GNUMDATA

(a program for Mastering DATA files)

Gilles J. Hunault & F. Beaujard

Mail adress for comments and bug reports:
gilles.hunault@univ-angers.fr

This manual is for GNUMDATA version 2.1.

Table des matières

1. Ce que peut gnumdata pour vous

gnumdata est un programme de manipulations ("actions") de fichiers textes dont le rôle est de servir d'intermédiaire entre les données brutes d'édition ou de saisie et les logiciels de traitement de ces données comme *Dbase*, *Excel*, *Gnuplot*, *Sas* ou même *GeomView*. A l'origine, **gnumdata** devait servir à compléter les colonnes vides, c'est à dire aider à passer d'un fichier qui contient, disons

```
XX  X    X
97  1    1
         2
         3
98  1    0
    2
    3
    4
    5
```

à un fichier où toutes les lignes sont remplies, par exemple avec le contenu de la ligne précédente, ce qui donnerait ici

```
XX  X    X
97  1    1
97  1    2
97  1    3
98  1    0
98  2    0
98  3    0
98  4    0
98  5    0
```

gnumdata a ensuite évolué vers des actions de gestion plus générales, comme mise à zéro, calcul du nombre de jours à partir du numéro de jour et de mois (voire d'année pour les bissextiles) et des actions de type calcul statistique (moyenne, variance, nombre de valeurs distinctes, dictionnaire des termes rencontrés...), des options (sortie *Dbase*, *Sas*, *Gnuplot*, *Gif*..., filtrage des premières ou des dernières lignes...).

Pour utiliser **gnumdata**, il faut avoir un fichier de données et un fichier d'actions, qui a le même nom que le fichier de données mais l'extension .stm.

Par exemple, si les données sont dans essai.dat, **gnumdata** ira lire essai.stm comme fichier d'actions, si les données sont dans Test.Data, le fichier d'actions sera Test.stm etc.

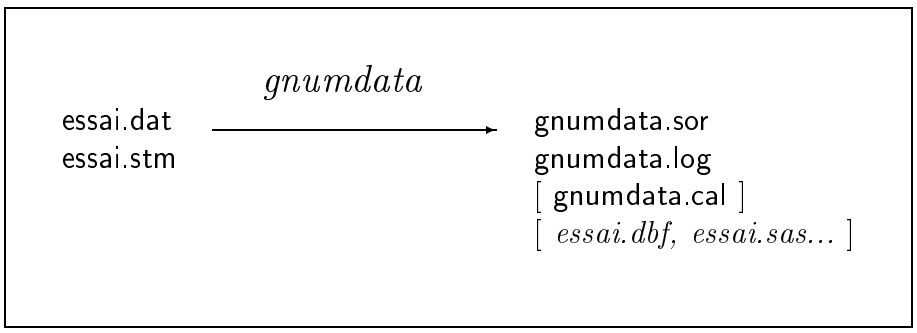
La syntaxe d'appel de gnumdata est

```
gnumdata nom_de_fichier
```

Le programme **gnumdata** commence par parcourir le fichier des actions. Ces actions ou "commandes de traitement" ne sont pas *case sensitive* et on peut donc les écrire indifférement en majuscules ou en minuscules. Les actions présentes dans ce fichier sont mémorisées puis le programme lit le fichier des données et enfin, les actions sont exécutées. Suivant ce qui a été demandé, on dispose alors des fichiers

- gnumdata.sor (fichier des résultats),
- gnumdata.cal (fichier des calculs),
- gnumdata.log (fichier-trace de l'exécution).

En fonction des options, d'autres fichiers peuvent être créés, avec un type traditionnel (comme .dbf, .sas, .plt, .gif etc.). Le déroulement normal de **gnumdata** peut donc être représenté comme suit, pour essai.dat comme fichier d'entrée.



