



BD/MSPS 

Guide d'utilisation des outils

Le présent guide décrit un certain nombre d'utilitaires qui sont fournis avec le BD/MSPS, mais qui n'en font pas directement partie.



Statistics
Canada

Statistique
Canada

Canada

Table des matières

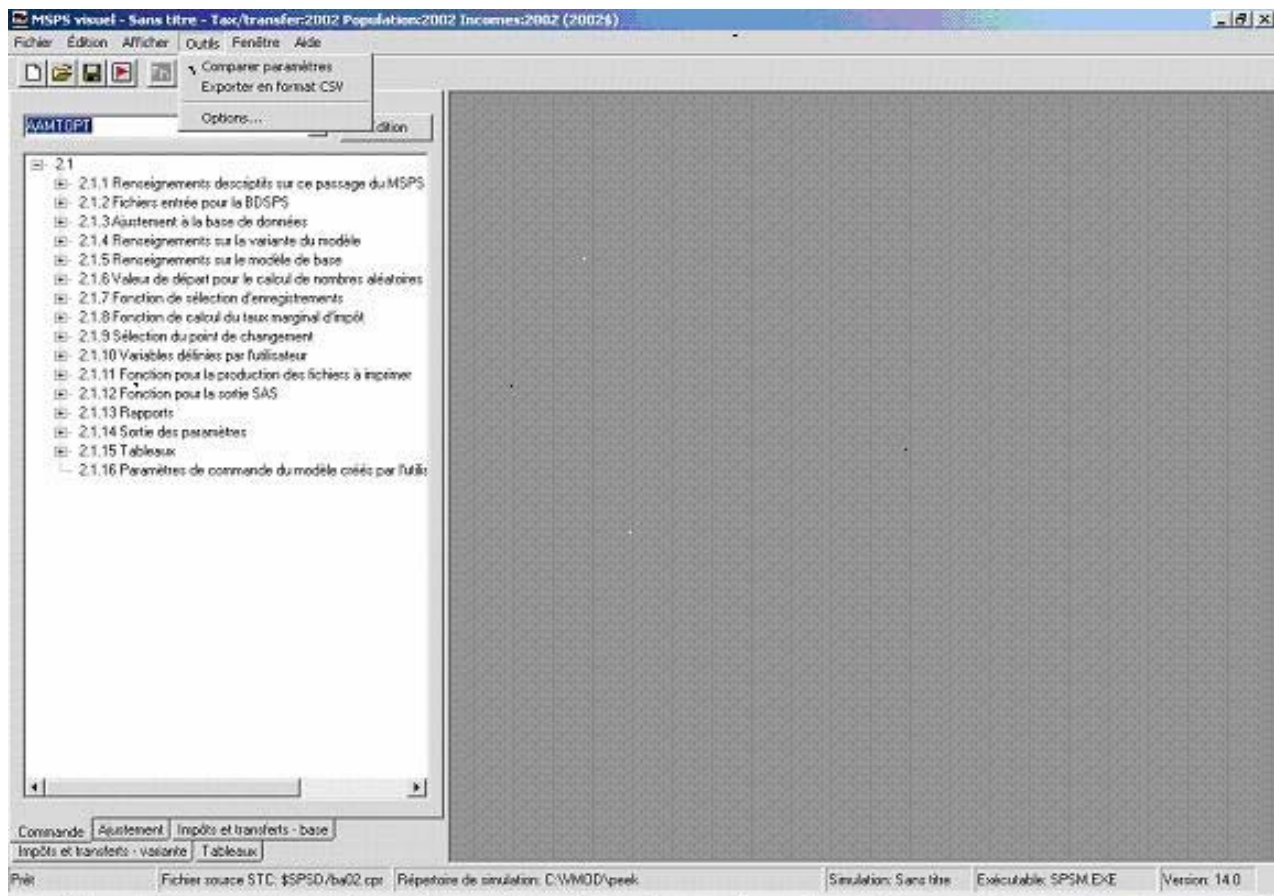
Introduction.....	1
Utilitaires du BD/MSPS.....	3
Exportation à un chiffrier (import.exe).....	3
MSPS Visuel.....	3
MSPS Classique.....	4
spsmiter:Fonction de poursuite du but du MSPS.....	5
Syntaxe de la fonction d'itération.....	6
compparm: Fonction de comparaison de fichiers de paramètres.....	9
MSPS Visuel.....	10
MSPS Classique.....	11
Utilisation de compparm pour faire la mise à jour des fichiers de paramètres.....	13
spsdinfo: Fonction d'en-tête de fichier BD/MSPS.....	15
pupdate : Fonction de mise à jour des paramètres du MSPS.....	16
Facteurs de mise à jour actuellement utilisés dans les fichiers mpr.....	18
Facteurs de mise à jour actuellement utilisés dans les fichiers apr.....	19
Syntaxe.....	19
Exemples :.....	19
bldspd: Fonction de création de base de données .spd.....	21
bldfxv: Fonction de création de base de données .fmx.....	22
bldwgt: Fonction de création de base de données .wgt.....	23
Fonction de création de bases de données bldmrs: .mrs.....	24
Exemple: Créer une base de données de sous-ensemble.....	26
Utilitaires de programmation.....	28
grep.....	28
sumskip.....	28

Introduction

Le présent guide décrit un certain nombre d'outils autonomes qui ne font pas directement partie du programme MSPS. Ils ont été conçus de façon à prolonger de diverses façons les capacités du MSPS. Tous les outils exécutés dans le MSPS Classique et plusieurs peuvent aussi être exécuté en mode MSPS Visuel.

- La section des utilitaires de la BD/MSPS fournit une description des utilitaires disponibles. Par exemple, on y trouve la description de l'utilitaire `compparm`, qui produit un rapport donnant toutes les différences qu'il y a entre deux fichiers de paramètres. Comment utiliser les outils dans le MSPS Visuel ainsi que le MSPS Classique est discuté (lorsque applicable).
- La section de la programmation des utilitaires décrit les utilitaires qui ont utilisés principalement en mode "boîte de verre", bien que l'utilitaire `grep` trouve son application de façon plus générale.

On peut accéder aux outils en utilisant l'interface du MSPS Visuel simplement en sélectionnant un titre sous **Outils** dans la barre menu.



En mode MSPS Classique, il suffit de taper le nom des outils pour les invoquer, avec les arguments nécessaires, sur la ligne de commande. Ainsi, pour afficher toutes les différences qu'il y a entre les deux fichiers de paramètres `ba97.mpr` et `ba98.mpr`, l'utilisateur taperait l'instruction suivante :

```
compparm ba97.mpr ba98.mpr
```

Les outils se trouvent dans le répertoire `\spsm\win32`. Si vous accédez aux outils à l'aide de l'invite du MSPS Classique, le bon chemin sera déjà dans votre environnement et vous pourrez simplement taper le nom de l'outil à l'invite.

En mode MSPS Classique, lorsque la notation est utilisée, la syntaxe d'outils particuliers suivante est utilisée :

- `compparm` les termes utilisés en début, en police Courrier, représentent l'instruction utilisée pour lancer l'outil.
- `[-t]` les crochets indiquent une option qui peut être utilisée ou non.
- `[- | !]` une ou plusieurs barres verticales séparant des caractères (ou des chaînes de caractères) à l'intérieur de crochets d'option indiquent que cette option permet plusieurs valeurs. La barre verticale fonctionne comme séparateur seulement et il ne faut pas l'insérer

dans l'instruction. Habituellement, les caractères fonctionnent comme drapeaux désignant que l'option est activée ou désactivée, bien que certaines options aient des valeurs multiples.

- *file1* les termes inscrits en italiques représentent habituellement des arguments obligatoires, qui doivent être présents dans l'instruction utilisée pour lancer l'outil (p. ex. les noms de fichiers d'entrées et de sorties). L'exception à cette règle est l'argument représentant une valeur associée à une option que l'utilisateur doit entrer.

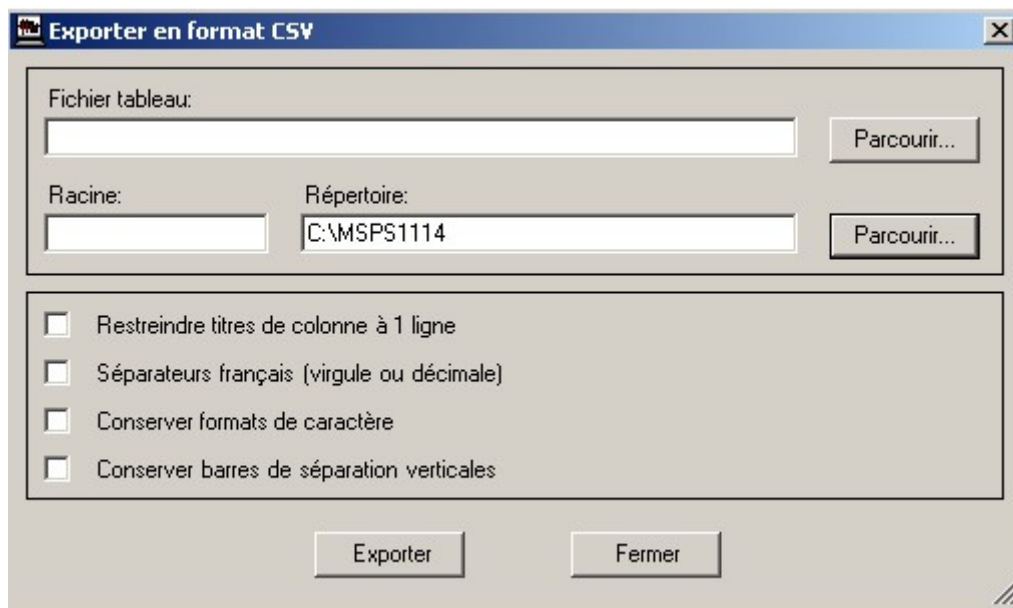
Utilitaires du BD/MSPS

EXPORTATION À UN CHIFFRIER (IMPORT.EXE)

