDES
C      (I4)  UN FACTEUR MULTIPLICATIF S'APPLIQUANT A CHAQUE VALEUR O/D
C              (CAR LES VALEURS UTPS SONT SAUVEES COMME ENTIERS)
C      (13A1)  INSERTION DE LA CHAINE DE CARACTERES: '0123456789.*M'
C
C   RESTRICTION: ON LIT PLUSIEURS MATRICES A LA FOIS DE
C                EMME/2, C'EST DONC DIRE QUE TABLE NUMBER EST
C                MISE AU NUMERO DE LA MATRICE COURANTE LUE
C                DEBUTANT AVEC UN  ET INCREMENTANT DE UN.
C                (BITS 8-15 DE MOT 2 DES ENREGISTREMENTS DE UTPS)
C
C   S'IL Y A PLUS QU'UNE MATRICE EMME/2 A TRANSFORMER
C      ALORS IL FAUT SOUMETTRE PAR LA SUITE LE PROGRAMME SUIVANT:
C            A017.P0108.ABMATEM(MATPOSTX)
C      QUI PREND COMME FICHIER D'ENTREE, LE FICHIER DE SORTIE
C      DU PRESENT PROGRAMME, LE FICHIER INTERMEDIAIRE ETANT:
C            A017.P0108.MAT.INTER.POSTX
C   S'IL N'Y A QU'UNE MATRICE EMME/2 A TRANSFORMER
C      ALORS LE FICHIER DE SORTIE DU PRESENT PROGRAMME EST
C      LE RESULTAT FINAL  I.E. ON PEUT CHANGER
C            A017.P0108.MAT.INTER.POSTX
C      POUR LE FICHIER QU'ON VEUT SAUVER (VOIR POUR FT08F001)
C
      LOGICAL FIRST,LAST
      INTEGER LIG(72),ITYP,NCEN,NWORDS(1000)
      REAL VECT(1000)
      COMMON /NUM/ INUM(13)
      DATA JCOMP,I1BYT,I2BYT,I4BYT/'COMP','1BYT','2BYT','4BYT'/
      DATA NIN,NOUT/8,9/
C
C   NIN = UNITE D'ENTREE DE LA MATRICE EMME/2
C   NOUT = UNITE DE SORTIE DE LA MATRICE UTPS
C
      DATA LRECL/400/
C
C   LRECL : LONGUEUR DES RECORDS LOGIQUES EN MOTS DE 4 BYTES
C           EN FAIT CETTE LONGUEUR CORRESPOND A
C           LRECL*4 + 4  BYTES DU LRECL DU JCL POUR UNITE NOUT
C           LRECL*4 + 8  BYTES DU BLKSIZE DU JCL DE UNITE NOUT
C
C
C   PASSER LES DEUX PREMIERES LIGNES DE LA MATRICE EMME/2
C    (I.E. 'T MATRIX' ET LA LIGNE DE LA DESCRIPTION DE LA MATRICE)
C
      READ(NIN,10) LIG
      READ(NIN,10) LIG
   10 FORMAT(72A1)
C
C   LIRE LE TYPE DE MATRICE UTPS ET LE NOMBRE DE CENTROIDES
C     SUR L'UNITE 5 (STREAM INPUT)
C
      READ(5,20) ITYP
      READ(5,30) NCEN
   20 FORMAT(1A4)
   30 FORMAT(1I4)
      WRITE(6,25)
   25 FORMAT('1'//'     - - M A T E M U T  - -'//)
      READ(5,30) IFACT
      FACT=FLOAT(IFACT)
      READ(5,35) INUM
   35 FORMAT(13A1)
      LAST=.FALSE.
      FIRST=.TRUE.
      ITARN=1
      ICOMP=0
      NTOT=0
C
C   ITARN = TABLE NUMBER (BITS 8-15 WORD 2 ON UTPS MATRIX RECORDS)
C   CONTINUATION RECORDS (BIT 0 WORD 2 ON UTPS MATRIX RECORDS, BIT ON)
C   ICOMP = FORMAT CODE (BITS 1-7 WORD 2 ON UTPS MATRIX RECORDS)
C
      IF(ITYP.EQ.JCOMP) ICOMP=0
      IF(ITYP.EQ.I1BYT) ICOMP=1
      IF(ITYP.EQ.I2BYT) ICOMP=2
      IF(ITYP.EQ.I4BYT) ICOMP=4
      IF(ICOMP.EQ.0) VMAX=999999.
      IF(ICOMP.EQ.1) VMAX=255
      IF(ICOMP.EQ.2) VMAX=32767
      IF(ICOMP.EQ.4) VMAX=999999.
      DO 40 I=1,1000
      VECT(I)=0.
