STATA 18

Statistics • Visualization • Data manipulation • Reporting

GETTING STARTED WITH STATA

日本語マニュアル

FOR MAC

Statistical software for data science

STATA ガイド Getting Started

Mac 版

リリース

18

訳ライトストーン

Translated by LightStone Corp.

A Stata Press Publication

StataCorp LLC

Version 18

Published by Stata Press, 4905 Lakeway Drive, College Station, Texas 77845
Typeset in TEX

ソフトウェアに関する引用は次のように行ってください。

StataCorp. 2023. Stata: Release 18. Statistical Software. College Station, TX: StataCorp LLC.

目次

Stata の他のマニュアル参照について

本マニュアルを読み進めて行くと、他の Stata のマニュアルを参照している箇所があります。例えば、次のように表現されます。

[U] 26 Overview of Stata estimation commands
[R] regress
[D] reshape

1 行目は User's Guide の 26 章、Overview of Stata estimation commands を参照しています。2 行目は Base Reference Manual の regress を、3 行目は Data Management Reference Manual の reshape を参照しています。上記 [U] のように Stata のマニュアルには略称が割り振られています。

[GSM]	Getting Started with Stata for Mac
[GSU]	Getting Started with Stata for Unix
[GSW]	Getting Started with Stata for Windows
[U]	Stata User's Guide
[R]	Stata Base Reference Manual
[ADAPT]	Stata Adaptive Design: Group Sequential Trials Reference Manual
[BAYES]	Stata Bayesian Analysis Reference Manual
[BMA]	Stata Bayesian Model Averaging Reference Manual
[CAUSAL]	Stata Causal Inference and Treatment Effects Estimation Reference Manual
[CM]	Stata Choice Models Reference Manual
[D]	Stata Data Management Reference Manual
[ERM]	Stata Extended Regression Models Reference Manual
[FMM]	Stata Finite Mixture Models Reference Manual
[FN]	Stata Functions Reference Manual
[G]	Stata Graphics Reference Manual
[IRT]	Stata Item Response Theory Reference Manual
[DSGE]	Stata Linearized Dynamic Stochastic General Equilibrium Reference Manual
[XT]	Stata Longitudinal-Data/Panel-Data Reference Manual
[ME]	Stata Multilevel Mixed-Effects Reference Manual
[MI]	Stata Multiple-Imputation Reference Manual
[MV]	Stata Multivariate Statistics Reference Manual
[PSS]	Stata Power and Sample-Size Reference Manual
[P]	Stata Programming Reference Manual

[SP] Stata Spatial Autoregressive Models Reference Manual

[SEM] Stata Structural Equation Modeling Reference Manual

[SVY] Stata Survey Data Reference Manual

[ST] Stata Survival Analysis Reference Manual

[TABLES] Stata Customizable Tables and Collected Results Reference

[TS] Stata Time-Series Reference Manual

[I] Stata Index

[M] Mata Reference Manual

このマニュアルについて

これは、**Mac** 版 **Stata** のマニュアルです。Windows 版 Stata をご使用の方は「Stata Getting Started Windows 版」を、Unix版 Stata をご使用の方は「Stata Getting Started Unix 版」をそれぞれご覧ください。このマニュアルは Stata 入門者または Stata の Mac 版を初めて使用する方を対象として作成しました。既存ユーザには Mac 版 Stata の新機能のチュートリアルとしてもご利用いただけます。
このマニュアルの本編が 19 章、付録が 3 章です。付録には Mac 版 Stata 専用の情報を記載しました。
ユーザ登録をした方は複数のテクニカルサポートを利用できます。 [GSM] **4 Getting Help (**ヘルプ・ヒントを見つける**)** では Stata のコマンドや機能を学ぶ手助けとなるリソースの紹介をしています。リソースの一つには Stata のウェブサイト (http://www.stata.com) があります。サイト上にはよくある質問 (FAQ) 等、多くの情報があります。同ウェブサイトおよび [GSM] **19 Updating and extending Stata—Internet functionality (Stata** のアップデートと拡張**—**インターネットでの機能**)** に記載の方法で調べても依然、不明点が残る場合、 [U] **3.8 Technical Support** を参照してください。

マニュアルを使用する

Stata 入門者はこのマニュアルを演習テキストとして、各例題を実際にコンピュータで操作しながら学ぶことをお勧めします。例題はステップ形式になっているので同じデータを複数回にわたり使用していきます。ちょうど統計学そのものに多くの手法と奥深さがあるように、Stata は奥深く豊富な統計機能を持つソフトウェアです。例題に取り組むことで統計の知識も身に付くので、実際にデータ分析を行う際に練習の効果が表れるはずです。

これは Stata 入門者向けのマニュアルですが、熟練ユーザでもこのマニュアルから学べることがあるかもしれません。熟練ユーザはまず目次を見て、何か新しいものがないか、忘れていることはないか、と項目を確認してみてください。

表記法について

Stata のユーザ設定（preferences）は一般的な macOS のアプリケーションと同様、アプリケーション（Application）メニューにあります。アプリケーションメニュー名はご利用の Stata のエディションに応じて、**Stata/MP 18.0**、**Stata/SE 18.0**、**Stata/BE 18.0** のいずれかになります。本マニュアルでは、一律に **Stata** と呼びます。すなわち、「ユーザ設定は **Stata >** ユーザ設定で見つけることができる」との記載は、最初のメニュー名は実際とわずかに異なっています。