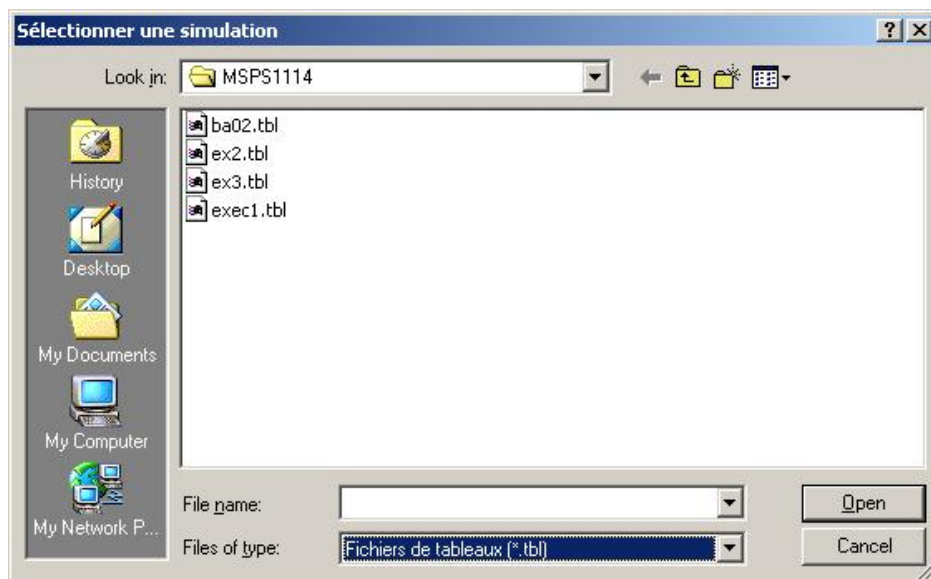
Cet utilitaire convertit chaque table dans le format de fichier exigé par un progiciel de chiffrier, comme Excel.

MSPS Visuel

Cet outil se trouve sous **Outils, Exporter en format CSV**. L'interface suivante apparaît :



Premièrement, sélectionnez un fichier tableau à exporter. Les fichiers tableaux comportent l'extension *.tbl*. Si une simulation est actuellement ouverte, le nom de fichier *.tbl* apparaît sur la ligne Fichier tableau et le nom de sa racine (le préfixe qui sera donné à chaque tableau *.csv*) apparaît dans la case Racine. Sélectionnez **Parcourir** pour choisir un fichier *.tbl* à exporter. Un dossier nommé \CSV est généré automatiquement dans le répertoire de simulation lorsque vous sélectionnez **Exporter**.



Autres options :

- Restreindre titres de colonne à 1 ligne : les en-têtes de colonnes sont tronqués et ne s'étendent que sur une rangée du tableau
- Séparateurs français (virgule au lieu de la décimale) : la virgule française sert de décimale séparatrice dans la conversion des tableaux en format csv au lieu du point
- Conserver formats de caractère : les champs numériques doivent demeurer dans la forme des caractères formatés et non être convertis en chiffres
- Conserver barres de séparation verticale : les symboles de barre verticale (|) sont conservés

MSPS Classique

En mode Classique, ceci est l'utilitaire `import`. Il est lancé de la façon suivante :

```
import [-b] [-t] [-f] [-s] infile stemname
```

où :

- `-t` est un drapeau qui indique que les champs numériques doivent être laissés dans la forme de caractères structurés, et non être convertis en nombres.
- `-b` est un drapeau qui indique que les symboles de barre verticale (|) seront conservés aussi.
- `-f` est un drapeau qui indique que la virgule est utilisée comme séparateur des décimales dans la conversion des tableaux en format csv au lieu du point anglais.
- `-s` est un drapeau qui indique que les en-têtes de colonnes seront tronqués de façon à occuper seulement un rang du tableau.
- `infile` est le nom du fichier contenant les tables MSPS (fichier .tbl).

- `stemname` est une chaîne qui est utilisée pour générer les noms de fichier à la sortie. Les fichiers produits ont la forme `stemname_tableid.prn`.
- `tableid` identifie la table extraite. Elle contient la chaîne alphanumérique qui suit une occurrence de la chaîne “Table” dans `infile`.

Dans le cas des tables à trois dimensions et plus, un sous-identificateur numérique est ajouté à `tableid`. Supposons, par exemple, que le fichier `run1.tbl` existe dans le répertoire courant; si l’instruction :

```
import run1.tbl tab
```

est donnée, et si `run1.tbl` contient les tables intégrées 0, 2 et 2A, ainsi que les tables définies par l’utilisateur 1U et 2U, alors `import` génère les fichiers suivants :

```
tab_0.csv
tab_2.csv
tab_2A.csv
tab_1U.csv
tab_2U1.csv
tab_2U2.csv
tab_t.csv
```

Le fichier `tab_t.csv` contient les lignes d’en-tête de page qui se trouvent au haut du fichier `run1.tbl`. Ces lignes d’en-tête de page ne sont pas comprises dans aucun des fichiers `.csv`. L’exemple ci-dessus suppose que la table définie par l’utilisateur 2U était tridimensionnelle et contenait deux tranches.

`import` convertit habituellement les entrées de tables numériques en entrées de chiffriers numériques. Ceci permet que l’on fasse le traitement des nombres à l’intérieur du chiffrier après l’importation. Le drapeau facultatif `-t` indique que les champs numériques doivent être laissés dans leur forme de caractères structurés et qu’ils ne doivent pas être convertis en nombres. Si, en outre, le drapeau `-b` est spécifié, les symboles de barre verticale (`|`) seront aussi conservés.

SPSMITER:FONCTION DE POURSUITE DU BUT DU MSPS

La fonction de poursuite du but du MSPS fournit un moyen permettant de modifier la valeur, en dollars, de un ou de plusieurs paramètres spécifiés, par un processus itératif visant à produire une valeur désirée pour une variable en particulier dans la fonction d’impôt/de transfert. La fonction permet maintenant de sélectionner des paramètres tant scalaires que de tableau pour la poursuite du but. Les résultats de la fonction de poursuite du but peuvent alors être enregistrés dans le fichier des résultats d’origine du MSPS en y ajustant les paramètres pertinents.

Cet outil est utile quand on désire obtenir des réponses à des questions du genre “et si”. Par exemple, quel serait le taux de surtaxe fédérale nécessaire pour l’obtention de la neutralité du revenu, si la taxe fédérale avait été éliminée sur les produits et services en 2001 ? Il serait possible de répondre à cette question en exécutant d’abord un cas avec la base de 2001 afin d’observer le niveau des recettes tirées de l’impôt fédéral quand la TPS est perçue, puis en créant un cas de variante qui met à zéro la TPS.

L'analyste pourrait alors créer une table en utilisant la fonction Tableaux croisés (voir les documents *Guide d'utilisation*, et *Guide d'utilisation des tableaux croisés*, pour obtenir une description de la fonction Tableaux croisés) qui donne l'écart entre les revenus obtenus de l'impôt fédéral dans ces deux cas (la perte de revenus de l'impôt fédéral du fait de l'élimination de la taxe). La fonction de poursuite du but pourrait alors être utilisée pour estimer le niveau de la surtaxe fédérale qui est nécessaire à la réduction à zéro de l'écart entre l'exécution de base et l'exécution de variante, ce qui permettrait d'obtenir la neutralité de revenu.

L'utilisateur peut spécifier soit la méthode de Newton, soit une recherche binaire, comme méthode d'itération. Les deux méthodes exigent que l'utilisateur fournisse une estimation initiale du paramètre scalaire à ajuster. En outre, l'option de recherche binaire exige des limites supérieure et inférieure du paramètre à ajuster. Si elle est suffisamment proche de la solution, la méthode convergera, autrement elle fera de nouvelles suppositions. La section suivante donne les détails dont on a besoin pour accéder à la fonction `spsmiter` et l'utiliser.

