INITIATION AU LOGICIEL SAS

(version 9.1.3 sous Windows)

Hélène HAMISULTANE

Bibliographie :

- Initiation au logiciel SAS(9) pour Windows, Coqué N. (juin 2006). www.agroparistech.fr/IMG/pdf/polySas.pdf

- SAS base, Introduction à SAS SQL, SAS IML. www.cnam.fr/mathematique/IMG/pdf/SAS_introduction_1.pdf

- SAS 9.1.3 Documentation, http://support.sas.com/documentation/onlinedoc/91pdf/index_913.html

Le logiciel SAS est destiné au traitement statistique des données. Il est constitué de plusieurs modules qui regroupent des programmes pré-installés appelés "procédures" (certains modules ne sont pas disponibles lors de l'installation de SAS). Parmi ces modules, nous pouvons citer brièvement :

- SAS Stat : procédures destinées aux régressions diverses et aux statistiques descriptives ;
- SAS ETS: procédures permettant le traitement des séries temporelles ;
- SAS IML : module permettant l'usage du langage matriciel.

Nous allons étudier les différentes fonctionnalités de ce logiciel à travers un exemple.

Soit le tableau de données suivant :

Année	Trimestre	Ventes	Pub
1988	1	164	34
1988	2	198	36
1988	3	85	32
1988	4	179	29
1989	1	168	45
1989	2	201	67
1989	3	98	76
1989	4	197	75

Nous souhaitons effectuer une régression de Ventes sur Pub par la méthode des Moindres Carrés Ordinaires (MCO). Pour ce faire, nous devons d'abord saisir les données du tableau dans le logiciel SAS.

ETAPE 1 : ENREGISTREMENT, LECTURE ET MODIFICATION DES DONNEES SOUS SAS

_ Entrez dans le logiciel en cliquant sur l'icône de SAS.

_ Nous nous trouvons devant 4 fenêtres :

- fenêtre Explorateur ;
- fenêtre Journal (ou fenêtre LOG) ;
- fenêtre Sortie (ou fenêtre OUTPUT) ;
- fenêtre Editeur (ou fenêtre EDITOR).

Fenêtre Explorateur

La fenêtre Explorateur contient les bibliothèques (ou Libraries) : WORK, SASUSER et SASHELP.

La bibliothèque WORK comprend les tables SAS dites temporaires tandis que la bibliothèque SASUSER regroupe les tables SAS dites permanentes (il est bien sûr possible de créer des bibliothèques "personnalisées" pour ranger nos tables permanentes).

Les tables temporaires sont automatiquement détruites à la fin de la session SAS ce qui n'est pas le cas pour les tables permanentes.

Fenêtre Journal

La fenêtre Journal dresse le compte-rendu de l'exécution d'un programme. C'est dans cette fenêtre qu'apparaissent les messages d'erreur.

Fenêtre Sortie

La fenêtre Sortie affiche les résultats d'un programme lorsque ce dernier s'est exécuté sans erreur.

Fenêtre Editeur

La fenêtre Editeur permet la saisie des instructions d'un programme.

Remarque :

Tout programme SAS est constitué d'une étape DATA et d'une étape PROC.

Au cours de l'étape DATA, les données, les variables et les tables SAS sont crées et modifiées (SAS ne peut travailler qu'avec des tables converties au format SAS). Tandis que l'étape PROC permet l'application d'opérations pré-programmées sur les données enregistrées au cours de l'étape DATA en faisant appel à des procédures (la régression de Ventes sur Pub sera effectuée à l'aide de la procédure PROC REG ou PROC AUTOREG).

A/ Etape DATA : création d'une table SAS

Il est possible de créer une table SAS soit en saisissant au clavier les données soit en important les données d'une table externe (table de format texte, Excel ou Access).