また、多くの操作で右クリックをするように求められます。ボタンが 1 つだけのマウスの場合、

Control+ クリックが同様の操作になります。

1 Stata の紹介―サンプルセッション

1.1 Stata の紹介

この章ではサンプルのワークセッションを実際に使用しながら Stata で実行出来る基本的な操作の説明をします。データセットを開く、データセットの内容を調べる、記述統計量を使用する、グラフを作成する、そして簡単な回帰分析を行う――これらの内容を紹介します。いずれも導入的な簡単な内容です。Stata で何ができるのか、どのように動作するのかを理解する助けとなります。説明はなるべく簡潔に記してあります。必要に応じてこのマニュアルの別の箇所、もしくはシステムヘルプや他のマニュアルへのリファレンス情報を示しますので参照してください。メニューとダイアログによる操作と、コマンドによる操作を併用して説明しますので、どちらの操作も体験できます。Stata のメニュー表記が本書と異なる場合、読みやすいように本書と同じ言語に変更することをお勧めします。設定法法については [P] **set locale ui** をご覧ください。

コンピュータの前に座り、この本で勉強していきましょう。

1.2 サンプルセッション

このセッションではアメリカ国内における 1978 年の年代物の自動車販売データを利用します。

カーソルを合わせてクリックを行う操作はメニュー > メニュー内項目 > サブメニュー項目 > などのように表記します。コマンドウィンドウを使用して操作を行う場合は、(.) の後に続くコマンドを画面下部にあるコマンドと書かれた小さなウィンドウに入力してください。何かコマンドの構成の中で気を付けるべきことがある場合、“構文メモ”として記します。

では、まず auto データセットをロードしましょう。このデータセットは Stata に初めから入っています。メニューを使用して、以下のように操作します。

1. ファイル > 例題データセット... と操作します。
2. 表示されるウィンドウ内から Example datasets installed with Stata をクリックします。
3. auto.dta (リストの一番上) の横にある use をクリックします。

このコマンドの結果は、大きく次のように表れます。

- 画面中央の大きな結果ウィンドウには次のコマンドを表示します：

```
. sysuse auto
(1978 automobile data)
```

このウィンドウはコマンドとその結果を表示します。コマンドは「sysuse auto.dta」とある太字体で、ピリオド (.) の後に続きます。結果は「(1978 automobile data)」とある標準字体で、データセットの簡単な説明です。

> メモ：コマンドの意味や使用方法が知りたい場合、コマンドウィンドウに「help 半角スペース」の後に「コマンド名」をひとつ入力すると、そのコマンドに関するヘルプを表示します。また、メニューバーでヘルプ > 検索... と選択すると、さらに詳しい情報がいつでも検索できます。

- 画面右上の小さな変数ウィンドウには変数の一覧が表示されます。
- 画面右下にある、小さなプロパティウィンドウにはデータセットの 1 番目の変数、make についての情報を表示します。

コマンドウィンドウに「sysuse auto」と入力してから Return キーを押してもデータセットは開きます。これも一度試してみてください。sysuse はサンプル (システム) データセットをロード (使用) するコマンドです。このセッションで利用するように、Stata のコマンドはとても単純なのでコマンドウィンドウに直接入力して使用したほうが作業時間を短くできます。Stata を日常的に使用する場合、使用頻度の高いコマンドを覚えておくと効率的に作業できるようになります。

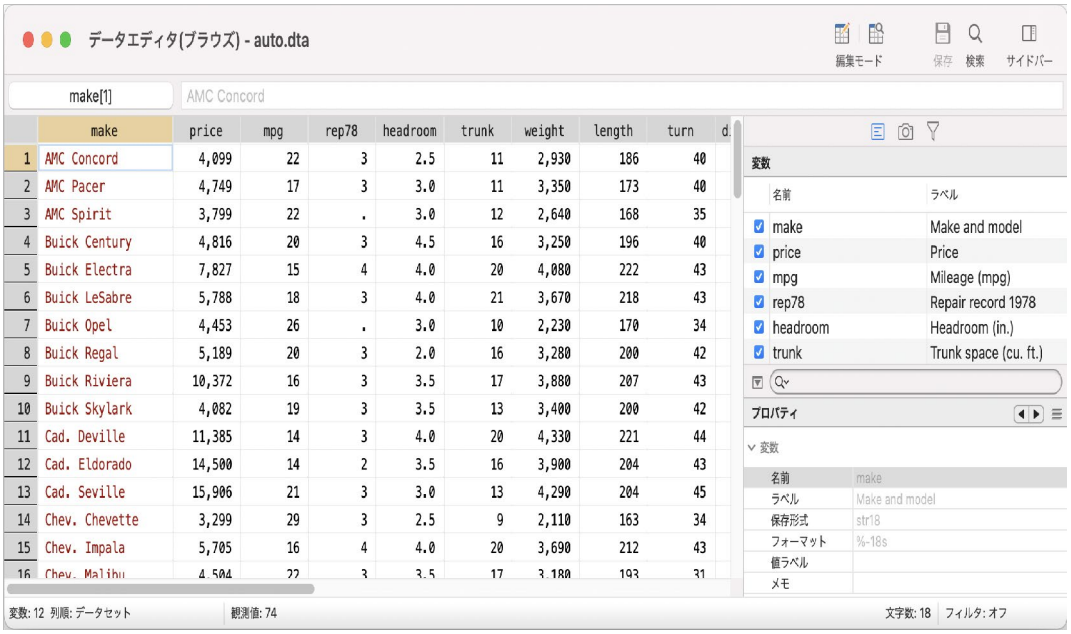
構文メモ：上記の例では sysuse が Stata のコマンドで、auto は Stata のデータファイルの名前です。