Syntaxe de la fonction d'itération

Toutes les commandes nécessaires à l'exécution de la fonction `spsmiter` peuvent être entrées en mode MSPS Classique soit sur la ligne de commande directement ou encore à partir d'un fichier de traitement par lots (le document *Comment exécuter le MSPS* décrit la fonction de traitement par lots du MSPS). Pour connaître la syntaxe de l'instruction, taper `spsmiter`. L'explication suivante de la fonction `spsmiter` s'affiche à l'écran :

```
usage: spsmiter [-t] [-a] [-b<inférieure/supérieure>] échant modèle cprfile tableid
ligne col cible tolres parm.
-t          - imprimer seulement le sommaire d'action (optionnel)
-a          - modifier les paramètres dans le fichier .apr au lieu de .mpr
-b          - utiliser une recherche binaire au lieu de la méthode de Newton
échant     - taille de(s) l'échantillon(s) (ex. 0.05/1.00 )
modèle     - nom du modèle (habituellement spsm)
cprfile    - nom du fichier .cpr à itérer pour la convergence
tableid    - numéro d'identification du tableau (Ex. 2)
ligne      - numéro de la ligne dans le tableau (incluant l'entête)
col        - numéro de la colonne dans le tableau (incluant l'entête)
cible      - valeur cible
tolres     - tolérance pour la convergence
parm       - nom du paramètre à modifier pour obtenir la convergence
(paramètres supplémentaires)
Exemple: spsmiter 1.00 spsm.exe iter1.cpr 0 36 2 40000 10 BOAS
```

Description des options de `spsmiter` :

`[-t]:`

Lorsque l'expression `-t` est insérée entre l'instruction `spsmiter` et l'expression de l'échantillon dans l'instruction de traitement par lots, seul le sommaire de l'action s'affichera à l'écran. Il faudrait utiliser cette option pour vérifier si l'instruction `spsmiter` a référé aux tables et aux valeurs des tables que l'analyste désirait. Les analystes devraient noter que, si un fichier des résultats de variante n'a pas été créé, le paramètre de commande `INPVARMPR` contiendra les paramètres d'impôt/de transfert du modèle MSPS original et, par conséquent, toute itération effectuée peut endommager ce fichier. La vérification du sommaire de l'action avant l'exécution de l'itération devrait empêcher cette erreur. Ci-dessous vous avez un exemple du Sommaire de l'action qui montre les

phrases produites pendant l'exécution de l'Exemple pratique qui suit la description des options de spsmiter.

SOMMAIRE DE L'ACTION :

Le programme `spsm.exe` sera exécuté avec le fichier de paramètre de commande `test.cpr`. Le paramètre `FSURR1` (sans autre paramètre) dans le fichier `test.mpr` sera modifié jusqu'à ce que l'entrée du rang 24 et la colonne 2 de la table 0 dans le fichier `test.mpr` atteigne la valeur désirée de 511479,8. La valeur actuelle de cette entrée est 534702.

[-a] :

Lorsque l'expression `-a` est insérée entre l'instruction `spsmiter` et l'expression d'échantillon de l'instruction de traitement par lots, la fonction modifie tout paramètre d'ajustement qui se trouve dans le fichier `.apr` par rapport à un paramètre de modèle qui se trouve dans le fichier `.mpr`. Ceci donne à l'utilisateur la possibilité d'ajuster certains paramètres de croissance afin de produire une variable de modèle désirée. Ainsi, on peut ajuster le facteur de croissance qui contrôle le revenu agricole en Saskatchewan de façon à forcer la variable modélisée de revenu agricole à se conformer à un total de contrôle.

[-b<inférieure/supérieure>] :

Lorsque l'expression `-b` est insérée entre l'instruction `spsmiter` et l'expression d'échantillon de l'instruction de traitement par lots, la fonction utilise la méthode d'itération par recherche binaire plutôt que la méthode de Newton. Cette méthode est plus efficace lorsque la relation entre le paramètre à ajuster et la variable cible n'est pas uniforme et continue. Il faut noter que les limites supérieures et inférieures du paramètre doivent être spécifiées avec la variable cible, comme `-b0.5/1.3`, où 0,5 est la limite inférieure et 1,3 la limite supérieure.

échant :

L'instruction `échant` permet à l'utilisateur de sélectionner la taille de l'échantillon pour la deuxième itération, phase 2, entre 0 % et 100 % de la base de données. La phase 1 exécutera l'itération en utilisant 10 % d'un fichier 100 % de la base de données `BDSPS`. Cependant, si l'itération réfère à la sortie `MSPS` qui a été exécutée avec l'échantillon de +5 % de la base de données, l'itération de phase 1 sera exécutée avec l'échantillon de 5 % et non celle de 10 % (le message 10 % sera quand même affiché à l'écran). L'utilisateur devrait noter qu'une méthode d'itération exécutée sur 100 % de la base de données peut exiger beaucoup de temps et il est donc utile d'exécuter l'itération en utilisant juste la phase 1 en fixant à zéro l'instruction `échant`, en vérifiant le résultat, puis en exécutant l'itération encore une fois avec l'instruction `échant` à la valeur 1.

Si seulement un exemple est donné alors, la Phase 1 est automatiquement exécuté avec la taille de l'échantillon donné et la Phase 2 à 100 %. Il est possible de sauter la phase I en fixant l'échantillon de la phase 1 à 0 % et la phase 2 à 100 % : `0.0/1.0`.

modèle :

L'instruction modèle devrait référer seulement au fichier `spsm.exe`, à moins que l'utilisateur n'ait créé une autre version du modèle avec la méthode de traitement en mode "boîte de verre".

`cprfile` :

L'instruction `cprfile` devrait référer au fichier `.cpr` qui a produit la table contenant la variable sélectionnée pour la poursuite du but.

`tableid`:

L'instruction `tableid` identifie le tableau à utiliser selon son numéro d'identification. Dans l'exemple donné ci-dessous, la table 0 de `test.cpr` a été sélectionnée. Les tables intégrées et les tables générées par la fonction Tableaux croisés (voir le document *Guide d'utilisation* pour obtenir une description de la fonction Tableaux croisés) peuvent toutes être sélectionnées.

`ligne` :

L'instruction `ligne` indique le rang de la table spécifiée qui contient la variable sélectionnée pour la poursuite du but.

`col` :

L'argument `col` identifie la colonne de la table spécifiée qui contient la variable sélectionnée pour la poursuite du but. Les numéros de rang et de colonne identifient la cellule de la table qui contient la variable sélectionnée pour la poursuite du but. La variable peut être soit une variable intégrée, soit une variable définie par l'utilisateur (voir le document *Guide d'utilisation* pour obtenir une description de la fonction Variable définie par l'utilisateur). Dans l'exemple ci-dessous, la variable Taxe fédérale se trouve au rang 24, colonne 2, de la table 0.

`cible` :

La valeur cible est le montant désiré, en dollars, de la variable que l'analyste a désignée par les coordonnées de rang et de colonne. Dans l'exemple ci-dessous, la valeur cible choisie par l'analyste comme d'impôt fédéral total est 146,194 \$.

`tolres` :

La tolérance de la convergence devrait être fixée à une certaine valeur; dans notre exemple, il s'agit de 10 millions (10), et, par conséquent, un résultat d'itération se situant à moins de 10 millions de dollars de la valeur cible sera sélectionné et la fonction `spsmter` s'arrêtera. L'utilisateur devrait noter que, plus la tolérance est faible en vue de la convergence, plus le temps de traitement de la fonction sera grand.

`parm` :

Il est possible de sélectionner **jusqu'à 50 paramètres d'impôt et de transfert** qui peuvent être modifiés en vue de la convergence. Pour le traitement de paramètres qui contiennent un vecteur ou une matrice de valeurs, il faut donner le nom du paramètre avec la cellule que vous désirez modifier. Par exemple, si vous désirez modifier le taux marginal d'imposition pour le groupe à plus faible revenu, le paramètre devrait être écrit comme `FTX[0][2]`, pour le premier rang et la troisième colonne. L'utilisateur peut déterminer quels paramètres sont scalaires en consultant le document *Guide des paramètres* ou en observant la structure des paramètres dans le fichier `.mpr`. Dans l'exemple pratique de la section suivante, le paramètre `FSURR1`, taux de surtaxe fédérale 1, a été sélectionné pour modification.

Exemple pratique

L'exécution de l'instruction suivante en mode de traitement par lots

```
D:\SPSMTEST> spsmitter 1,0 spsm.exe test.cpr 0 25 2 146194 10 FSURR1
```

a donné les résultats suivants :

SOMMAIRE DE L'ACTION :

Le programme `spsm.exe` sera exécuté avec le fichier de paramètres de commande `test.cpr`. Le paramètre `FSURR1` (avec d'autres 0) du fichier `test.mpr` sera modifié jusqu'à ce que la valeur contenue au rang 25, colonne 2, de la table 0 du fichier `test.tbl` atteigne la valeur désirée de 146194. La valeur actuelle de cette entrée est de 121338, correspondant à la valeur 0 du paramètre.

Phase 1: Échantillon à 10%

Itération=0 (10%) `FSURR1=0` Résultat=117526 Erreur=-28668,9

Itération=1 (10%) `FSURR1=1` Résultat=200494 Erreur=54299,2

Itération=2 (10%) `FSURR1=0,345541` Résultat=146191 Erreur=3,1

Sortie - Tolérance désirée obtenue

Phase 2: 100% des exécutions

Itération=0 (100%) `FSURR1=0,345541` Résultat=151485 Erreur=5290,8

Itération=1 (100%) `FSURR1=0,281776` Résultat=145921 Erreur=-273

Itération=2 (100%) `FSURR1=0,284905` Résultat=146194 Erreur=0

Sortie - Tolérance désirée obtenue

Nous pouvons alors répondre à la question posée ci-dessus : "Quel serait le taux de surtaxe fédérale nécessaire pour obtenir la neutralité du revenu, si la TPS avait été éliminée en 2001 ?" La fonction de poursuite du but a estimé que le taux de surtaxe fédérale augmenterait, passant de 0 % à 28 %, pour que la neutralité du revenu soit conservée.