Il n'est pas possible d'utiliser directement différents fichiers d'actions pour le même fichier de données. Toutefois, il suffit de recopier à son tour chaque fichier d'action avec le type .stm pour utiliser le fichier d'actions correspondant. Par exemple, avec les deux fichiers d'actions essai.St1 et essai.St2 pour le même fichier de données essai.dat, il suffit d'écrire

```
(sous Dos)
copy     essai.St1     essai.stm
gnumdata essai.dat
copy     essai.St2     essai.stm
gnumdata essai.dat

(sous Unix)
cp       essai.St1     essai.stm
gnumdata essai.dat
cp       essai.St2     essai.stm
gnumdata essai.dat
```

pour éxécuter successivement les deux fichiers d'actions et stocker les résultats "là ou il faut !". Les fichiers de résultats (*gnumdata.sor*, *gnumdata.cal*, *gnumdata.log*) sont recopiés directement par **gnumdata** grâce aux actions OUTS, OUTC, et OUTL.

2. Format des données, des actions

Les données doivent être soit cadrées au format SDF (Standard Data Form) c'est à dire bien alignées, unité sous unité, dizaine sous dizaine, etc. soit vides. Bien sûr, les données numériques sont cadrées à droite et les données caractères à gauche.

Le fichier .stm, comme le fichier de données, peut comporter des lignes de commentaire, repérées par le symbole #.

Attention ! Les lignes vides de données et les lignes vides d'actions ne sont pas traitées de la même façon. Une ligne vide de données contient des données (qui correspondent souvent aux valeurs des lignes précédentes en même position), les lignes vides d'actions sont purement et simplement ignorées.

Le format des actions supportées par **gnumdata** est le plus souvent

```
nact numc nomc  [ parm1 [parm2 [ ... ] ]
```

où

- `nact` le nom de l'action à exécuter,
- `numc` est le numéro de la colonne à traiter,
- `nomc` le nom de cette colonne,
- `parm1`, `parm2`... les paramètres éventuels à utiliser.

Par exemple, pour utiliser l'action `BNVAL` (nombre de valeurs) sur la colonne 3 nommée `POIDS`, on écrira

```
NBVAL 3 POIDS
```

De même, pour indiquer que l' unité utilisée est, disons les kilos, on utilisera l'action `UNIT` sur la même colonne avec le paramètre `kg`, soit

```
UNIT 3 POIDS kg
```

3. Liste des actions de gnumdata

Les actions suivantes sont disponibles dans gnumdata. On trouvera pour chacune un exemple d'utilisation un peu plus loin dans ce manuel. On peut classer en gros les actions en trois classes : gestion (G), calcul (C) ou option (O). Certaines actions ne sont effectuées par gnumdata que si la valeur pour la colonne correspondante est vide (comme l'action `PREC`), d'autres ne le sont que si la valeur dans la colonne est non vide (comme l'action `NBVAL`).

Liste des actions de Gestion pour **gnumdata**

ADD	addition de deux colonnes
CONC	concaténation de deux colonnes
CONST	création d'une colonne constante
CREATE	création d'une colonne complète (Nom,Typ,Long,Dec,Unit,vpf)
DAYS	comptage du nombre de jours
DELE	suppression de la colonne
DESCV	description complète (Nom,Typ,Long,Dec,Unit) de la colonne
DIV	division de deux colonnes
LEFT	remplacement par la valeur à gauche
LENGTH	définition de la longueur de la colonne
MUL	multiplication de deux colonnes
NAME	désignation de la colonne
NOBS	sortie du numéro de ligne de données
PREC	remplacement par la valeur en ligne précédente
RIGHT	remplacement par la valeur à droite
SUB	soustraction de deux colonnes
TYPE	type de la variable (C ou N)
ZERO	mise à zéro

Liste des actions de Calcul pour **gnumdata**

CROSS	tri croisé (tableau de contingence)
CV	coefficient de variation (σ/m)
FREQ	tri à plat (fréquences de chaque valeur)
MAX	maximum des valeurs non vides
MIN	minimum des valeurs non vides
MEAN	moyenne m des valeurs non vides
NBVAL	nombre de valeurs non vides
STD	écart-type σ des valeurs non vides
SUM	somme des valeurs non vides
UNIT	unité associée à la colonne
VAR	variance des valeurs non vides