1.3 簡単なデータ管理

データセットはデータエディタで見ることができます。データエディタ (ブラウズ) ボタン  をクリックするか、データ > データエディタ > データエディタ (ブラウズ) とメニュー操作するか、あるいは「browse」コマンドをコマンドウィンドウに入力すると、データエディタが開きます。

構文メモ：データエディタボタンをクリックするだけならば、データセットや解析に影響を与えないので、新しくコマンドが出力されません。

データエディタウィンドウを開くと Stata がデータを表形式で表示します。これはすべての Stata のデータセットについて同じです。列は変数を、行は観測値 (データ) を表します。変数には分かりやすい名前が付き、観測値には番号が振られます。



データは複数の色で表示されます。一見すると黒は数値を (ダークモードでは白色)、他の色は文字を示しているようです。では、確認してみましょう。変数 make の下にあるセルを、1 つクリックします。ウィンドウ上の入力ボックス (ウィンドウ内の上部、ボタンのあるツールバーの下にある灰色のエリア) には車のメーカーが表示されます。変数 foreign が見えるまで右にスクロールし、その列のセルを 1 つクリックします。クリックしたセルには "Domestic" と表示していますが、入力ボックスには 0 が表示されます。Stata はデータ分類のカテゴリーを数字で保存します。しかし、数値のままではその意味が分かりづらいので、一目で分類内容が伝わるように文字を表示できます。これを値ラベルと呼びます。最後に変数 rep78 は数値データを表しているように見えますが、いくつかのセルはピリオド (.) だけを表示しています。このピリオドは欠損値を表します。

データエディタで見るデータは見やすいですが、データセットについての情報は限られます。データを分析するとき、何を表すデータなのか、どのように保存しているのか、という詳細が分かると便利です。データエディタを閉じて Stata のメインウィンドウに戻ります。

データセットの構造は describe コマンドで詳しく確認できます。データ > データの内容表示 > メモリ/ファイル内のデータの内容表示とメニュー選択し **OK** をクリックするか、コマンドウィンドウに「describe」と打ち込み、Return キーを押します。どちらの方法でも同じ結果を表示します。

```
. describe
Contains data from /Applications/Stata/ado/base/a/auto.dta
 Observations:            74                  1978 automobile data
    Variables:            12                  13 Apr 2022 17:45
                                              (_dta has notes)
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
make            str18   %-18s                 Make and model
price           int     %8.0gc                Price
mpg             int     %8.0g                 Mileage (mpg)
rep78           int     %8.0g                 Repair record 1978
headroom        float   %6.1f                 Headroom (in.)
trunk           int     %8.0g                 Trunk space (cu. ft.)
weight          int     %8.0gc                Weight (lbs.)
length          int     %8.0g                 Length (in.)
turn            int     %8.0g                 Turn circle (ft.)
displacement    int     %8.0g                 Displacement (cu. in.)
gear_ratio      float   %6.2f                 Gear ratio
foreign         byte    %8.0g       origin    Car origin
-------------------------------------------------------------------------------
Sorted by: foreign
```

リストの一番上にデータセット全体の情報、例えばデータの保存場所や最終保存時間を簡潔に表示します。太字の 1978 automobile data はデータセットが開かれた時に表示される簡単な説明で、Stata ではこれをデータセットのラベルと呼びます。 dta has notes の部分はデータセットにメモが添付されていることを表します。メモの内容はコマンドウィンドウに「notes」と打ち Return を押すと結果ウィンドウで確認できます。

```
. notes
_dta:
  1.  From Consumer Reports with permission
```

元データに関する簡単なメモを見ることができます。

describe コマンドによるリストを見返すと、元のデータ以外の情報を Stata が保持していることが分かります。全ての変数には次に示すフィールドが用意されています。

- variable name (変数名) には Stata で操作するためのデータの名前です。variable name は Stata が利用する name の 1 つです。詳しくは [U] **11.3 Naming conventions** をご覧ください。
- storage type (保存タイプ) はデータの保存形式です。現時点では str がつくタイプは文字列 (テキスト) 変数を表し、その他のタイプは数値であることを理解していれば十分です。このデータセットの中にはありませんが、Stata では任意の長い文字列 strL(スタール) も使用出来ます。

strL はバイナリ形式も格納できます。詳しくは [U] **12.4 Strings** をご覧ください。

- display format は表出力時の表示設定です。詳しくは [U] **12.5 Formats:Controlling how data are displayed** をご覧ください。
- value label (値ラベル) 数値と文字列との紐づけです。統計処理は数値で、表示は文字列でそれぞれ行えるようにします。値ラベルがセットされている場合のみ表示されます。詳しくは [GSM] **9 Labeling data (データのラベリング)** と [U] **12.6.3 Value labels** をご覧ください。
- variable label (変数ラベル) は変数の説明です。データ作成者以外でも変数の情報が分かるように用意されました。この変数ラベルは表作成時に使用します。