Création d'une table SAS à partir de la saisie des données :

Dans la fenêtre Editeur, on écrit les lignes d'instructions suivantes :

```
/*  Ecriture des données dans une table SAS */

LIBNAME  lsas "C:\travaux SAS\"  ;
DATA lsas.essai  ;
INPUT Ventes Pub ;
CARDS (ou DATALINES) ;
164  34
198  36
85  179
168  45
201  67
98  76
197  75
;
PROC PRINT DATA=lsas.essai;
RUN ;
```

On enregistre ensuite ce programme qui sera sauvegardé avec une extension .sas

On exécute le programme en cliquant sur "le personnage qui court" apparaissant dans la barre d'outils.

Remarques :

- On peut saisir des dates au clavier en précisant le format de la variable date à l'aide de DDMMYYw. (pour des dates écrites sur w ≥ 6 caractères), YYMMDDw. , MONYYw. (pour des dates écrites sur w ≥ 5 caractères où MON correspond au 3 premières lettres du mois en anglais) et ANYDTDTEw. (pour tous formats).

  Exemple :

```
LIBNAME  lsas "C:\travaux SAS\"  ;
DATA lsas.essai  ;
INPUT days DDMMYY10. Ventes Pub ;
CARDS ;
01/01/1988  164  34
01/04/1988  198  36
01/08/1988  85  179
```

01/12/1988 168 45
;
RUN ;
PROC PRINT DATA=lsas.essai;
FORMAT days date9. ;
/* Les instructions FORMAT days date9. permettent d'afficher lisiblement les dates sous la forme 01JAN1988, 01AVR1988, …, 01DEC1988. Date9. permet d'écrire les dates sur 9 caractères */
RUN ;

- On peut inscrire des commentaires à l'aide des caractères /* */ ou en plaçant en début de ligne *.
- L'instruction LIBNAME permet de créer une bibliothèque où sera rangée la table SAS qui sera crée. On a appelé cette bibliothèque lsas. Elle est construite dans le dossier "travaux SAS" qui se trouve sur le disque dur C.

  ATTENTION : Le nom de la bibliothèque ne doit pas dépasser 8 caractères, ne doit pas comporter de lettre accentuée ni de chiffres !

- ATTENTION : Les instructions sous SAS sont suivies d'un point virgule !!
- SAS ne distingue pas les minuscules et les majuscules. On peut écrire donc DATA ou data.
- L'instruction DATA permet de créer une table SAS.
  Pour créer une table SAS temporaire, on écrira : DATA nomTable.
  Pour créer une table SAS permanente, on écrira : DATA nomLIBRARY.nomTable

  La syntaxe de l'instruction DATA est :

  DATA nomTable

  …. ;

  RUN ;

  La table SAS crée aura pour extension .sas7bdat

- L'instruction INPUT déclare les variables qui composeront la table SAS. Ici les variables sont Ventes et Pub. Ce sont des variables numériques qui ne contiennent que des valeurs. Lorsque l'on souhaite déclarer des variables caractères, il faut ajouter $ après la déclaration de la variable caractère.

  Exemple :

  DATA lsas.essai ;
  INPUT Societe $ Ventes Pub ;
  CARDS ;

Dupont 164 34
AAA 198 36
LSA 85 179
Store 168 45
;
PROC PRINT;
RUN ;

ATTENTION : Le nom des variables ne doivent pas comporter de lettre accentuée !
Les nombres décimaux s'écrivent avec un point !

- L'instruction CARDS permet de rentrer manuellement les données dans la table SAS. Sa syntaxe est :

  CARDS ;
  données
  ;

- La procédure PRINT permet de visualiser les résultats dans la fenêtre Sortie.

Création d'une table SAS à partir d'un fichier de données externe :

A partir d'un fichier texte

On possède les données de Ventes et de Pub dans le fichier texte appelé essai.txt qui se trouve dans le dossier Documents sur le disque dur C. Pour transférer ces données vers la table SAS appelée essai, on tape les lignes suivantes dans la fenêtre Editeur :

LIBNAME lsas "C:\travaux SAS\" ;
DATA lsas.essai ;
INFILE "C:\Documents\essai.txt" EXPANDTABS FIRSTOBS=2 ;
INPUT Ventes Pub ;
PROC PRINT ;
RUN ;

Remarques :

- L'instruction EXPANDTABS indique à SAS que les données du fichier essai.txt sont séparées par un espace.