Liste des actions d'Options pour **gnumdata**

FIRST	sortie des n premières lignes de données
SKIP	omission en sortie des n premières lignes de données
LAST	sortie des n dernières lignes de données
OUTC	copie du fichier de calculs (gnumdata.cal)
OUTS	copie du fichier de sortie (gnumdata.sor)
OUTL	copie du fichier de trace (gnumdata.log)
RULER	affichage d'une réglette graduée
DBF	sortie des données au format *Dbase* (fichier .dbf)
	(requiert le programme externe CreeDbf.exe pour *Dos/Windows*)
PLOT	sortie au format gnuplot (fichier .plt)
GIF	sortie au format *gnuplot* (fichier .plt)
	avec instructions de création d'un fichier image .gif
	(requiert *gnuplot* version 3.5 patchlevel 349 ou supérieure)
SAS	sortie au format .sas pour créer une table avec les données

On se méfiera de ce que les actions ne sont pas indépendantes. Certaines acions n'ont pas le même sens avec ou sans d'autres instructions. Par exemple, avec notre fichier de départ

```
XX  X   XX
97  1    1
         2
         3
98  1    0
    2
    3
    4
    5
```

si l'action `NBVAL` est utilisée sans l'action `PREC`, **gnumdata** trouve 2 valeurs non vides pour la colonne 1 alors que si on exécute l'action `PREC` et l'action `NBVAL`, alors **gnumdata** trouve 8 valeurs non vides pour la colonne 1.

Par contre, l'ordre des actions (`PREC` puis `NBVAL` ou `NBVAL` puis `PREC`) est indifférent et aboutit aux mêmes résultats. On notera aussi que les options `FIRST`, `LAST` et `SKIP` ne s'appliquent pas aux calculs mais au fichier de sortie gnumdata.sor.

4. Comment gnumdata reconnait les colonnes

gnumdata se base sur le cadrage de la première ligne du fichier de données qui n'est pas un commentaire pour repérer les positions physiques des colonnes dans les lignes de données. Il n'est pas possible de se baser sur la première ligne de données car celle-ci peut ne pas avoir le bon format. Par exemple la ligne `AXT␣␣␣1␣␣12` ne permet pas de savoir si la colonne 2 comporte un ou plusieurs chiffres. On peut indiquer les positions des colonnes par une suite de `X` consécutifs ou par un label plus lisible ; on prendra soin de compléter des labels courts par des espaces soulignés pour que **gnumdata** trouve bien ces positions. Ainsi la ligne `XXXX␣XX␣XXX` indique que la colonne 1 occupe les positions 1 à 4 dans la ligne, la colonne 2 les positions 6-7 et la colonne 3 les positions 9-12. Il serait sans doute plus lisible d'écrire `nom_␣i_␣nbe` mais pour les longues lignes, le `X` est souvent plus simple à écrire. D'autre part, comme **gnumdata** peut supprimer des colonnes, il écrit systématiquement une ligne correcte de "X" avant la première ligne de données (l'ancienne ligne de description est mise en commentaire).

Pour que **gnumdata** lise vos fichiers de données, il suffit donc de rajouter une ligne en début de fichier pour indiquer comment déterminer les colonnes. L'analyse du format et le découpage des colonnes est systématiquement indiqué par **gnumdata** dans le fichier de trace (gnumdata.log). Il est possible avec l'option `RULER` n d'afficher une réglette sur les n premiers caractères des 5 premières lignes pour vérifier les cadrages définis. On obtient par exemple pour notre fichier *essai.dat* avec l'action `RULER 15` les spécifications

```
POSITIONS OF THE COLUMNS IN THE INPUT FILE
   column   1 named : An              start   1 end   2
   column   2 named : Série           start   5 end   5
   column   3 named : Nb._test(s)     start  10 end  10
```

Les premières lignes du fichier de sortie gnumdata.sor sont alors

```
# old format
#XX  X    X
#....+....1....+
#97  1    1
#         2
#         3
#98  1    0
#    2
```