データセットにはデータのみではなく、より多くの情報を付加できます。これらの情報があればデータ作成者以外の研究者にとっても便利です。

describe コマンドはデータ構成に関する情報をユーザに提供しますが、データについてはほとんど説明しません。このデータの要約を表示するには統計 > 要約/表/検定 > 要約と記述統計量 > 記述統計量と操作し、**OK** ボタンをクリックします。あるいはコマンドウィンドウに「summarize」と打ち込み、Return を押します。結果はデータセット内すべての変数に関する記述統計量を表形式で出力します。

```
. summarize
    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
        make |          0
       price |         74    6165.257    2949.496       3291      15906
         mpg |         74     21.2973    5.785503         12         41
       rep78 |         69    3.405797    .9899323          1          5
    headroom |         74    2.993243    .8459948        1.5          5
-------------+---------------------------------------------------------
       trunk |         74    13.75676    4.277404          5         23
      weight |         74    3019.459    777.1936       1760       4840
      length |         74    187.9324    22.26634        142        233
        turn |         74    39.64865    4.399354         31         51
displacement |         74    197.2973    91.83722         79        425
-------------+---------------------------------------------------------
  gear_ratio |         74    3.014865    .4562871       2.19       3.89
     foreign |         74    .2972973    .4601885          0          1
```

この簡単な記述統計量から、データの様子が少し分かります。まず価格 (price) が現代の車とは全く異なります。アンティーク並みの古い車なので不思議ではありません。また、燃費 (mpg) も決してよくありません。自動車愛好家ならば他の細かい特徴からも性能について想像できるでしょう。

さらに重要なポイントが 2 つあります。

- 変数 make の観測値 (Obs) が 0 です。この変数は文字列 (テキスト) の変数で、数値データはありません。
- 変数 rep78 は他の数値的な観測数よりも 5 つ少なくなっています。これは rep78 に 5 つの欠損値があることを示しています。

summarize コマンドと describe コマンドを使用すれば、データセットの概要を確認できます。Stata にはデータセットをより深く、細部にわたり説明をするコマンドとして codebook があり、構成、内容、変数の値など幅広く表示します。コマンドウィンドウに「codebook」と入力して Return を押すか、メニューからデータ > データの内容表示 > コードブックの表示と選択し **OK** をクリックします。このシンプルなコマンド 1 つで多くの情報を表示します。必要に応じて結果ウィンドウをスクロールバックし、今までの出力結果も確認しましょう。これから変数 make, rep78, foreign の出力について詳しく見ていきます。

調査を始めるにあたり 1 つの変数、例えば make だけに codebook コマンドを実行します。この操作もコマンドとメニュー、どちらからでも実行できます。メニュー操作で変数を選ぶには、まずメニューからデータ > データの内容表示 > コードブックの表示と操作してダイアログを開きます。ダイアログを使用して変数 make にだけ codebook を適用する場合、次に示す 2 つの方法があります。

- 変数欄に直接「make」と入力します。

```
. codebook make

make                                                    Make and model

                  Type: String (str18), but longest is str17
        Unique values: 74                         Missing "": 0/74
             Examples: "Cad. Deville"
                       "Dodge Magnum"
                       "Merc. XR-7"
                       "Pont. Catalina"
              Warning: Variable has embedded blanks.
```

- 変数欄は直接入力の他にリストから選択もできるようになっています。欄の右端にあるドロップダウンを示す下三角形をクリックすると、データセット内にある変数のリストを表示します。このリスト変数 make を選択すると、編集エリアに make が入ります。

もっとも、コマンドウィンドウに「codebook make」と入力し、Return を押すのが一番簡単です。出力した結果は次の通りです。

出力結果の最初の列は変数名 (make) と変数ラベル (Make and Model) を表しています。変数は文字

列 (string) として保存されています。文字列は最長 17 文字（str17）ですが、18 文字（str18）で保存しているようです。全ての値がユニークなので、必要に応じて変数 make は観測値の識別子になります。識別子は複数の元データからデータセットを取りまとめる時や、データ内からエラーを抽出するのに便利です。欠損値 (missing) はありませんが、make の文字列の中にスペースがあります。変数 make が一単語 (スペースなし) の文字列変数だと想定しているなら、気を付けなければなりません。

```
. codebook foreign

foreign                                                         Car origin

                  Type: Numeric (byte)
                 Label: origin
                 Range: [0,1]                         Units: 1
         Unique values: 2                             Missing .: 0/74
            Tabulation: Freq.   Numeric  Label
                          52         0  Domestic
                          22         1  Foreign
```