  Si les données sont séparées par une virgule, on écrit alors :

  INFILE "C:\ Documents \essai.txt" DLM=',' ;

- L'instruction FIRSTOBS indique à SAS de ne pas prendre en compte la 1ère ligne du fichier. Cette instruction est à utiliser lorsque la 1ère ligne du fichier comporte le nom des variables alors que SAS s'attend à trouver en 1ère ligne des valeurs numériques.

A partir d'une autre table SAS

On peut créer une table SAS (table essai) à partir d'une table SAS existante (table donnee) en tapant les lignes suivantes (ici on crée une table temporaire) :

```
DATA donnee  ;
SET essai ;
RUN ;
```

A partir d'un fichier Excel

Pour transférer des données d'un fichier Excel vers une table SAS, on peut utiliser la procédure IMPORT ou l'assistant d'importation (fenêtre "Import Wizard"). Pour faire appel à la procédure IMPORT, on écrit les lignes suivantes :

```
LIBNAME  lsas "C:\travaux SAS\";
PROC IMPORT OUT= lsas.essai
        DATAFILE= "C:\Documents\essai.xls"
        DBMS=EXCEL REPLACE;
    SHEET="Feuil1";
    GETNAMES=YES;
    MIXED=NO;
    USEDATE=YES;
    SCANTIME=YES;
RUN;
PROC PRINT DATA=lsas.essai;
RUN;
```

Remarques :

- Avant d'importer les données du fichier Excel, il faut s'assurer que les valeurs manquantes sont remplacées par un blanc, que les décimaux sont bien écrits avec une virgule (SAS ne comprend pas les décimaux écrits avec un point sous Excel) et que l'intitulé des variables figure en 1ère ligne.

  Le fichier Excel peut comporter une colonne date au format 01/01/1993 ou mar-98 par exemple.

- L'instruction OUT permet d'indiquer à SAS le nom de la table SAS ainsi que l'emplacement où l'on souhaite la placer.
- L'instruction DBMS renseigne SAS sur le format du fichier à importer.
- L'instruction REPLACE écrase une version antérieure de la table SAS portant le même nom.

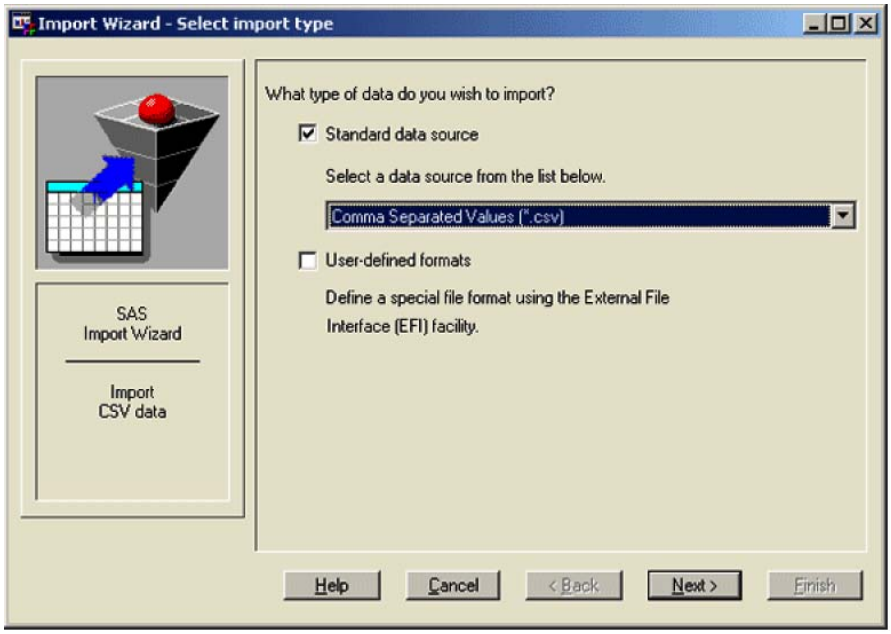
- L'instruction SHEET="Feuil1" demande à SAS d'importer les données qui se trouvent dans la première feuille Excel appelée Feuil1.
- L'instruction GETNAMES=YES permet de reprendre les intitulés des variables qui se trouvent en 1ère ligne du fichier Excel.
- L'instruction MIXED=NO informe SAS que les données sont soit des caractères soit des valeurs. Si on a des valeurs manquantes, il est souhaitable d'utiliser MIXED=YES pour pouvoir lire les données.
- L'instruction USEDATE=YES permet à SAS de lire la variable date (dans un format non défini).
- L'instruction SCANTIME=YES demande à SAS de lire la variable date tant que cette dernière ne possède pas un format date.