5. Un exemple détaillé

Reprenons comme fichier de référence *essai.dat* qui contient :

```
# exemple pour gnumdata
  An    Longueur
  96      150
          130
  97      128.30
          129.30
          130.30
  98
          140
  99      145.2
```

Notre première action sera de rajouter les années là où elles manquent, en utilisant l'année de la ligne précédente. Notre seconde (de circonstance), sera de passer du format YY au format YYYY c'est à dire de remplacer les 98 par 1998 etc. Notre troisième sera de compter le nombre de données par année, de calculer la moyenne et l'écart-type du champ longueur. Enfin, notre dernière action viendra construire des fichiers pour *Dbase*, *gnuplot* et *SAS*.

Utilisons (même si pour l'exemple ce n'est pas flagrant) l'option `FIRST` n pour tester nos manipulations sur les n premières lignes du fichier. Le fichier *essai.stm* contient donc les deux actions

```
# test sur les 5 premires lignes du fichier
First 5
# on complte la colonne An avec la ligne prcdente
Prec 1 Anne
```

et on exécute ces actions en tapant

```
gnumdata essai.dat
```

On obtient les messages écrans suivant pendant le traitement

```
 gnumdata : GNU Mastering Data -- (gH) version 2.1

 Analysis/Management of the file essai.dat via  essai.stm
   Reading        +             8 line(s)
   Processing     +             8 line(s)
   Writing        +             5 line(s) option(s) FIRST 5
 You may use :    gnumdata.log
                  gnumdata.sor

 Copyright  gilles.hunault@univ-angers.fr
            http://www.info.univ-angers.fr/pub/gh/gh.html
```

et le fichier gnumdata.sor contient alors

```
# exemple pour gnumdata
XX XXXXXXXX
96      150
96      130
97    128.3
97    129.3
97    130.3
```

Le fichier de trace, gnumdata.log, quant à lui contient :

```
 ANALYSIS/MANAGEMENT OF THE FILE essai.dat VIA  essai.stm
 -- 04/01/1999 16:47

 ACTION NUMBER     1
    THE OPTION *FIRST* WILL BE USED
    WHICH MEANS  :  utilisation des n premires lignes
    USING THE FOLLOWING PARAMETER(S) :  5

 ACTION NUMBER     2
    THE ACTION *PREC*
    WHICH MEANS  :  ajout de la valeur en ligne Prcdente si vide
    WILL BE USED ON COLUMN NUMBER   1 NAMED  "Anne"
```

```
 POSITIONS OF THE COLUMNS IN THE INPUT FILE
    column   1 named : Anne           start   3 end   4
    column   2 named : ?????????????? start   9 end  16

 -- end of processing essai.dat
```

Remarquons qu'il aurait été possible de nommer la variable 2 avec l'action `NAME`, de cadrer les longueurs en utilisant `TYPE`, mais c'est du détail pour ce qui nous intéresse ici. Nous le ferons avec l'étape suivante qui est de passer de 96 à 1996. Il nous faut commencer par créer une colonne qui contiendra les valeurs 1900, puis additionner cette colonne à la colonne "An" soient les instructions

```
# test sur les 5 premires lignes du fichier
First 5
# on complte la colonne An avec la ligne prcdente
Prec  1 Anne
# on nomme la colonne 2
Name 2 Longueur
# on invente la colonne de valeur 1900
Const 3 amnc 1900
# et on met dans la colonne 1 la somme colonne 1 + colonne 3
Add   1 Anne 3
# on n'a plus besoin de la colonne 3 donc on la dtruit
Dele  3 amnc
# mettons 4 positions pour la colonne 1
Length 1 Anne 4
```

Attention ! La colonne 1 utilisait deux chiffres avant notre manipulation. La ligne qui déclarait son format était `␣␣An␣␣␣␣Longueur`. Comme désormais la colonne 1 a 4 chiffres, il faut prévoir cette nouvelle longueur en forcant la machine à décaler les colonnes de sortie. C'est le role de l'action `LENGTH`.