構文メモ：「codebook make」コマンドは引数として *varlist* (変数リスト) を使用するコマンドの一例です。

次に変数 foreign から値ラベルについて学びましょう。この変数のコードブック出力を確認します。コマンドウィンドウにコマンドを入力する方が簡単なので「codebook foreign」と入力します。(以降、「Return キーを押す」という記述は省略します。) 次のような出力結果になります。

出力さ れた表から、変数 foreign の値は 0 と 1 だけなのでダミー変数だと分かります。変数には値ラベルがあり、0 の時には“Domestic”、1 の時は“Foreign”と数字の代わりに表示します。このデータ表示形式の利点は 2 つあります。

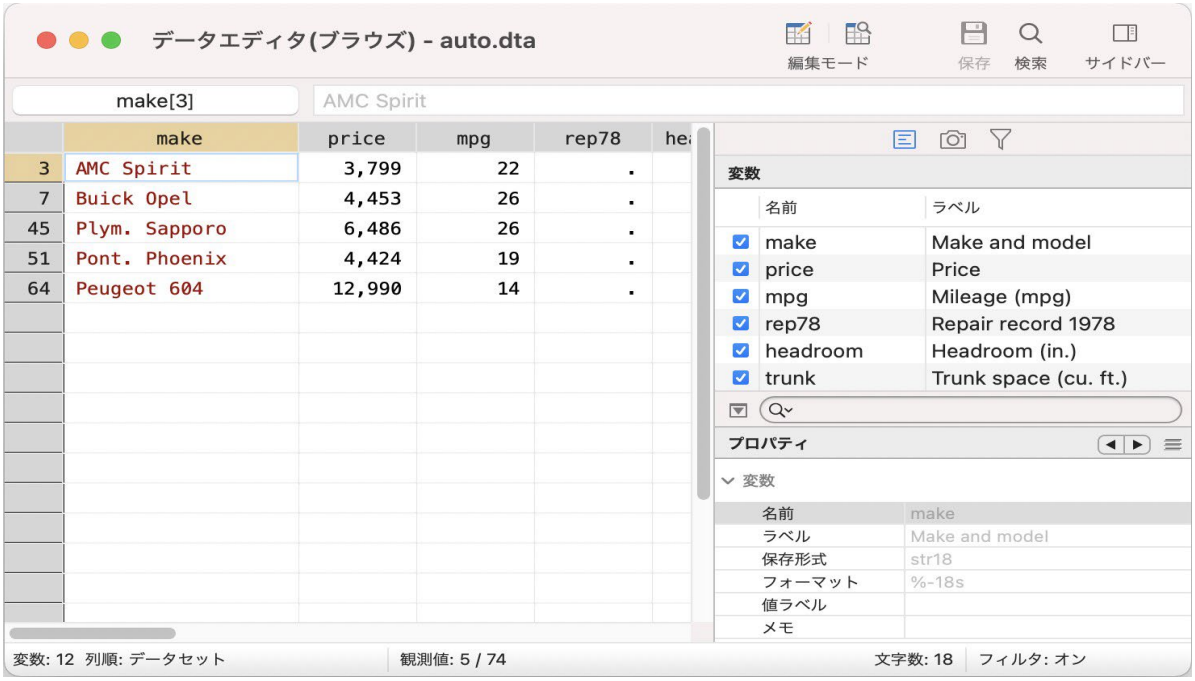
```
. codebook rep78

rep78                                                    Repair record 1978

                  Type: Numeric (int)
                 Range: [1,5]                         Units: 1
         Unique values: 5                             Missing .: 5/74
            Tabulation: Freq.  Value
                          2  1
                          8  2
                         30  3
                         18  4
                         11  5
                          5  .
```

- 変数が使用するメモリ量を減らします。数値の場合、容量は 1 バイトのみですが、文字列 “Domestic”の場合 8 バイトになります。詳しくは [U] **12.2.2 Numeric storage types** をご覧ください。
- ダミー変数として統計モデルに組み込むことができます。詳しくは [U] **25 Working with categorical data and factor variables** をご覧ください。

最後にラベル付けが不十分で、欠損値がある例を変数 rep78 から見ていきましょう。コマンドウィンドウに「codebook rep78」を入力し、実行すると次のようになります。

rep78 はカテゴリー変数のようです。しかしデータにはこれ以上の説明がないので、カテゴリー分けした数字が何を意味するのか分かりません。(値にラベルを付けるには [GSM] **6 Using the Data Editor (データエディタを使用する)** の「データを変更する」と [GSM] **9 Labeling data (データのラベリング)** をご覧ください。) この変数には欠損値が 5 つあります。これは 5 つの車種の修理記録 (repair record) が存在しないことを示します。データエディタを使用してこれらの 5 つの観測値を詳しく確認します。データエディタ (ブラウズ) をクリックするのと同様のコマンドは browse です。変数 rep78 では欠損値だけを確認したいので、次のようにコマンドウィンドウに入力します。

```
. browse if missing(rep78)
```



表示されたデータエディタを見ると、「. 」の値は欠損値であることが分かります。他の変数にも欠損値があっても問題はありません。「. 」は数値欠損値のデフォルトの表示形式です。また、Stata