L'analyste a choisi de consigner les résultats de la poursuite du but dans le fichier `test.mpr`, ce qui change les valeurs des variables pertinentes dans les tables intégrées et dans les tableaux croisés créés par l'analyste, pour révéler l'effet de la nouvelle surtaxe.

COMPPARM: FONCTION DE COMPARAISON DE FICHIERS DE PARAMÈTRES

L'utilitaire `compparm` compare deux fichiers de paramètres et produit un rapport qui donne les écarts en détail. Un certain nombre d'options permettent de déterminer le fonctionnement de `compparm`. Ces options ont toutes des valeurs par défaut qui conviennent pour l'utilisation

directe avec les fichiers de paramètres complets, ou avec des parties de fichiers de paramètres (c.-à-d., les fichiers d'inclusion de paramètres).

En outre, compparm comporte une option qui peut être utilisée pour la création d'un fichier d'inclusion de paramètres. Les différences qu'il y a entre deux fichiers de paramètres se retrouvent dans un fichier d'inclusion qui peut être lu par le MSPS. Cette option est très utile lorsque l'on fait la mise à jour des fichiers de paramètres pour exécuter une nouvelle version du MSPS ou une version boîte de verre (voir la section ci-dessous portant sur la mise à jour des fichiers de paramètres).

MSPS Visuel

Cet outil se trouve sous **Outils, Comparer paramètres**. La boîte suivante apparaît :

Comparer paramètres (compparm)

Fichier 1:
C:\spsd\ba10.mpr Parcourir...

Fichier 2:
 Parcourir...

Largeur maximale des lignes de sortie: 80

Utiliser des lignes de séparateur: ☒

Comparer les paramètres communs aux deux fichiers ☒

Inclure les valeurs des paramètres ne figurant que dans le fichier 1 ☐

Inclure les valeurs des paramètres ne figurant que dans le fichier 2 ☐

Créer fichier à inclure: ☒

Nom du fichier: Parcourir...

Exécuter Fermer

Le Fichier 1 et le Fichier 2 sont des arguments obligatoires, puisqu'il s'agit du nom des fichiers de paramètres existants de la BD/MSPS, soit les noms des paramètres et les valeurs numériques ou textuelles connexes.

Il n'est pas nécessaire que les fichiers soient des fichiers de paramètres complets et les paramètres peuvent apparaître dans n'importe quel ordre dans les fichiers d'entrée. Le rapport sera ordonné de la même manière que le Fichier 1. Il dresse la liste des valeurs du Fichier 1 dans une colonne et celles du Fichier 2 dans la colonne adjacente.

Le paramètre par défaut compare les paramètres communs aux deux fichiers. Un argument facultatif permet de changer ce paramètre (voir ci-dessous).

Il existe quelques arguments facultatifs dans `compparm` :

- La Largeur maximale des lignes de sortie précise le nombre total de positions d'impression que le rapport occupera. La valeur par défaut est 80 et convient à l'affichage à l'écran. Les valeurs des paramètres dont la longueur est de plus de 2 seront tronquées. La troncature est indiquée au moyen d'un caractère « + » à la droite de la valeur du paramètre.
- Utiliser des séparateurs précise si l'utilisateur souhaite insérer des lignes de séparation horizontales et des commentaires dans le rapport. La valeur par défaut est un rapport accompagné de commentaires et formaté en entier.
- L'utilisateur peut également demander la fonction `compparm` pour créer un fichier d'inclusion des paramètres (`.apd`, `.cpd`, `.mpd`) contenant les différences entre les deux fichiers de paramètres. S'il sélectionne cette option, il doit parcourir à un répertoire existant et préciser le nom du fichier avec l'extension appropriée.
- Tel que susmentionné, le paramètre par défaut permet de comparer les paramètres communs aux deux fichiers. L'utilisateur peut choisir que les seuls paramètres qui figurent dans le Fichier 1 soient ou non des données de sortie. Une autre option précise si l'utilisateur souhaite que les paramètres figurant uniquement dans le Fichier 2 soient des données de sortie. Ces trois options peuvent être combinées.

MSPS Classique

Voici la syntaxe complète de l'instruction permettant de lancer `compparm` :

```
compparm [!|-]i [-w width] [-|!]sep [[!|-][12|1|2]]... file1 file2
```

où :

[!|-]i

Cet argument facultatif précise que la fonction `compparm` créera un fichier d'inclusion de paramètres (par exemple ceux avec le suffixe `c.pi`, `.api` ou `.mpi`) contenant les différences qu'il y a entre deux fichiers de paramètres. La valeur par défaut est `!`; elle fait en sorte que l'on obtient un rapport de `compparm` plutôt qu'un fichier d'inclusion. Le rapport `compparm` standard donne la liste des valeurs pour `file1` dans une colonne et les valeurs pour `file2` dans une colonne adjacente.

-w width

Cet argument facultatif précise le nombre total de positions d'impression qu'occupera le rapport résultant. La valeur par défaut est 80, ce qui convient pour l'affichage à l'écran. Toute valeur de paramètre plus longue que `width/2` est tronquée. Quand une valeur de paramètre est tronquée, elle est suivie d'un caractère `+`.

[-|!]sep

Cet argument facultatif détermine si l'utilisateur désire que les lignes de séparation horizontales et les commentaires soient inclus (-) ou non (!) dans le rapport. La valeur

par défaut est -; elle fait en sorte qu'on obtient un rapport entièrement mis en forme et commenté. L'utilisateur peut désactiver l'option `sep` pour créer un rapport qui est plus facile à convertir en fichiers d'inclusion de paramètres.

[!|-]12

Cet argument facultatif précise si l'utilisateur désire ou non que les valeurs des paramètres communs aux deux fichiers soient comparées. La valeur par défaut fait en sorte que ces valeurs sont comparées.

[!|-]1

Cet argument facultatif précise si l'utilisateur désire ou non que les paramètres trouvés seulement dans le premier fichier spécifié (file1) soient produits à la sortie. La valeur par défaut fait en sorte que les paramètres ne soient pas produits à la sortie dans le rapport.

[!|-]2

Cet argument facultatif précise si l'utilisateur désire ou non que les paramètres trouvés seulement dans le deuxième fichier spécifié (file2) soient produits à la sortie. La valeur par défaut fait en sorte que les paramètres ne soient pas produits à la sortie dans le rapport.

file1, file2

Ces deux arguments sont requis, puisqu'ils sont les noms des fichiers de paramètres BD/MSPS existants qui comportent les noms de paramètres ainsi que la valeur numérique ou de texte qui leur sont associées.

Les fichiers d'entrées n'ont pas à être des fichiers de paramètres complets et les paramètres peuvent être dans n'importe quel ordre à l'intérieur du fichier d'entrées. Le rapport aura le même ordre que celui de file1.

Si l'utilisateur entre l'instruction

```
compparm \spsd\squ08.mpr \spsd\ba08.mpr
```

le rapport résultant sera affiché à l'écran de la façon suivante (seule une partie du rapport est donnée ici) :

```

=====
                        RAPPORT DES DIFFÉRENCES DU FICHIER DES PARAMÈTRES
=====
                        \spsd\squ08.mpr          |          \spsd\ba08.mpr
Mon Oct 17 12:44:14 2005          | Mon Oct 17 12:41:36 2005
=====
###
## 2.3.1 Description des fichiers de paramètre
###
=====
                        Description des paramètres fiscaux/de transfert
MPRDESC      Valeurs courantes pour 2008          MPRDESC      Valeurs courantes
pour 2008
=====
###
## 2.3.2.6 Crédit pour taxe fédérale sur les ventes
###
=====
```

Montant du crédit pour taxe fédérale sur les ventes pour le			
déclarant			
FSTCF	50.00	FSTCF	70.00

Montant du crédit pour taxe fédérale sur les ventes pour le conjoint			
FSTCS	50.00	FSTCS	70.00

Montant du crédit pour taxe fédérale sur les ventes pour une personne			
à charge			
FSTCC	25.00	FSTCC	35.00

Niveau de réduction du crédit pour taxe fédérale sur les ventes			
FSTCL	15000.00	FSTCL	16000.00
=====			

Le produit de sortie de `compparm` peut être réacheminé vers un fichier ou vers une imprimante. Dans le dernier cas, il peut être souhaitable d'augmenter la largeur du rapport afin d'éviter que les valeurs des paramètres soient tronquées. Pour ce faire, il suffit de taper l'instruction suivante :

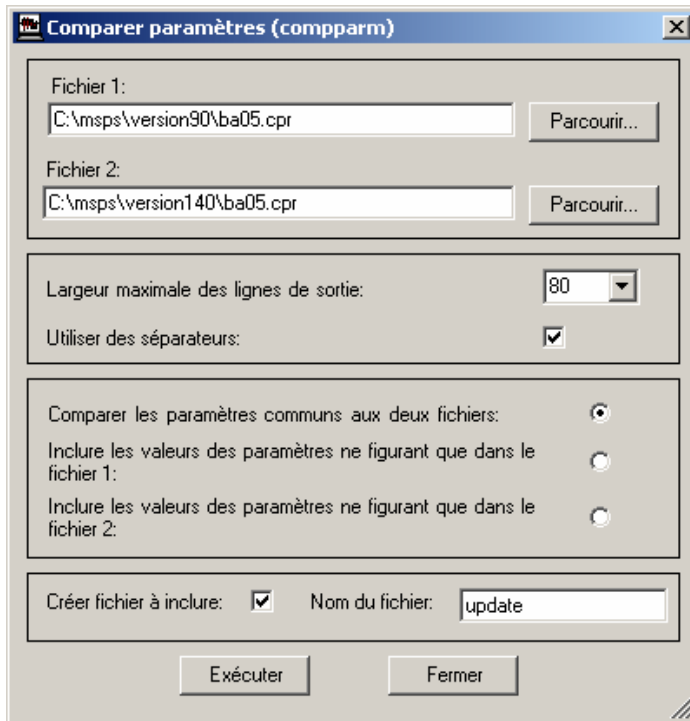
```
C>compparm -wl32 sq08.mpr ba08.mpr >prn
```