Une autre façon de faire, si le fichier de données le permet, est de rajouter deux espaces soulignés ou deux signes moins au nom de la colonne 1, soit la ligne de déclaration `An--␣␣␣␣Longueur`.

Calculons maintenant le nombre de valeurs par année dans la colonne 1 et la moyenne puis l'écart-type de la colonne 2 par les lignes

```
#     compltion de la colonne 1
Prec  1 An
#     puis calcul du nombre de valeurs distinctes
Nbval 1 An
Freq  1 An
#     moyenne et cart-type, cdv de la colonne 2
Nbval 2 Longueur
Mean  2 Longueur
Std   2 Longueur
Cv    2 Longueur
Unit  2 Longueur cm
```

On obtient alors comme fichier de calculs

```
column 1 "An"        number of non empty values   :        8
column 2 "Longueur"  number of non empty values   :        7
column 2 "Longueur"  mean of non empty values     :      136.157 cm
column 2 "Longueur"  std of non empty values      :        8.187 cm
column 2 "Longueur"  std/m of non empty values    :        6.013 %
column 1 "An"        frequency of the 4 distinct non empty values
    value     96.00   97.00   98.00   99.00
    count      2.00    3.00    2.00    1.00
      %       50.00   75.00   50.00   25.00
```

Terminons par la création des fichiers .dbf, .sas et .plt pour les logiciels *Dbase*, *SAS* et *gnuplot* respectivement avec le tracé de y=Longueur en fonction de x=An. Il suffit de rajouter les actions :

```
DBF
SAS
PLT 1 An 2 Longueur
```

pour obtenir les fichiers demandés. On notera qu'en prime, **gnumdata** a créé un fichier d'extraction qui ne contient que les colonnes à tracer, le fichier essai.dag. Le fichier de tracé essai.plt contient alors les différents ordres pour *gnuplot*.

Toutes nos manipulations peuvent bien sûr être mises dans un seul et même fichier en respectant l'ordre des manipulations, soit

```
#
## un exemple de traitement
#

   # on complte la colonne An avec la ligne prcdente
   PREC 1 Anne
   # on nomme la colonne 2
   Name 2 Longueur

    # on invente la colonne de valeur 1900
    Const 3 amnc 1900
    # et on met dans la colonne 1 la somme colonne 1 + colonne 3
    Add   1 Anne 3
    # on n'a plus besoin de la colonne 3 donc on la dtruit
    Dele  3 amnc
    # demandons  la machine de mettre 4 positions pour la colonne 1
    Length 1 Anne 4
    # utilisons la description longue pour la colonne 2
    Descv 2 Longueur N 7 2 cm

    # calculs statistiques
    Nbval 1 An
    Freq  1 An
    #     moyenne et cart-type, cdv de la colonne 2
    Nbval 2 Longueur
    Mean  2 Longueur
    Std   2 Longueur
    Cv    2 Longueur

    # sauvegardes formats divers
    DBF
    SAS
    PLT 2 Longueur 1
```

On remarquera que nous avons enlevé l'action `FIRST` puisque nous voulons travailler désormais avec toutes les lignes du fichier.

Le fichier essai.sas construit est

```
** File : essai.sas ;
** created by gnumdata (gH) from essai.dat and essai.stm ;
data essai ;
input  ANNEE $ 1-2 LONGUEUR 3-10 ANNEE 11-12 ;
datalines ;
96     1501900
       1301900
97     1281900
       1291900
       1301900
98       01900
       1401900
99     1451900
;
run ;
** pour vrification : ;
proc print ;
run ;
```