   40 CONTINUE
C
C   IWORD = NOMBRE DE MOTS - 2  (ENREGISTREMENTS UTPS)
C
      IWORD=NCEN
      IF(ICOMP.EQ.1) IDIV=4
      IF(ICOMP.EQ.2) IDIV=2
      IF(ICOMP.EQ.4) IDIV=1
      IF(ICOMP.GE.1) IWORD = (NCEN-1)/IDIV + 1
   50 READ(NIN,10,END=500) LIG
      I=0
   37 I=I+1
      IF(I.GT.10) GOTO 52
      IF(LIG(I).NE.INUM(13)) GOTO 37
      GOTO 50
   52 CALL SCANO(LIG,NO)
      IF(.NOT.FIRST) GOTO 53
      NORIG = NO
      FIRST = .FALSE.
   53 IF(NORIG.EQ.NO) GOTO 480
  490 NWORDS(2)=ICOMP*16777216 + ITARN*65536 + NORIG
      IF(ICOMP.NE.0) GOTO 100
      NB=0
      DO 70 I=1,NCEN
      IF(VECT(I).EQ.0.) GOTO 70
      IVAL1 = IFIX(VECT(I)*FACT+0.5)
   60 IVAL2=IVAL1
      IF(IVAL1.GT.262143) IVAL2=262143
      NB=NB+1
      NWORDS(NB+2)=I*262144+IVAL2
      NTOT=NTOT+IVAL2
      IF(IVAL1.LE.262143) GOTO 70
      IVAL1=IVAL1-262143
      GOTO 60
   70 CONTINUE
      NWORDS(1)=NB+1
      GOTO 400
C
  100 NWORDS(1)=IWORD+1
      IF(ICOMP.NE.4) GOTO 120
      DO 110 I=1,IWORD
      NWORDS(I+2)=IFIX(VECT(I)*FACT+0.5)
      NTOT=NTOT+NWORDS(I+2)
  110 CONTINUE
      GOTO 400
C
  120 IF(ICOMP.NE.2) GOTO 200
      DO 130 I=1,IWORD
      NW1=IFIX(VECT(I*2-1)*FACT+0.5)
      NW2=IFIX(VECT(I*2)*FACT+0.5)
      IF(NW1.GE.0.AND.NW1.LE.32767) GOTO 125
      II=I*2-1
      WRITE(6,122) NORIG,II,NW1
  122 FORMAT(///'  ATTENTION ORIGINE ',I4,' A DESTINATION ',I5,
     X       ' VALEUR ',I12,' MISE A 32767')
      NW1=32767
  125 IF(NW2.GE.0.AND.NW2.LE.32767) GOTO 132
      II=I*2
      WRITE(6,122) NORIG,II,NW2
      NW2=32767
  132 CONTINUE
      NWORDS(I+2)=NW1*65536 + NW2
      NTOT=NTOT+NW1+NW2
  130 CONTINUE
      GOTO 400
C
  200 DO 300 I=1,IWORD
      NW1=IFIX(VECT(I*4-3)*FACT+0.5)
      NW2=IFIX(VECT(I*4-2)*FACT+0.5)
      NW3=IFIX(VECT(I*4-1)*FACT+0.5)
      NW4=IFIX(VECT(I*4)*FACT+0.5)
      IF(NW1.GE.0.AND.NW1.LE.255) GOTO 220
      II=I*4-3
      WRITE(6,225) NORIG,II,NW1
  225 FORMAT(///'  ATTENTION ORIGINE ',I5,' A DESTINATION ',I5,
     X       ' VALEUR ',I8,' MISE A 255')
      NW1=255
  220 IF(NW2.GE.0.AND.NW2.LE.255) GOTO 240
      II=I*4-2
```

tableau D-2 (suite)

```
      WRITE(6,225) NORIG,II,NW2
      NW2=255
  240 IF(NW3.GE.0.AND.NW3.LE.255) GOTO 260
      II=I*4-1
      WRITE(6,225) NORIG,II,NW3
      NW3=255
  260 IF(NW4.GE.0.AND.NW4.LE.255) GOTO 310
      II=I*4
      WRITE(6,225) NORIG,II,NW4
      NW4=255
  310 CONTINUE
      NWORDS(I+2)=NW1*16777216+NW2*65536+NW3*256+NW4
      NTOT=NTOT+NW1+NW2+NW3+NW4
  300 CONTINUE
C
C   ECRIRE LE OU LES RECORDS SUR UNITE DE SORTIE NOUT
C    AVEC LE BIT DE CONTINUATION A 1 SAUF LE DERNIER RECORD
C    S'IL Y A LIEU
C
  400 CONTINUE
      L=NWORDS(1)+1
      LP=2
      IF(L.LE.LRECL) GOTO 420
      LRECLM=LRECL-1
C
C   METTRE LE BIT 0 (BIT LE PLUS A GAUCHE) DU MOT 2 A UN
C   2 ** 31 = 2147483648 = 2147483647 + 1
C
      NNW2=NWORDS(2)+2147483647
      NNW2=NNW2+1
      WRITE(NOUT) LRECLM,NNW2,(NWORDS(I),I=3,LRECL)
      LP=LRECL
  410 IF(L-LP.LE.LRECL-2) GOTO 420
      I1=LP+1
      LP=LP+LRECL-2
      WRITE(NOUT) LRECLM,NNW2,(NWORDS(I),I=I1,LP)
      GOTO 410
  420 I1=LP+1
      LLP=L-LP+1
      WRITE(NOUT) LLP,NWORDS(2),(NWORDS(I),I=I1,L)
      DO 450 I=1,1000
      VECT(I)=0.