Utilisation de `compparm` pour faire la mise à jour des fichiers de paramètres

L'exemple suivant fait la mise à jour d'un ancien fichier de paramètres (c.-à-d. la version 9.0) en apportant les modifications contenues dans le nouveau fichier de paramètres (c.-à-d. la version 14.0); ce n'est qu'une illustration de l'utilisation de `compparm`. **Il est fortement suggéré de toujours procéder à la mise en œuvre complète du BD/MSPS, car de nouveaux paramètres sont associés aux changements apportés au programme ou encore aux nouveaux programmes.**

MSPS Visuel

L'interface suivante illustre comment mettre à jour le fichier des paramètres de commande `\version90\ba05.cpr` (version 9.0) pour tenir compte des changements qui apparaissent dans le fichier de paramètre `\version140\ba05.cpr` (version 14.0).



Un fichier d'inclusion nommé par l'utilisateur, mis à jour, sera inscrit dans le répertoire de la simulation courante et appelé `update.cpd`. Pendant la simulation suivante du MSPS, l'analyste pourra lire le fichier d'inclusion des paramètres de commande pour mettre à jour l'ancien fichier des paramètres de commande. Le fichier d'inclusion créé à partir de cette commande apparaît automatiquement dans le MSPS Visuel et ressemble à celui créé à l'aide du MSPS Classique (voir ci-dessous). L'utilisateur peut également imprimer le fichier d'inclusion à partir du MSPS Visuel.

MSPS Classique

Lorsqu'il utilise la fonction `compparm`, l'analyste est avisé de maintenir les paramètres par défaut des options 1, 2 et 12 pour vérifier que les fichiers sont entièrement mis à jour. Par exemple, pour mettre à jour le fichier des paramètres de commande `\spsd3\ba05.cpr` (version 9.0) afin de tenir compte des changements qui apparaissent dans le fichier de paramètres `\spsd\ba05.cpr` (version 14.0), l'analyste entrerait la commande suivante :

```
C:\SPSMCOMP>compparm -i \spsd3\ba05.cpr \spsd\ba05.cpr > update.cpi
```

Le fichier d'inclusion résultant, `update.cpi`, serait enregistré dans le répertoire SPSMCOMP. Pendant l'exécution du MSPS, l'analyste peut faire la lecture de ce fichier d'inclusion de paramètres de commande pour faire la mise à jour de l'ancien fichier `.cpr`. Le fichier d'inclusion créé par cette instruction `compparm` ressemble à ce qui suit (il n'y a qu'une partie du fichier d'inclusion illustrée ci-dessous) :

```
#####
## Include file to convert from '\spsd3\ba05.cpr' to '\spsd\ba05.cpr' ##
#####
### ## 2.1.1 Descriptive information on this SPSM run ###
ALGDESC                                # Names of standard and alternate algorithms
```

Algorithm	Standard	Alternate
drv	\$Revision: 1.102 \$ Sep 12/05	None/aucun Jun 17/05
ui	\$Revision: 1.102 \$ Nov 17/05	None Jun 17/05
famod	\$Revision: 1.102 \$ Jun 17/05	Untitled Jun 17/05
oas	\$Revision: 1.102 \$ Jun 17/05	None Jun 17/05

```

### ## 2.1.2 SPSPD input files ###
INSPSPD      /spsd/v140y02.spd      # Name of SPSPD file (in)
### ## 2.1.3 Database adjustment ###
INPAPR       /spsd/ba02_05.apr      # Name of database adjustment parameter file

### ## 2.1.4 Variant Information ###
VARALG       Version 14.0: 88-09    # Name of variant algorithm
#####
## Parameters Added: ##
#####
CPRVER       14.0                  # Édition de la version de la BD/MSPS
CPRSFILE     $SPSPD/ba05.cpr       # Fichier de départ de paramètre de commande
LICOOPT      1                    # Définition T4 du SFR 1=Avant impôt 2=Après
impôt

```

L'analyste devrait noter que tout paramètre créé en mode boîte de verre est contenu dans file1 et est commenté dans la section des Paramètres supprimés du fichier d'inclusion. Pour activer ces paramètres dans le fichier de paramètres mis à jour, il suffit de supprimer le caractère de balisage des commentaires qui précèdent le paramètre (#).

SPSPDINFO: FONCTION D'EN-TÊTE DE FICHIER BD/MSPS

Les fichiers de base de données BDSPPS et les fichiers de résultats binaires du MSPS ont une en-tête qui contient de l'information sur la date d'établissement du fichier, le numéro de version et un certain nombre d'autres éléments utiles. La fonction `spsdinfo` affiche cette information sous forme imprimable. Par exemple, si l'utilisateur donne l'instruction :

```
C>spsdinfo \spsd\v140y02.spd
```

le rapport suivant s'imprimera au terminal de l'utilisateur :

```

File          : \spsd\v140y02.spd
Type          : spd
Database ver. : 14.00
Binary ver.   : 14.00
Date          : Mon Oct 17 14:45:13 2005
Label         : Version 140
Licensee      : Internal StatsCan use only
Households    : 80365
Numbers       : 0
(unused)      : 0
(unused)      : 0

```

Cet outil produit de l'information supplémentaire sur toute variable utilisateur contenue dans les fichiers .mrs. Plus précisément, toute étiquette de variable ou toute étiquette de niveau de classification créée dans UVAR s'affiche si la variable qui lui est associée a été spécifiée dans OUTMRSVARS.

L'exemple suivant illustre cette fonction :

```
File          : ba08.mrs
```



```

Type           : mrs
Database ver.  : 4.00
Binary ver.    : 4.00
Date           : Tue Nov 27 10:19:05 1994
Label          : Current values for 08
Licensee       : Statistics Canada
Households     : 2764
Numbers        : 23283
sizeof(MP)     : 11406
#bytes uv      : 190
Variables      : immicons newtax1 newincl
Variable info:
label(newtax1)="newtax1";
label(newincl)="Total income Group";
levels(newincl)=
    "Min-5,000",
    "5,000-10,000",
    "10,000-50,000",
    "50,000-100,000",
    "100,000-MAX";

```

PUPDATE : FONCTION DE MISE À JOUR DES PARAMÈTRES DU MSPS

La fonction **pupdate** sert à faire croître les paramètres des fichiers `.mpr` (paramètres du modèle d'impôt/de transfert) ou `.apr` (paramètres de rajustement de la base de données). Les fichiers inclus partiellement (« `.mpi` », « `.mpd` », « `.api` », « `.apd` ») peuvent également être augmentés.

La fonction `Pupdate` crée un fichier de paramètres (`out=outfile`) en augmentant un fichier de paramètre (`cur=file1`) à l'aide de facteurs de mise à jour donnés (`name=value`). Si un deuxième fichier est donné, (`nxt=file2`), le fichier de sortie n'augmentera que les paramètres augmentés ou manquants dans le deuxième fichier. Par exemple, si dans le deuxième fichier un taux d'imposition est donné sans aucune indication de mise à jour, à savoir qu'un crédit d'impôt a été augmenté à l'aide de l'IPC, le fichier de sortie maintiendra le taux d'imposition hors du deuxième fichier, mais il augmentera le crédit d'impôt à l'aide du facteur de mise à jour pertinent.

Les valeurs relatives aux paramètres de mise à jour figurent à la ligne de commande et elles peuvent être activées sous la forme `name=value` ou elles peuvent être transférées à la fonction `pupdate` dans un fichier.

Syntaxe :

pupdate *cur=file1* [*nxt=file2*] *out=outfile* [*name=value*]...

ou

pupdate -f filename où *filename* contient

```

cur=file1
out=outfile
[nxt=file2]
[name1=value1]
.
.

```

.
[name=value]
[CURNAME=label]

Fichiers de paramètres

cur - fichier de paramètres initial
nxt - deuxième fichier de paramètres (facultatif)
out - nouveau fichier de paramètres

Indications de source et de mise à jour

Les observations suivantes accompagnent chaque paramètre qui peut être augmenté :

Source - origine de la valeur des paramètres.
Update - méthode d'échelonnement ou d'établissement des valeurs

p. ex.

```
GISST      3          GIS take-up rate: single pensioner by
benefit level
      0      0.322      (0.0005)
      701     0.680      (0.0001)
      2909     1.000      (0.0001)
                                # Source: Grown from ba99.mpr using
CPI=1.014
                                # Update: Factor[1]=CPI
```

Source

La source de la valeur des paramètres provient de certaines publications officielles ou de l'information produite par la fonction pupdate.

p. ex. # Source: Redbook, 1995 Edition
ou
Source : Grown from ba99.mpr using CPI=1.014
Source: Copied from ba99.mpr
Source: Given as LABEL=...