では「.a」から「.z」までのユーザ欠損値を設定できますが、このデータセットの中にはありません。詳しくは [U] **12.2.1 Missing values** をご覧ください。確認が終了したら、ウィンドウ左上の x ボタンをクリックしてデータブラウザを閉じます。

構文メモ：上記のように if コマンドを使用すると観測値 (データ) のサブセットを表示します。

データを一通り確認してもなぜ特定の値が欠損しているのか分かりません。この場合、データの出典元に初めから数値が無い可能性と、誤って数値を省いた可能性を確認します。変数 make の値はユニークなので修理記録に欠損値がある車の情報をリストすれば情報の有無を確認できます。メニューおよびダイアログで操作します。

1. データ > データの内容表示 > データの一覧表示と選択します。
2. 変数欄の右端にある下三角形をクリックして変数名を表示します。
3. その中から make を選んで変数欄に入力します。
4. ダイアログ内の **by/if/in** タブをクリックします。
5. missing(rep78)を条件式ボックスに打ち込みます。
6. 適用をクリックします。すると、ダイアログは開いたままでコマンドを実行します。コマンドを試すとき、調べるとき、そして複雑なものを作成するとき等に適用ボタンはとても便利です。このサンプルでは基本的に適用を使用します。ここで **OK** を押してダイアログを閉じても構いません。

1.4 記述統計量

コマンドウィンドウに「list make if missing(rep78)」と入力しても上記メニュー操作と同じ結果になります。list コマンドは観測値（データ）のリストを作るものであり、コマンド入力の方が簡単です。出力結果を次に示します。

```
. list make if missing(rep78)

     +-------------+
     |        make |
     |-------------|
  3. | AMC Spirit  |
  7. | Buick Opel  |
 45. | Plym. Sapporo |
 51. | Pont. Phoenix |
 64. | Peugeot 604 |
     +-------------+
```

データの出典元にはこれ以上の情報が無く、この欠損値をなくすことはできません。詳しくは[GSM] **10 Listing data and basic command syntax** で list コマンドの機能をご覧ください。

構文メモ：このコマンド (if 条件と missing()関数) は私たちに 2 つの新しいコンセプトを提供します。if 条件は if 以下の条件に当てはまる観測値にのみコマンドを実行します。詳しくは [U] **11.1.3 if exp** をご覧ください。missing()関数は各観測値に欠損値があるかどうかを調べます。詳細は [FN] **Programming functions** をご覧ください。

では、データセットそのものが分かってきたのでデータ自体の調査に移りたいと思います。

1.4 記述統計量

前のセクションから、summarize コマンドは簡単な要約統計情報をすべての変数について出力することが分かりました。データの要約統計量を見たところ、車の価格であるにもかかわらず、価格がとても安いことが気になります (1978 年なので安いのは当たり前ですが)。この変数 price をより詳しく調べる為、以下のように操作します。

1. 統計 > 要約/表/検定 > 要約と記述統計量 > 記述統計量を選択します。
2. 変数欄に直接 price と入力するか、右の下三角形のリストから選びます。
3. オプション内の追加の統計量を表示するのラジオボタンを選択します。
4. 適用をクリックします。

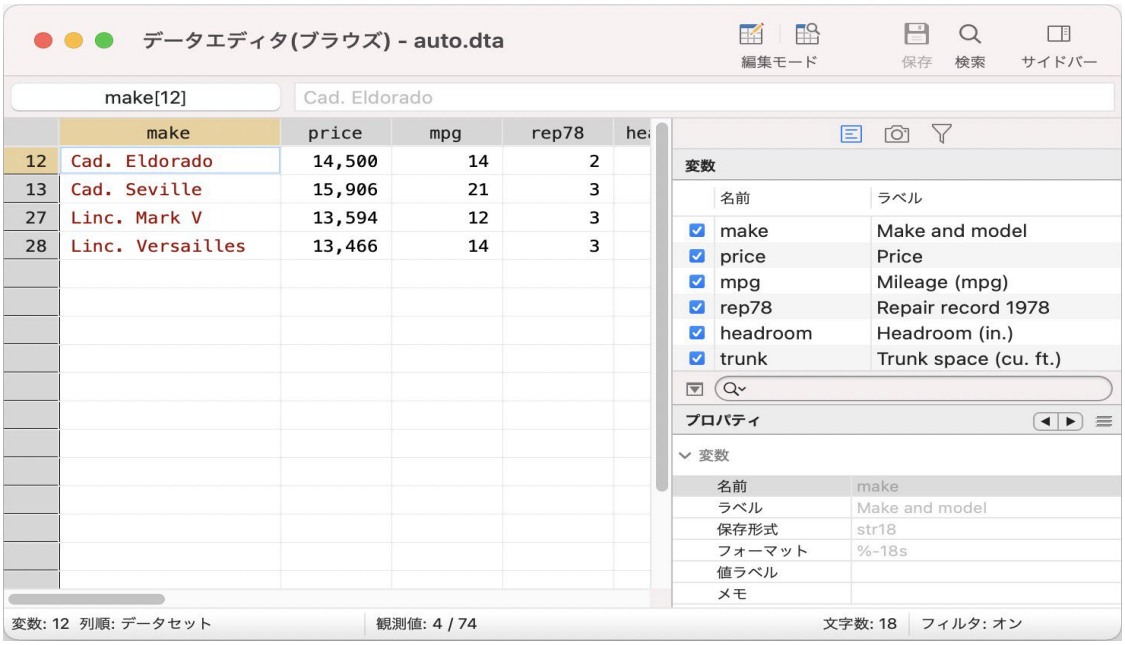
構文メモ：結果ウィンドウからも分かるように、「summarize price, detail」とコマンドウィンドウに入力しても結果は同じです。カンマの後の部分は Stata コマンドではオプションを表します。つまり、以下の構文では detail はオプションの例となります。

```
. summarize price, detail
                            Price
-------------------------------------------------------------
      Percentiles      Smallest
 1%         3291            3291
 5%         3748            3299
10%         3895            3667       Obs                  74
25%         4195            3748       Sum of wgt.          74

50%       5006.5                       Mean           6165.257
                          Largest       Std. dev.      2949.496
75%         6342           13466
90%        11385           13594       Variance       8699526
95%        13466           14500       Skewness       1.653434
99%        15906           15906       Kurtosis       4.819188
```