- Il est possible d'utiliser PROC IMPORT sans mentionner la plupart des instructions ci-dessus en écrivant simplement :

```
LIBNAME lsas "C:\travaux SAS\";
PROC IMPORT DATAFILE="C:\Documents\essai.xls" OUT=lsas.essai;
/*  S'il existe déjà une table essai, on utilise l'instruction REPLACE à la suite de OUT */
RUN ;
PROC PRINT DATA=lsas.essai;
RUN;
```

Pour faire appel à l'assistant d'importation, on va dans Fichier puis on choisit Importer données.

Remarque : L'assistant d'importation permet d'importer également des formats .txt, .cvs, …



Pour l'importation des données, on suit les étapes suivantes :

1. Dans la liste déroulante, on sélectionne le format du fichier qui contient les données à importer puis on clique sur Suivant (ou Next).

2. Une fenêtre apparaît ("Connect to MS Excel"). Dans l'encadré "Workbook", on indique en cliquant sur Browse l'emplacement du fichier Excel puis on clique sur Ouvrir et sur OK.

3. Une autre fenêtre apparaît où on demande quelle table on souhaite importer. Si le nom de la feuille Excel qui apparaît nous convient, on clique sur Suivant.

4. Une nouvelle fenêtre nous demande de renseigner le nom de la bibliothèque (Library) vers laquelle on veut transférer les données (on choisit dans la liste déroulante : SASUSER pour la création d'une table permanente ou WORK pour une table provisoire).

Dans l'encadré "Member", on attribue un nom à la table SAS qui va être crée (ici on l'appelle essai).

5. Une dernière fenêtre apparaît et on clique Terminé (ou Finish).

B/ Etape DATA : lecture d'une table SAS

La lecture d'une table SAS s'effectue à partir de l'instruction SET. Pour lire les données d'une table SAS existante dans la bibliothèque lsas, on tape dans la fenêtre Editeur les lignes suivantes :

```
LIBNAME lsas "C:\travaux SAS\";
DATA lsas.essai;
SET lsas.essai;
RUN;
```

C/ Etape DATA : modification d'une table SAS

Ajout d'une variable

Pour ajouter une variable appelée lVentes qui provient du calcul du logarithme népérien de la variable Ventes de la table SAS existante appelée essai, on écrit les lignes suivantes :

```
LIBNAME lsas "C:\travaux SAS\";
DATA lsas.essai;
SET lsas.essai;
lVentes=log(Ventes);
t=_n_;      /* création de la variable temps t */
RUN;
```

Remarques :

- Pour qu'une modification puisse être prise en compte dans la table SAS, il faut d'abord fermer la vue de la table (fenêtre ViewTable qui apparaît en allant dans la fenêtre Explorateur) avant de procéder à toute modification.

- On peut créer une nouvelle variable en effectuant des opérations sur plusieurs variables.