Mise à jour

Les paramètres comportant une indication de mise à jour seront augmentés; les autres seront simplement copiés dans le fichier de sortie. L'indication de mise à jour contiendra l'une des écritures suivantes précédées de l'en-tête de mise à jour. # Update :

Valeur=value - Accorde la *valeur* au paramètre ou aux rangs de paramètres.
Facteur=value - Multiplie la *valeur* par les valeurs de colonne dans le(s) ligne(s) pour calculer les valeurs d'augmentation.

Facteur[]=*value* - Multiplie la *valeur* par les valeurs des colonnes choisies dans le(s) ligne(s) pour calculer les valeurs d'augmentation.

p. ex. # Update: Value=LABEL
 # Update: Factor=CPIM3
 # Update: Factor[2,3]=DEFAULT

Fichier Cur

Si seulement un fichier (*cur*) est admis à titre d'argument et qu'il existe une indication et un facteur de mise à jour, le paramètre est augmenté. Si aucune indication de mise à jour n'existe, le paramètre est copié du fichier *cur* au fichier de sortie. S'il existe une indication de mise à jour, mais qu'aucun facteur de mise à jour n'est fourni, un message d'erreur sera produit.

Fichier Nxt

Si un deuxième fichier (*nxt*) est admis et que le paramètre ne figure pas dans *nxt* ou que la source fournie dans *nxt* est 'Grown', 'Copied' ou 'Given', une écriture d'augmentation est effectuée dans le fichier de sortie. Si l'indication de source ne renferme aucune donnée de mise à jour, les données relatives aux paramètres sont simplement copiées du fichier *nxt* au fichier de sortie.

Arguments

cur - fichier de l'année précédente
nxt - deuxième fichier facultatif
out - fichier de sortie créé
CURNAME - fichier de l'année de base; s'il n'est pas fourni, prend explicitement la valeur du fichier *CUR*
tous les facteurs de mise à jour utilisés dans le fichier *cur*

Erreurs et avertissements

Des avertissements s'affichent si l'indication de source d'un paramètre n'est pas trouvée dans le fichier '*cur*' ni dans le fichier '*nxt*' ou si des méthodes de mise à jour différentes sont utilisées pour le même paramètre dans les deux fichiers.

Des messages d'erreur s'affichent et le programme est interrompu si les fichiers ne sont pas trouvés ou si les paramètres de référence ne sont pas admis comme des arguments à la ligne de commande ou s'ils ne figurent pas dans le fichier de commandes.

Facteurs de mise à jour actuellement utilisés dans les fichiers *mp*

LABEL description du processus appliqué
CPI Indice des prix à la consommation
CPIR Déflateur de l'IFC afin de calculer le revenu pour l'année précédente
CPIRR Déflateur de l'IFC afin de calculer le revenu de deux années auparavant
CPIM3 Indice des prix à la consommation - au delà de 3 %
CPINF Indice des prix à la consommation - Terre-Neuve

CPIPE Indice des prix à la consommation - île-du-Prince-Édouard
 CPINS Indice des prix à la consommation - Nouvelle-Écosse
 CPINB Indice des prix à la consommation - Nouveau-Brunswick
 CPIQU Indice des prix à la consommation - Québec
 CPION Indice des prix à la consommation - Ontario
 CPIMA Indice des prix à la consommation - Manitoba
 CPISA Indice des prix à la consommation - Saskatchewan
 CPIAL Indice des prix à la consommation - Alberta
 CPIBC Indice des prix à la consommation - Colombie-Britannique
 CPILAG Indice des prix à la consommation - IPC de l'année précédente
 CPINFLAG Indice des prix à la consommation - IPC de Terre-Neuve de l'année précédente
 CPIALLAG Indice des prix à la consommation - IPC de l'Alberta de l'année précédente
 CPISALAG Indice des prix à la consommation - IPC du Saskatchewan de l'année précédente
 CPIM3LAG Indice des prix à la consommation - au delà de 3 % de l'année précédente
 AIW moyenne des salaires hebdomadaires moyens de l'industrie
 UIR taux de croissance du maximum des salaires ouvrant droit à l'AC/AE
 NONE valeur 1, utilisée pour la déflation
 DEFAULT valeur 1, utilisée pour la déflation
 YEAR Année cible (4 chiffres)
 VERSION Version d'édition de la BD/MSPS [chaîne]
 SFILE Fichier original de paramètre fiscal/transfert [chaîne]

Facteurs de mise à jour actuellement utilisés dans les fichiers apr

LABEL description du processus appliqué
 CPI Indice des prix à la consommation
 AIW moyenne des salaires hebdomadaires moyens de l'industrie
 UIR taux de croissance du maximum des gains ouvrant droit à l'AC/AE
 VERSION Version d'édition de la BD/MSPS [chaîne]

Syntaxe

Les valeurs des paramètres de mise à jour s'affichent à la ligne de commande lorsqu'elles sont activées sous la forme ``name=value'` ou si elles sont transférées à la fonction `pupdate` dans un fichier.

`pupdate cur=file1 [nxt=file2] out=outfile [name=value]...`

ou

pupdate -f filename où *filename* contient

```

    cur=file1
    out=outfile
    [nxt=file2]
    [name1=value1]
    .
    .
    .
    [name=value]
    [CURNAME=label]
  
```

Exemples :

Pour mettre à jour le fichier ba99.mpr entier pour l'année prochaine et y insérer un taux de croissance de l'IPC fédéral de 3 % :

```
C:> pupdate cur=ba99.mpr nxt=ba00.mpr LABEL=Higher_Growth
YEAR=2000 CPI=1.03 CPIM3=1.0 AIW=1.026 CPIR=.952 CPIRR=.952
UIR=1.000 CPINF=1.018 CPIPE=1.018 CPINS=1.018 CPINB=1.018
CPIQU=1.018 CPION=1.018 CPIMA=1.018 CPISA=1.018 CPIAL=1.018
CPIBC=1.018 CPILAG=1.018 CPINFLAG=1.018 CPISALAG=1.018
CPIALLAG=1.018 CPIM3LAG=1.0 DEFAULT=1.0 NONE=1.0 VERSION=10.2
SFILE=ba99.mpr out=myout.mpr
```

OU

```
C:> pupdate -f growmpr.lst
```

où growmpr.lst contient les écritures suivantes :

```
cur=ba99.mpr
nxt=ba00.mpr
LABEL=Higher_Growth
YEAR=2000
CPI=1.03
CPIM3=1.0
AIW=1.026
CPIR=.952
CPIRR=.952
UIR=1.000
CPINF=1.018
CPIPE=1.018
CPINS=1.018
CPINB=1.018
CPIQU=1.018
CPION=1.018
CPIMA=1.018
CPISA=1.018
CPIAL=1.018
CPIBC=1.018
CPILAG=1.018
CPINFLAG=1.018
CPISALAG=1.018
CPIALLAG=1.018
CPIM3LAG=1.00
DEFAULT=1.0
NONE=1.0
VERSION=10.2
SFILE=ba99.mpr
out=myout.mpr
```

Pour augmenter les fichiers de rajustement de la base de données après 2004 :

```
C:> pupdate cur=ba96_03.apr LABEL=Grow_to_2004 CPI=1.018  
AIW=1.026 UIR=1.000 VERSION=10.2 out=new04.apr
```

Pour augmenter un fichier mpi qui ne renferme que les paramètres de l'impôt sur le revenu imposable de la Colombie-Britannique :

```
C:> pupdate cur=bctest.mpi out=newbc.mpi CPIBC=1.03 NONE=1
```

BLDSPD: FONCTION DE CRÉATION DE BASE DE DONNÉES .SPD

bldspd permet de convertir les fichiers de sortie ASCII, dans un format spécifique, en un fichier de format compressé pouvant être lu par le MSPS comme s'il s'agissait d'un fichier de base de données BDSPS; par conséquent, l'utilisateur peut créer une base de données contenant un sous-ensemble désiré ou une version révisée par l'utilisateur de la base de données BDSPS existante (v140y02.spd).

Par exemple, il est possible de créer une base de données contenant seulement un ménage en particulier destiné à être utilisé avec la fonction Points de changement, ou encore de créer une base de données contenant des cas rares de ménages qui peuvent servir au dépannage de nouveaux algorithmes. Les utilisateurs expérimentés peuvent aussi faire le "modélage" avancé des données.

Il faut noter que lorsqu'un sous-ensemble est créé, le résultat changera légèrement car le numéro choisi au hasard sera différent. Ceci implique que les transferts, qui dépendent de la participation (voir gis et GISST pour un exemple), peuvent changer. Si vous voulez avoir les mêmes résultats, il vous faudra désactiver tous les paramètres de participation reliés à celui-ci.

Les programmes de construction acceptent seulement des fichiers ASCII à zones fixes. Pour accéder facilement à ce style ASCII, il faut faire la lecture du fichier d'inclusion de paramètres de commande /spsd/bldspd.cpi, ainsi que le fichier d'inclusion des paramètres d'ajustement de base de données /spsd/bldspd.api.

Il faut noter que ces fichiers utilisent le style de sortie de la fonction Sortie ASCII (voir le document *Guide des paramètres* pour obtenir une description du paramètre ASCSTYLE). ASCSTYLE=5 est un format de fichier à zones fixes qui contient toutes les variables indiquées dans ASCVARS, dont le séparateur est un espace et qui contient tous les enregistrements par cas, en commençant par un enregistrement de ménage suivi par les enregistrements individuels.