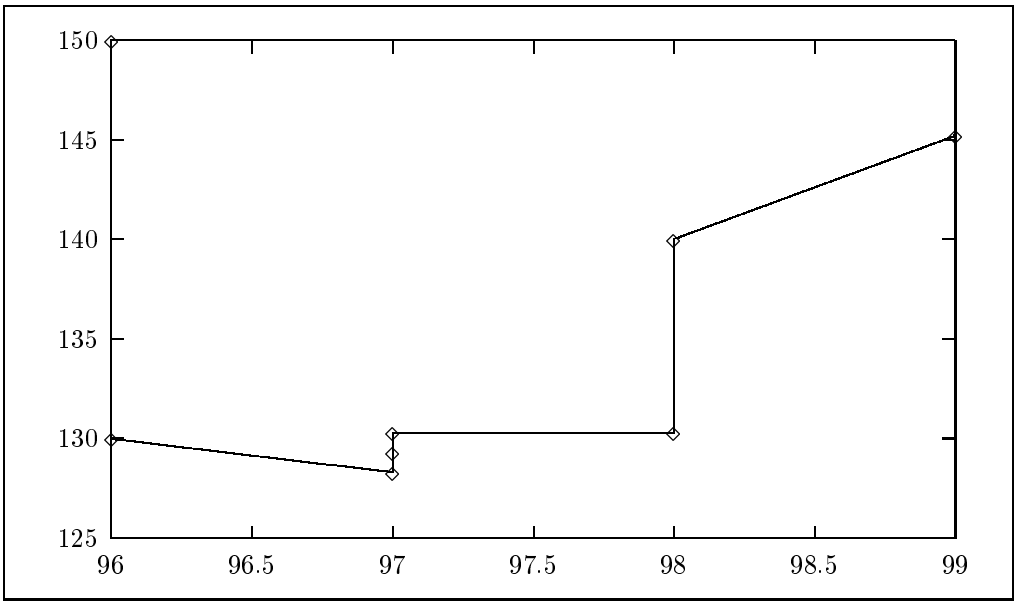
Pour *gnuplot*, **gnumdata** a créé le fichier de données essai.dag suivant

```
96   150
     130
97   128.30
     129.30
     130.30
98
     140
99   145.2
```

et le programme essai.plt pour *gnuplot* est

```
set nokey
set term gif
set output 'essai.gif'
plot "essai.dag" with lines, "essai.dag" with points
```

Si on l'exécute par gnuplot essai.plt, on obtient à l'écran à peu près ceci :



On peut tester sous *Dbase* que le fichier essai.dbf est utilisable :

```
. use essai
. list stru
     Structure de la base de donnes: C:essai.dbf
     Nombre total d'enregistrements :       8
     Date de la dernire mise  jour: 14/01/99
     Champ  Nom champ   Type          Dim    Dec
         1  AN          Numerique       4
         2  LONGUEUR    Numerique       7      2
. list
  Enreg.      AN LONGUEUR
      1       96   150.00
      2       96   130.00
      3       97   128.30
      4       97   129.30
      5       97   130.30
      6       98   130.00
      7       98   140.00
      8       99   145.20
```

6. Détail des actions de gnumdata

(to be written...)

6.1 Action ADD

6.2 Action CROSS

6.3 Action CONST

6.4 Action CV

6.5 Action DAYS

7. Ggnumdata par l'exemple

(to be written...)

7.1 Exemple 01

7.2 Exemple 02

7.3 Exemple 03

7.4 Exemple 04

7.5 Exemple 05

8. ANNEXES

8.1 Désignation des fichiers

.CAL	Fichier de calculs (moyenne, fréquences...) issu de gnumdata
.DAG	Fichier des données pour gnuplot
.DAT	Fichier des données traitées
.DBF	Fichier au format *Dbase* construit à partir des données
.GIF	Fichier image du tracé construit par *Gnuplot*
.LOG	Fichier trace des actions (NAME, NOBS...) issu de gnumdata
.PLT	Fichier des paramètres du tracé pour gnuplot
.SAS	Fichier-programme en langage *Sas* pour construire une table des données
.SDF	Fichier au format fixe (standard data form) construit à partir des données
.SOR	Fichier de sortie des données
.STM	Fichier des actions (NAME, NOBS...)
.STR	Fichier structure des colonnes pour *Dbase*

8.2 Syntaxe des actions

Pour ce qui suit, on convient de noter

nl	un nombre de ligne,
n et n_i	un numéro de colonne,
cn et cn_i	un nom de colonne,
fn	un nom de fichier,
nc	un nombre de caractères,
nd	un nombre de décimales,
rn	un décalage relatif,
v	une valeur (numérique ou caractère),
t	un type de données (N ou C),
u	un mot désignant l'unité de traitement (cm, kg...).
vpf	une valeur par défaut

Un mot entre crochets droits [et] est facultatif.