  Exemple :

  Benefice = Ventes – Pub ;

- Les opérations mathématiques peuvent s'effectuer à l'aide des fonctions suivantes :

  ABS(num) : donne la valeur absolue d'un nombre
  MAX(num_1, …, num_n) : donne le maximum d'une liste de nombres
  MIN(num_1, …, num_n) : donne le minimum d'une liste de nombres
  SQRT(num) : fournit la racine carrée d'un nombre
  INT(num) : donne l'entier d'un nombre
  EXP(num) : donne l'exponentiel d'un nombre
  SUM(num_1, …, num_n) : calcule la somme d'une liste de nombres
  MEAN(num_1, …, num_n) : calcule la moyenne d'une liste de nombres
  VAR(num_1, …, num_n) : détermine la variance d'une liste de nombres
  STD(num_1, …, num_n) : donne l'écart-type d'une liste de nombres

- On peut noter que les instructions de calcul se font toujours à l'intérieur de la syntaxe DATA.
- On peut écrire à la suite plusieurs instructions DATA. Par exemple :

```
LIBNAME lsas "C:\travaux SAS\";
PROC IMPORT DATAFILE="C:\travaux SAS\essai.xls" OUT=lsas.essai;
RUN ;
PROC PRINT DATA=lsas.essai;
RUN;
DATA lsas.essai;
SET lsas.essai;
lPub=log(Pub);
RUN;
DATA lsas.chgessai;    /* création d'une autre table dans la bibliothèque lsas */
SET lsas.essai;
RUN;
```

Suppression d'une variable

La suppression d'une variable est réalisée à l'aide de l'instruction DROP. Pour supprimer la variable Ventes de la table SAS appelée essai, on tape les lignes suivantes :

```
LIBNAME lsas "C:\travaux SAS\";
```

```
DATA lsas.essai;
SET lsas.essai;
DROP Ventes;
RUN;
```

Renommer une variable

Le changement de nom d'une variable s'effectue à l'aide de l'instruction RENAME. Pour renommer la variable Ventes de la table SAS appelée essai, on tape les lignes suivantes :

```
/*  1ère méthode pour renommer une variable  */
LIBNAME lsas "C:\travaux SAS\";
DATA lsas.essai;
SET lsas.essai;
RENAME Ventes=Sales;
RUN;
```

ou

```
/*  2ème méthode pour renommer une variable */
LIBNAME lsas "C:\travaux SAS\";
DATA lsas.essai;
SET lsas.essai;
Sales=Ventes;
DROP Ventes;
RUN;
```

D/ Etape DATA : exportation des données d'une table SAS

L'exportation des données peut être menée à l'aide de PROC EXPORT ou de l'assistant d'exportation (fenêtre "Export Wizard" qui apparaît en allant dans Fichier puis en choisissant Exporter données).

Création d'un fichier texte à partir d'une table SAS et de PROC EXPORT

```
LIBNAME lsas "C:\travaux SAS\" ;
PROC EXPORT DATA =lsas.essai
      OUTFILE='C:\travaux SAS\essai.txt'
      REPLACE
      DBMS=DLM ;
      DELIMITER=',' ;    /* sans cette instruction, le caractère de séparation des valeurs
est l'espace  */
RUN;
```

Création d'un fichier Excel à partir d'une table SAS et de PROC EXPORT

```
LIBNAME lsas "C:\travaux SAS\" ;
PROC EXPORT DATA=lsas.essai OUTFILE='C:\travaux SAS\essai.xls' REPLACE;
RUN;
```

ETAPE 2 : REGRESSION LINEAIRE PAR LES MCO

Pour réaliser une régression de Ventes sur Pub et sur une constante par les MCO, on peut utiliser PROC REG ou PROC AUTOREG.

PROC REG estime par les MCO les coefficients d'un modèle linéaire qui comporte des erreurs qui sont non autocorrélées (si les erreurs sont hétéroscédastiques, il est possible d'utiliser la méthode d'estimation des MCG à l'aide de l'instruction WEIGHT avec PROC REG).

PROC AUTOREG estime également par les MCO les coefficients d'un modèle linéaire avec des erreurs non autocorrélées ou autocorrélées (avec la méthode du maximum de vraisemblance et une structure autorégressive pour l'erreur).