Voici la syntaxe complète nécessaire à la compression du fichier de sortie ASCII en format de fichier lisible par machine, par la fonction bldspd :

```
bldspd [-a][-x] file1 file2
```

où :

```
[-|!]a
```

Cet argument facultatif spécifie si l'utilisateur désire ou non que le fichier ASCII soit compressé et joint à un fichier compressé existant (.spd). La valeur par défaut est !, ce

qui entraîne la création d'un nouveau fichier compressé. L'utilisateur peut utiliser l'option Ajout pour créer un fichier compressé à partir de plusieurs fichiers ASCII (par exemple, un fichier ASCII qui est si gros qu'il doit être réparti en sections pour économiser l'espace disque).

[- | !] x

Cet argument facultatif spécifie si le fichier ascii contient ou non des variables 'supplémentaire' additionnelles. L'option par défaut est de lire des variables supplémentaires dans la base de données. Ces champs sont laissés vide intentionnellement. Les utilisateurs peuvent utiliser ces variables soit avec la boîte de verre ou pour reconstruire la base de données.

file

file1 doit être un fichier de sortie ASCII (.prn) du format obtenu par ASCSTYLE=5. file2 doit être un nouveau fichier compressé (.spd) si l'option a n'est pas utilisée. Si l'option a est utilisée, file2 doit être un fichier compressé existant (.spd).

Le fichier bldspd.cpi se trouve dans le répertoire /spsd et contient 23 variables supplémentaires pour l'utilisateur. Ces fichiers d'inclusion bldspd.cpi et bldspd.api sont utilisés pour produire les fichiers d'entrées ASCII utilisés par bldspd.exe.

Exemple :

```
C:/SPSMBLD> bldspd test1.prn test1.spd
```

Le fichier de sortie ASCII test1.prn sera converti en fichier compressé et sera enregistré sous le nom test1.spd. Pour exécuter le MSPS avec la nouvelle base de données, test1.spd, il faut donner au paramètre de commande INSPSD la valeur test1.spd.

Dans l'exemple précédent, test1.prn a lu la disposition de sortie du fichier /spsd/bldspd.cpi qui fournit la disposition finale pour le fichier d'entrées dans BD/MSPS, et il lit le fichier /spsd/bldspd.api. S'il n'y a pas lecture de ces deux fichiers, il y aura des erreurs graves.

BLDFXV: FONCTION DE CRÉATION DE BASE DE DONNÉES .FMX

bldfxv permet de convertir un fichier de sortie ASCII en fichier compressé qui peuvent être lus par le MSPS comme des bases de données de L'enquête sur les dépenses des ménages; l'utilisateur peut donc modifier la base de données EDM existante dans BDSPS pour ajuster la BDSPS en fonction de diverses hypothèses sur les dépenses des familles.

Le fichier de sortie ASCII doit avoir un format particulier décrit ci-dessous. Ce format peut être invoqué par la lecture du fichier \spsd\bldfxv.cpi ainsi que du fichier correspondant \spsd\bldfxv.api.

Voici la syntaxe complète utilisée pour le lancement de bldfxv :

```
Bldfxv [-a] file1 file2
```

où :

[- | !] a

Cet argument facultatif précise si l'utilisateur désire ou non que le fichier ASCII soit compressé et joint à un fichier compressé existant (.fxv). La valeur par défaut est !, ce qui fait qu'un nouveau fichier compressé est créé. L'utilisateur peut utiliser l'option Ajout qui crée un fichier compressé à partir de plusieurs fichiers ASCII (par exemple, un fichier ASCII peut être si gros qu'il doit être réparti en sections pour conserver l'espace disque).

file

file1 doit être un fichier de sortie ASCII (.prn) obtenu avec ASCSTYLE=4., file2 doit être un nouveau fichier compressé (.fxv) si l'option a n'est pas utilisée. Si l'option a est utilisée, file2 doit être un fichier compressé existant (.fxv).

Exemple :

```
C:/MYDIR> bldfxv test1.prn test1.fxv
```

Le fichier de sortie ASCII test1.prn, qui a été produit avec /spsd/bldfxv.cpi, sera converti en fichier compressé, test1.fxv. Pour exécuter le MSPS avec la base de données BDSPS ajustée en fonction des nouvelles hypothèses de dépense des familles, test1.fxv, il faut donner au paramètre de commande INPFXV la valeur test1.fxv.

La fonction bldfxv peut être utilisée pour le traitement d'un fichier ASCII produit à partir du MSPS lorsqu'un sous-ensemble de ménages a été sélectionné (comme ceux d'une province en particulier). Cependant, l'utilisateur doit utiliser la procédure suivante pour créer le fichier ASCII en modifiant les valeurs de paramètre données dans /spsd/bldfxv.cpi.

1. Il faut créer une variable utilisateur dont la valeur est 1 pour le ménage et cette variable devrait être utilisée plutôt que fxclohhv dans ASCVARS (qui est la valeur par défaut dans le fichier /spsd/bldfxv.cpi). Donc, newvar=1/hhnin; ou HH:newvar=1; fonctionneraient tous les deux.
2. SELSPEC ne devrait pas comprendre une expression qui a la forme (hdfrstfx==1) ou (hdlastfx==1) ou encore une forme équivalente. Ceci forcera la production à la sortie d'un enregistrement EDM pour chaque ménage sélectionné autrement.

Ensemble, ces changements produiront un fichier ayant des vecteurs de régimes de dépenses correctement doublés. Ce changement permettra à l'utilisateur de créer un ensemble complet de fichiers .spd, .fxv et .wgt pour des provinces en particulier, ce qui améliorera la performance si c'est leur mode d'utilisation normal.

Dans l'exemple précédent, test.prn a lu la disposition de sortie du fichier /spsd/bldfxv.cpi qui donne la disposition finale pour le fichier d'entrées du BD/MSPS et il lit le fichier /spsd/bldfxv.api. Si ces deux fichiers ne sont pas lus, il y aura des erreurs graves.

BLDWGT: FONCTION DE CRÉATION DE BASE DE DONNÉES .WGT

`bldwgt` permet de convertir des fichiers de pondération ASCII en fichiers compressés qui peuvent être lus par le MSPS. Avec la fonction `bldwgt`, l'utilisateur peut modifier les fichiers de pondération existants de la BDSPP en vue de changer la composition de la base de données.

Par exemple, pour accroître le nombre de prestataires de l'A.-E. dans la base de données, l'utilisateur pourrait ajuster les poids d'A.-E. de façon à allouer des transferts d'A.-E. à plus d'individus à l'intérieur de la base de données. L'utilisateur devrait savoir que, bien que le processus d'ajustement des fichiers de poids soit simple, les répercussions de tout changement sont complexes et l'utilisateur devrait comprendre pleinement toutes les interactions entre les fichiers de poids existants avant de tenter d'apporter ses modifications.

Voici la syntaxe complète utilisée pour lancer `bldwgt` :

```
bldwgt [-a] totwgt file1 file2
```

où :

`[-|!|]a`

Cet argument facultatif précise si l'utilisateur désire ou non que le fichier ASCII soit compressé et joint à un fichier compressé existant (`wgt`). La valeur par défaut est `!`; elle fait en sorte qu'un fichier compressé nouveau est produit. L'utilisateur peut utiliser l'option `Ajout` pour créer un fichier compressé à partir de plusieurs fichiers ASCII (p. ex., un fichier ASCII qui serait si gros qu'il devrait être réparti en sections afin de conserver l'espace disque).

`totwgt`

Il s'agit de la somme des poids du fichier à créer. Il représente la population estimée dans des nombres de ménages. Dans le cas particulier où un échantillon complet est créé (c.-à-d. `SAMPLEREQ` est 1.0000), l'utilisateur peut placer la valeur 0 dans le champ et le programme `bldwgt` fait la somme des poids dans le fichier.

`file`

`file1` doit être un fichier de sorties ASCII (`.wgt`) obtenu avec `ASCSTYLE=4`. `file2` doit être un nouveau fichier compressé (`.wgt`) si l'option `a` n'est pas utilisée. Si l'option `a` est utilisée, `file2` doit être un fichier compressé existant (`.wgt`).

Exemple :

```
C:/SPSMBLD> bldwgt 0 test1.prn test1.wgt
```

Le fichier de sorties ASCII `test1.prn` sera compressé dans un fichier lisible par machine. Le fichier de poids peut être utilisé pour l'ajustement de la base de données pendant l'exécution du MSPS (donner aux paramètres de commande `INPWGT` la valeur `test1.wgt`).

FONCTION DE CRÉATION DE BASES DE DONNÉES BLDMRS: .MRS

Cet outil convertit un fichier texte en un fichier de résultats binaires (`.mrs`) du MSPS. La fonction `bldmrs` accepte trois arguments : le premier désigne le nombre de ménages, le deuxième représente le fichier d'entrée

ASCII et le troisième constitue le nom du fichier .mrs ainsi créé. Le nombre de ménages peut être trouvé en utilisant le 'spsdinfo' dans la base de données complète.