出力結果から、このデータセット内の車の値段の中央値はわずか$5,006.50 だと分かります。そして高価な車 4 台はすべて$13,400 から$16,000 の範囲にあります。この最も高価な価格帯にある車を詳しく調べるには (そしてデータエディタを少し使うには) まずデータブラウザボタン  を押します。

データエディタが開いたら観測値フィルタボタン  を押すと観測値フィルタウィンドウが出てきます。式によるフィルタ欄に「price > 13000」と打ち込むと$13,000 より高い車のみを表示します。

フィルタを適用するボタンを押すと最高価格帯にある 4 台の車が表示され、2 つは Cadillac 車 (変数 make の前半が Cad.) で残りの 2 つは Lincoln 車 (変数 make の前半が Linc.) です。この 4 台は決して燃費が良い車ではありません。



先ほどデータの内容を簡単に確認した時、外国製の車の修理記録の方が良かったようなので、これから外国製の車と修理記録の関係について調べようと思います。(ここで、カテゴリー 1、2、3、4、5 が何を意味するのか分かりませんが、Chevy の Monza (カテゴリー 2) は壊れやすいと評判でした。) では、データセット内の外国製の車の割合と、各修理記録の割合を見てみましょう。これは一元表 (one-way table) で確認できます。外国製の車に関する表を作成するには次のように操作します。統計 > 要約/表/検定 > 度数分布表 > 一元配置表と選択しカテゴリ変数欄でドロップダウンリストから変数 foreign を選択します。適用を押すと次の結果を表示します。

```
. tabulate foreign
Car origin |      Freq.     Percent        Cum.
------------+-----------------------------------
   Domestic |         52       70.27       70.27
    Foreign |         22       29.73      100.00
------------+-----------------------------------
      Total |         74      100.00
```

この結果からデータセット内の約 70% は国内製 (domestic) すなわちアメリカ製で、30% は外国製 (foreign) だと分かります。この表の Car type 欄では 0 と 1 の数値ではなく、見やすくなるように値ラベルを使用しています。

構文メモ：結果ウィンドウから、この一元表は tabulate コマンドの後に変数名 foreign を加えることでも作成できます。修理記録 (rep78) の一元表を作成するにはコマンドウィンドウに「tabulate rep78」と入力しましょう。次のように、カテゴリー別に表示されます。

```
. tabulate rep78
```

Repair record 1978	Freq.	Percent	Cum.
1	2	2.90	2.90
2	8	11.59	14.49
3	30	43.48	57.97
4	18	26.09	84.06
5	11	15.94	100.00
Total	69	100.00	

このカテゴリー“3”が何を意味するのかは分かりません。しかし、ほとんどの車は 3 以上のカテゴ リーに入っています。おそらく、カテゴリー 1 は最も悪い (修理記録の) 評価を、5 は良い評価を表しているのでしょう。この推測を元にデータセットの説明を続けていきます。度数 (Freq.) が 74 では無く 69 なので 5 つの欠損値の存在が確認できます。

外国製と国内製の修理記録を比較するには 2 つの一元表よりは、むしろ 1 つの二元表のほうが適しているのでそれを作成します。メニューで次のような操作をします。

1. 統計 > 要約/表/検定 > 度数分布表 > 二元配置表/関連係数を選択します。
2. 行の変数にドロップダウンリストから rep78 を選びます。
3. 列の変数にも同じように foreign を選びます。
4. 変数 foreign 内にはパーセント表示があるほうが良いのでセルの内容の行内の相対度数にチェックを付けます。
5. 適用をクリックします。

出力結果は次のようになります。

```
. tabulate rep78 foreign, row
```

Key
frequency *row percentage*

Repair record 1978	Car origin Domestic	Foreign	Total
1	2 100.00	0 0.00	2 100.00
2	8 100.00	0 0.00	8 100.00
3	27 90.00	3 10.00	30 100.00
4	9 50.00	9 50.00	18 100.00
5	2 18.18	9 81.82	11 100.00
Total	48 69.57	21 30.43	69 100.00