PROC AUTOREG permet aussi la mise en œuvre de certains tests comme le test de Jarque-Bera et celui de Chow.

Il existe sous SAS d'autres procédures d'estimation pour des cas plus spécifiques : CATMOD, GENMOD, GLM, LOGISTIC, MIXED, NLIN, PROBIT, etc...

Pour effectuer une estimation à l'aide de PROC REG ou de PROC AUTOREG, on écrit les lignes suivantes dans la fenêtre Editeur :

```
PROC REG DATA=lsas.essai ;
MODEL Ventes=Pub ;
RUN ;
```

ou

```
PROC AUTOREG DATA=lsas.essai ;
MODEL Ventes=Pub ;
RUN ;
```

Remarques :

- Il n'est pas nécessaire d'inclure une constante parmi les variables explicatives car celle-ci est ajoutée automatiquement. Pour mener une régression sans constante, on écrit :

```
PROC REG DATA=lsas.essai ;
```

MODEL Ventes=Pub / noint ;

RUN ;

PROC AUTOREG DATA=lsas.essai ;

MODEL Ventes=Pub / noint ;

RUN ;

- On peut demander le listage des résidus et des valeurs estimées de la variable endogène à l'aide des commandes suivantes :

PROC REG DATA=lsas.essai ;
MODEL Ventes=Pub / r p ;
/ * r fournit les résidus et p donne les valeurs estimées de Ventes dans la fenêtre Sortie */
RUN ;

ou

PROC REG DATA=lsas.essai ;
MODEL Ventes=Pub ;
OUTPUT OUT=lsas.result r=res p=fit;
/*création d'une table result dans la bibliothèque lsas contenant les résidus et les valeurs estimées pour Ventes */
RUN ;
QUIT ; /* commande nécessaire pour pouvoir lire la table result si on ne quitte pas la session*/

ou

PROC AUTOREG DATA=lsas.essai ;
MODEL Ventes=Pub ;
OUTPUT OUT=result rm=residu p=ventesprev;
/* création d'une table result (table temporaire) dans Work contenant les résidus et les valeurs estimées pour Ventes */
RUN ;
/* on peut noter que l'on n'ajoute pas ici QUIT */

- La régression de y sur plusieurs variables explicatives x1, x2, x3 et constante s'effectue de la façon suivante :

PROC REG ;
MODEL y=x1 x2 x3 ;
RUN ;

A/ Etape PROC : régression linéaire avec des variables retardées

```
LIBNAME lsas "C:\travaux SAS\";
PROC IMPORT DATAFILE="C:\travaux SAS\essai.xls" OUT=lsas.essai REPLACE;
RUN ;
DATA lsas.essai ;
SET lsas.essai;
/*  déclaration des variables retardées dans l'étape DATA */
lagVentes=lag1(Ventes);
lagPub1=lag1(Pub);
lagPub2=lag2(Pub);
RUN;
PROC PRINT DATA=lsas.essai;
RUN;
/* régression à l'aide des variables retardées */
PROC REG DATA=lsas.essai ;
MODEL Ventes=lagVentes Pub lagPub1 / r ;
TITLE 'Régression avec des variables retardées' ;
RUN ;
```

B/ Etape PROC : régression linéaire avec une variable indicatrice (muette ou dummy)

```
LIBNAME lsas "C:\travaux SAS\";
DATA lsas.essai ;
INPUT days Ventes Pub ;
CARDS ;
01011988  164  34
01041988  198  36
01081988  85  179
01121988 168  45
;
RUN;
/*  1ère méthode pour créer une variable muette */
DATA lsas.essai;
SET lsas.essai;
DUAG88=(days=01081988);
RUN;
```

ou

```
/*  2ème méthode pour créer une variable muette */
DATA lsas.essai;
SET lsas.essai;
IF days=01081988 THEN DUAG88=1; ELSE DUAG88=0;
RUN;
```

Remarques :

- Si la variable days est une chaîne de caractères, les lignes de commandes s'écrivent :
  DATA lsas.essai;
  SET lsas.essai;
  DUAG88=(days="01081988");
  RUN;