Action	Paramètres	Exemple d'utilisation			
ADD	$c_1\ cn_1\ c_2\ cn_2$	`Add`	`1`	`Long`	`2 Larg`
CONC	$n_1\ cn_1\ n_2\ cn_2\ n_3\ cn_3$ [Sym.v]	`Conc`	`3`	`Time 1 Hr 2 Min Symbol h`	
CONST	$n\ cn\ v\ t\ nc\ nd\ u$	`Const`	`4`	`Weight`	`-1 N 3 0 kg`
CREATE	$n\ cn\ t\ nc\ nd\ u\ vpf$	`Create`	`1`	`Long`	`N 7 1 mile(s) -1`
CROSS	$c_1\ cn_1\ c_2\ cn_2$	`Cross`	`1`	`Sex`	`2 Town`
CV	$c\ cn$	`Cv`	`1`	`Width`	
DAYS	$c\ cn\ c_1\ cn_1\ c_2\ cn_2$	`Days`	`9`	`Nbd`	`7 Dy 8 Mth`
DBF		`Dbf`			
DELE	$n\ cn\ nc$	`Dele`	`1`	`Long`	
DESCV	$n\ cn\ t\ nc\ nd\ u$	`Descv`	`1`	`Long`	`N 7 1 mile(s)`
DIC	$n\ cn$	`Dic`	`1`	`Textline`	
DIV	$c_1\ cn_1\ c_2\ cn_2$	`Div`	`1`	`Sum1`	`2 Sum2`
FIRST	nl	`First`	`9`		
FREQ	$c\ cn$	`Freq`	`2`	`Town`	
GIF	$c_1\ cn_1\ c_2\ cn_2$	`Gif`	`1`	`Month`	`2 Long`
LAST	nl	`Last`	`5`		
LEFT	$n\ cn\ [rn]$	`Left`	`4`	`Ref`	`-1`
LENGTH	$n\ cn\ nc$	`Length`	`5`	`Year`	`4`
MAX	$c\ cn$	`Max`	`1`	`Width`	
MEAN	$c\ cn$	`Mean`	`1`	`Width`	
MIN	$c\ cn$	`Min`	`1`	`Width`	
MUL	$c_1\ cn_1\ c_2\ cn_2$	`Mul`	`1`	`Sum1`	`2 Sum2`
NAME	$n\ cn$	`Name`	`7`	`Count`	
NBVAL	$c\ cn$	`Nbval`	`1`	`Width`	
NOBS		`Nobs`			
OUTC	fn	`Outc`		`test.cal`	
OUTL	fn	`Outl`		`comp.trace`	
OUTS	fn	`Outs`		`f15.out`	
PLOT	$c_1\ cn_1\ c_2\ cn_2\ [GIF]$	`Plot`	`1`	`Month`	`2 Long`
PREC	$n\ cn$	`Prec`	`5`	`Year`	
RIGHT	$n\ cn\ [rn]$	`Right`	`4`	`Ref`	`1`
RULER	$[nc]$	`Ruler`			
SAS		`Sas`			
SKIP	nl	`Skip`	`5`		
STD	$c\ cn$	`Std`	`1`	`Width`	
SUB	$c_1\ cn_1\ c_2\ cn_2$	`Sub`	`1`	`Sum1`	`2 Sum2`
SUM	$c\ cn$	`Sum`	`1`	`Width`	
TYPE	$n\ cn\ t$	`Type`	`2`	`Flower`	`C`
UNIT	$c\ cn\ u$	`Unit`	`1`	`Width`	`cm`
VAR	$c\ cn$	`Var`	`1`	`Width`	
ZERO	$n\ cn\ t$	`Zero`	`5`	`Year`	