  450 CONTINUE
      IF(LAST) GOTO 510
      IF(NORIG.LE.NO) GOTO 470
      WRITE(6,460) ITABN,NTOT
  460 FORMAT('   UTPS MATRIX CREATED  TABLE NUMBER = ',I4/
     +     20X,'MATRIX TOTAL = ',I12)
      NTOT=0
      ITABN=ITABN+1
  470 NORIG=NO
C
  480 CALL SCAN(LIG,VECT,NCEN,VMAX)
      GOTO 50
C
  500 LAST=.TRUE.
      GOTO 490
C
  510 CONTINUE
      WRITE(6,460) ITABN,NTOT
      STOP
      END
      SUBROUTINE SCANO(L,N)
      INTEGER L(72)
      IL=0
      NO=0
   10 IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
      IF(INOS.EQ.-1) GOTO 10
   20 NO=NO*10+INOS
      IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
      IF(INOS.GE.0.AND.INOS.LE.9) GOTO 20
      N=NO
      RETURN
      END
      INTEGER FUNCTION INTC(LC)
      COMMON /NUM/ INUM(13)
      DO 10 I=1,12
      IF(LC.NE.INUM(I)) GOTO 10
      INTC=I-1
      RETURN
   10 CONTINUE
      INTC=-1
      RETURN
      END
      SUBROUTINE SCAN(L,V,NCEN,VMAX)
      REAL V(1)
      INTEGER L(72)
      IL=0
   10 IL=IL+1
      IF(IL.GT.72) RETURN
      IF(INTC(L(IL)).EQ.-1) GOTO 10
   20 IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
      IF(INOS.GE.0.AND.INOS.LE.9) GOTO 20
   30 IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
   35 ND=0
      IF(INOS.EQ.-1) GOTO 30
   40 ND=ND*10+INOS
      IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
      IF(INOS.GE.0.AND.INOS.LE.9) GOTO 40
   50 IL=IL+1
      IF(IL.GT.72) RETURN
      INOS=INTC(L(IL))
      X=0.
      Y=0.
      YY=0.
      IF(INOS.EQ.-1) GOTO 50
      IF(INOS.NE.11) GOTO 58
      V(ND)=VMAX
   55 IL=IL+1
      INOS=INTC(L(IL))
      IF(INOS.EQ.11) GOTO 55
      GOTO 35
   58 IF(INOS.EQ.10) GOTO 70
   60 X=X*10.+FLOAT(INOS)
      IL=IL+1
      IF(IL.GT.72) GOTO 90
      INOS=INTC(L(IL))
      IF(INOS.GE.0.AND.INOS.LE.9) GOTO 60
      IF(INOS.NE.10) GOTO 90
   70 IL=IL+1
      IF(IL.GT.72) GOTO 80
      INOS=INTC(L(IL))
      IF(INOS.EQ.-1) GOTO 80
      Y=Y*10.+FLOAT(INOS)
      YY=YY-1.
      GOTO 70
   80 X=X+(Y*10.**YY)
   90 IF(ND.LE.0.OR.ND.GT.NCEN) GOTO 100
      V(ND)=X
      IF(IL.LT.72) GOTO 30
      RETURN
  100 WRITE(6,110) ND,NCEN,X
  110 FORMAT(/'  ATTENTION !!  DESTINATION ZONE = ',I6,
     +    '  (MAX. ZONE IS ',I6,')  WITH  VALUE = ',F10.3,/,
     +    20X,'IS SKIPPED')
      RETURN
      END
/*
//GO.SYSIN  DD  *
2BYT
 699
  10
0123456789.*M
/*
//GO.FT08F001  DD  DSN=A017.P0108.EMPCH141,DISP=SHR
//GO.FT09F001  DD  DSN=A017.P0108.MAT.EMMEUTPS.ISO,UNIT=3380,
//  VOL=SER=MTQ302,SPACE=(TRK,(10,3),RLSE),DISP=(,CATLG),
//  DCB=(RECFM=VBS,LRECL=1604,BLKSIZE=1608)
```

D.3.3 Traçage des isochrones

Le fichier créé plus haut, peu importe la méthode, peut être maintenant passé aux programmes de traçage d'isochrones.

La documentation de ces programmes peut être obtenue sur demande au service des Systèmes d'information en transport des personnes.