Le fichier d'entrée à la fonction `bldmrs` respecte une présentation particulière qui se compose de deux sections : une section d'en-tête suivie d'une section de données. La première ligne de la section d'en-tête renferme les noms des variables. Ces noms peuvent être facultativement placés entre guillemets (") et peuvent être séparés par un ou plusieurs espaces. Cette ligne est suivie d'une section facultative qui peut renfermer des étiquettes et des données de niveau propres aux variables d'utilisateur. Ces renseignements descriptifs sont fournis entre une série d'indications du MSPS délimitée par des deux points à l'aide des indications `label` et `levels` décrites dans le *Guide de l'utilisateur*. Les lignes de renseignements descriptifs doivent commencer par un caractère alphabétique minuscule à la première colonne.

La section des données se compose d'une ligne pour chaque personne prise en compte dans le MSPS. Chaque ligne comprend les valeurs des variables propres à cette personne, séparées par un espace.

L'exemple qui suit montre la présentation du fichier d'entrée utilisé pour produire un fichier de résultats comportant les variables modélisées du MSPS pour le revenu disponible et le revenu de consommation. Il pourrait aussi être produit en utilisant la fonction `Sortie en texte` et `ASCSTYLE` est égale à 3.

```
immdisp immicons
0 0
1000 987
25000 22500
.
.
.
```

L'exemple ci-après indique la façon de créer un fichier .mrs qui peut être utilisé pour modifier la variable de province dans un cycle du MSPS (à l'aide de la fonction de valeur de référence).

```
hdprov
5
5
5
.
.
.
```

L'exemple ci-dessous montre la façon de créer un fichier .mrs qui renferme des variables d'utilisateur. Une variable de classification pour le niveau de pauvreté et une variable de revenu de base sont également définies.

```
baseinc povstat
label(baseinc) = "Base Income";
label(povstat) = "Poverty Status";
levels(povstat) = "Poverty", "Near Poverty", "Non-poverty";
10000 0
10000 0
10000 0
50000 2
.
.
.
```

EXEMPLE: CRÉER UNE BASE DE DONNÉES DE SOUS-ENSEMBLE

Dans l'exemple suivant, une base de données sera créée pour tous les individus de Terre-Neuve, ce qui constitue un sous-ensemble de l'ensemble de la base de données. Le processus se déroule en trois étapes et exige au moins trois exécutions distinctes du modèle.

1. CONSTRUCTION DES FICHIERS DE POIDS (.wgt)

A) Exécuter le MSPS. On commence l'exécution en sélectionnant le fichier de paramètres de commande par défaut et en donnant NFLDWGT aux noms de fichiers de sortie.

B) Lire le fichier d'inclusion `\spsd\bldwgt.cpi` pour faire la mise à jour des paramètres de commande.

C) Modifier les paramètres de la fonction Sélection et les paramètres de la fonction Tableaux croisés, de la façon suivante :

```
SELFLAG      1
SELSPEC      hdprov==NFLD
XTFLAG       1
XTSPEC       HH:units:S=0}
```

D) Utiliser `bldwgt.exe` pour créer le nouveau fichier de poids appelé `NFLDYY.WGT`, où YY a trait à l'année du fichier de poids sélectionné pendant l'exécution du modèle. La somme des poids se trouve dans la table de tableaux croisés du fichier `.tbl`.

```
C:>bldwgt 162282 nfldwgt.prn nflddy.wgt
```

Parce qu'un fichier de poids différent est lu pour chaque année, vous devez répéter toute l'étape pour chaque année pour laquelle vous désirez faire l'analyse.

2. CONSTRUCTION DU FICHIER EDM (.fxv)

A) Exécuter le MSPS. Il faut commencer l'exécution en sélectionnant le fichier de paramètres de commande par défaut de l'année de base et donner NFLDFXV comme noms des fichiers de sortie.

B) Lire le fichier d'inclusion `\spsd\bldfxv.cpi` pour faire la mise à jour des paramètres de commande.

C) Modifier la fonction sélection comme ceci :

```
SELSPEC      hdprov==NFLD
```

D) Modifier le paramètre de fonction Variable utilisateur de la façon suivante :

```
UVARFLAG     1
UVAR          a=1.0/hhnin;
```

E) La variable a est utilisée pour faire un double correct des vecteurs de dépenses et il faudrait l'utiliser plutôt que `fxclohhv` dans le paramètre `ASCVARS` de la fonction Sortie texte. Modifier `ASCVARS` de la façon suivante :

```
ASCVARS      fxseqhv fxseqhv hdseqhh a
```

F) Lire le fichier d'inclusion \spsd\bldfxv.api pour faire la mise à jour des paramètres d'ajustement, puis exécuter le modèle.

G) Utiliser bldfxv.exe pour créer le nouveau fichier FXV appelé NFLDYY.FXV.

```
C:\bldfxv nflfxv.prn nflddy.fxv
```

3. CONSTRUCTION DU FICHIER DE BASE DE DONNÉES (.spd)

A) Exécuter le MSPS. Il faut commencer l'exécution en sélectionnant le fichier de paramètres de commande par défaut de l'année de base et donner NFLDSPD aux noms de fichiers de sorties.

B) Lire le fichier d'inclusion \spsd\bldspd.cpi pour faire la mise à jour des paramètres de commande.

C) Modifier la fonction sélection comme ceci :

```
SELSPEC      hdpv==NFLD
```

D) Lire le fichier d'inclusion \spsd\bldspd.api pour faire la mise à jour des paramètres d'ajustement puis exécuter le modèle.

E) Utiliser bldspd.exe pour créer le nouveau fichier SPD appelé NFLDYY.SPD

```
C:\bldspd nflspd.prn nflddy.spd
```

4. CONSTRUCTION D'UN FICHIER D'INCLUSION (.cpi)

A) Créer un fichier d'inclusion NFLDYY.CPI pour remplacer les fichier d'entrées BDSPS INPWGT, INPFXV, INPSPD par les nouveaux fichiers que l'on vient de créer. Ce fichier devrait contenir les instructions suivantes :

```
INPSPD      nflddy.spd
FXVFLAG      1
INPFXV      nflddy.fxv
WGTFLAG      1
INPWGT      nflddy.wgt
```

Lorsque vous exécutez le MSPS avec la nouvelle base de données, il suffit de lire le fichier d'inclusion nflddy.cpi pour faire la mise à jour des paramètres de commande de façon qu'ils pointent vers les fichiers de données appropriés.

AVERTISSEMENT

Les résultats différeront légèrement de la BDSPS complète. Cela provient du fait de l'utilisation de nombres aléatoire dans le calcul des taux de participation au SRG peu importe les autres taux de participation qui sont utilisés dans l'année modèle. Les personnes se verront attribuer un nombre différent de la séquence des nombres aléatoires. Si vous désirez comparer les résultats à ceux qui ont été obtenus de la base de données

complète, il faut donner la valeur zéro à GISTURFLAG et à tous les autres paramètres de participation.

Utilitaires de programmation

GREP

Cette fonction fait la recherche de tous les fichiers spécifiés (des spécifications de fichiers à caractères de remplacement multiples dans le répertoire courant sont permises) pour une chaîne donnée et affiche les lignes dans lesquelles la chaîne se trouve. Deux options sont permises : -n précède chaque ligne où il y a concordance, avec son numéro de ligne, et -l donne seulement une liste des noms de fichier qui respectent les critères. `grep` est très utile comme outil de renvoi en direct pour la consultation des fichiers de code source ou des fichiers de paramètres. Par exemple, si on donnait l'instruction

```
C>grep -l capg *.c
```

dans le répertoire `\spsm\glass`, le produit de sortie, qui indique tous les modules qui ont rapport aux gains en capital, ressemblerait à ce qui suit :

```
agis.c
amemol.c
atxcalc.c
atxinet.c
atxitax.c
atxqinet.c
gis.c
memol.c
txcalc.c
txinet.c
txitax.c
txqinet.c
```

Dans un autre exemple d'une version antérieure de la BD/MSPS, si l'instruction

```
C>grep CTCPC *.mpr
```

était donnée dans le répertoire `\spsd`, la sortie ressemblerait à ce qui suit :

ba82.mpr: CTCPC	343.00	# Child tax credit per child
ba84.mpr: CTCPC	367.00	# Child tax credit per child
ba85.mpr: CTCPC	384.00	# Child tax credit per child
ba86.mpr: CTCPC	454.00	# Child tax credit per child
ba87.mpr: CTCPC	489.00	# Child tax credit per child
ba88.mpr: CTCPC	559.00	# Child tax credit per child
ba88y88.mpr: CTCPC	474.06	# Child tax credit per child
ba89.mpr: CTCPC	565.15	# Child tax credit per child
ba89y88.mpr: CTCPC	460.40	# Child tax credit per child
sq88.mpr: CTCPC	524.00	# Child tax credit per child
sq88y88.mpr: CTCPC	444.38	# Child tax credit per child

Comme on peut le voir, la valeur de CTCPC dans chaque fichier de paramètre d'impôt/de transfert du répertoire `\spsd` est affichée.

SUMSKIP

Il s'agit d'un fichier exécutable qui aide Statistique Canada à vérifier si vous avez installé correctement votre version du BD/MSPS ou s'il y a tout problème associé à un fichier.

Voici un exemple de cette instruction :

```
C>sumskip c:\spsd\ba08.mpr
```

Cette instruction produit à la sortie quelque chose qui ressemble à ce qui suit :

```
skip=194 bytes=254255 checksum=3234115767
```