- Si l'on souhaite faire appel au rang d'observation de la variable days pour construire la variable muette, on écrira :

  DATA lsas.essai;
  SET lsas.essai;
  DUAG88=(_n_=3); /* la date 01081988 correspond à la 3ème observation */
  RUN;

ETAPE 3 : REALISATION DES GRAPHIQUES SOUS SAS

PROC REG DATA=lsas.essai ;
MODEL Ventes=Pub / r ;
PLOT Ventes*Pub; /* Trace le graphique de Ventes en ordonnée et de Pub en abscisse avec la droite d'ajustement */
RUN ;

PROC REG DATA=lsas.essai ;
MODEL Ventes=Pub / r ;
PLOT r.*p. ; /* Trace le graphique des résidus en ordonnée et des valeurs estimées de Ventes en abscisse */
RUN ;

LIBNAME lsas "C:\travaux SAS\" ;
DATA lsas.essai;
INPUT Date Ventes Pub;
CARDS;
1 164 34
2 198 36
3 85 179
4 168 45
5 201 67
6 98 76
7 197 75
;
PROC PRINT DATA=lsas.essai;
RUN;
PROC GPLOT data=lsas.essai;
PLOT Ventes*Date; /* Trace le graphique de Ventes en fonction du temps */
RUN;

ETAPE 4 : UTILISATION DES TESTS STATISTIQUES SOUS SAS

Tests de normalité

```
PROC AUTOREG DATA=lsas.essai ;
MODEL Ventes=Pub / NORMAL ;
/*  NORMAL pour obtenir le test de Jarque Bera */
RUN ;
```

ou

```
PROC REG DATA=lsas.essai ;
MODEL Ventes=Pub;
OUTPUT OUT=result  r=res p=Ventesprev ;
RUN;
/*  Mise en oeuvre des tests de normalité de Kolmogorov-Smirnov, Cramer-Von Mises et
Andersen-Darling */
PROC UNIVARIATE DATA=result  NORMAL;
VAR res;
HISTOGRAM res / NORMAL;
RUN;
```

Test d'autocorrélation d'ordre 1 : test de Durbin-Watson

```
PROC REG ;
MODEL Ventes=Pub / DWPROB ;
RUN ;
```

ou

```
/*  PROC AUTOREG fournit directement les résultats du test de DW dans la fenêtre Sortie */
PROC AUTOREG DATA=lsas.essai ;
MODEL Ventes=Pub  ;
RUN ;
```

Test d'hétéroscédasticité de White

```
LIBNAME  lsas "C:\travaux SAS\"  ;
DATA lsas.essai  ;
INPUT Ventes Pub Charges ;
CARDS ;
164  34  16
198  36  24
85  179   8
168  45  18
201  67  32
```

98 76 10
197 75 19
;
PROC MODEL;
PARMS B0 B1 B2;
Ventes=B0+B1*Pub+B2*Charges;
FIT Ventes / WHITE;
RUN;

ou

PROC REG DATA=lsas.essai ;
MODEL Ventes=Pub Charges / SPEC;
RUN ;

Test de stabilité des coefficients : test de Chow

PROC AUTOREG DATA=lsas.essai ;
MODEL Ventes=Pub / CHOW=(3);
RUN ;

ETAPE 5 : PREVISION

LIBNAME lsas "C:\travaux SAS\" ;
DATA lsas.essai ;
INPUT Ventes Pub ;
CARDS (ou DATALINES) ;
164 34
198 36
85 179
168 45
201 67
98 76
197 75
/* Ajout des valeurs pour Pub pour la prévision de Ventes hors de l'échantillon. Les valeurs manquantes pour Ventes sont signalées à l'aide d'un point. * /
. 84
. 90
;
RUN ;
PROC REG DATA=lsas.essai ;
MODEL Ventes=Pub / p ;
/* p affiche les valeurs prévues pour Ventes hors de l'échantillon */
RUN ;