出力結果から、修理記録では外国製の車の方が国内製の物よりも全般的に良いことが分かります。ダイアログには他の仮説検定のコマンドがありますが、この場では省きます。

構文メモ：結果ウィンドウの表示から「tabulate rep78 foreign, row」をコマンドウィンドウに打ち込めば同じ表が出力できます。つまり、tabulate コマンドの後に変数を 2 つ入力すると二元表を作成します。row がオプションとしてあるのは、ダイアログで「行内の相対度数」を選択したからです。

row オプションを使用することで tabulate コマンドをデフォルトから変更できます。

次に外国製と国内製の燃費を比較したいと思います。それぞれの記述統計量を見ることから始めましょう。if 条件を使用し、変数 mpg を foreign で分けてから summarize コマンドを実行します。

```
. summarize mpg if foreign==0
```

Variable	Obs	Mean	Std. dev.	Min	Max
mpg	52	19.82692	4.743297	12	34

```
. summarize mpg if foreign==1
```

Variable	Obs	Mean	Std. dev.	Min	Max
mpg	22	24.77273	6.611187	14	41

結果から外国製の車の方が燃費は良いようです。この点については後ほど検定を行います。

構文メモ：相等性の検定には等号 2 個 (==) が必要です。等号 2 個はプログラミングを行った経験がある方は見覚えがあるかもしれません。等号 2 個を使う構文は、Stata 初心者によく見られるエラー原因の 1 つなので気を付けてください。相等性を“完全に等しい” (だから、等号 2 個で強調している) として考えるとタイピングのミスは少なくなります。

記述統計量を出力するには他に 2 つの方法があります。こちらのほうが操作としては簡単です。1 つ目の方法は今説明した方法を 1 回の操作で行います。2 つのサブセット (Domestic と Foreign) にそれぞれコマンドを実行します。メニューでは以下の手順で操作します。

1. 統計 > 要約/表/検定 > 要約と記述統計量 > 記述統計量を選択して、リセットボタン  を押します。
2. 変数欄のドロップダウンリストから mpg を選びます。
3. (未選択ならば) オプション内の標準の表示を選択します。
4. **by/if/in** タブをクリックします。
5. グループごとにコマンドを実行するのチェックボックスにチェックを付けます。
6. グループ変数欄にリストから foreign 選ぶか、直接入力します。
7. 適用をクリックします。

先程の表と一致する結果が出力されます。この方法は、数値ではなく値ラベル (Domestic と Foreign) が使われているため、上記 2 つのコマンドより見やすくなっています。グループを分類する変数の値を考える必要なく表が作成できます。

```
. by foreign, sort: summarize mpg
-------------------------------------------------------------------------------
-> foreign = Domestic
    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
         mpg |         52    19.82692    4.743297        12         34
-------------------------------------------------------------------------------
-> foreign = Foreign
    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
         mpg |         22    24.77273    6.611187        14         41
```

構文メモ： この相等性に関するコマンドはこれまでのコマンドとは少し異なります。この構文には by プレフィックスという前置コマンドが含まれます。by プレフィックスは独自のオプションとして主に「sort」があり、類似するデータを隣り合わせた状態で概要にまとめることができます。この by プレフィックスはデータ操作の理解とサブ集団 (subpopulation) で作業する際に大切なポイントになります。必要であればメモを補い、詳細の確認は [U] **11.1.2 by varlist** と [U] **13.7 The by construct** をご覧く

ださい。Stata には他にもコマンドに特殊効果を付与する前置コマンドがあります。詳しくは[U] **11.1.10 Prefix commands** をご覧ください。

生産地ごとの車の燃費を表にして比較します。つまり、変数 foreign の一元表 (foreign 対 domestic) の中に燃費の記述統計量を組み込みます。メニューから統計 > 要約/表/検定 > 度数分布以外の表 > 平均/標準偏差/度数と操作し、ダイアログの変数 **1** に foreign を、要約を表示する変数に mpg を入力します。そして適用をクリックすると以下のような表を出力します。

```
. tabulate foreign, summarize(mpg)
```

Car origin	Summary of Mileage (mpg) Mean	Std. dev.	Freq.
Domestic	19.826923	4.7432972	52
Foreign	24.772727	6.6111869	22
Total	21.297297	5.7855032	74

構文メモ：これは一元表なので、tabulate コマンドは変数を 1 つだけ使用します。記述統計量を求める変数は tabulate コマンドのオプションとして入力します。ここでは行いませんが、summarize() オプションを使用して二元表も作成できます。

1.5 簡単な仮説検定

では、Foreign と Domestic の平均燃費の差について仮説検定を行いましょう。メニューでは、統計 > 要約/表/検定 > 伝統的な仮説検定 > **t** 検定 **(**平均比較検定**)** と選択し、ダイアログを開きます。グループ別の二標本ラジオボタンを選択してから、変数名欄に mpg を、グループ変数名欄に foreign を入力し、適用をクリックします。*t* 検定を実行して仮説検定の表を表示します。

```
. ttest mpg, by(foreign)
Two-sample t test with equal variances
```

Group	Obs	Mean	Std. err.	Std. dev.	[95% conf.	interval]
Domestic	52	19.82692	.657777	4.743297	18.50638	21.14747
Foreign	22	24.77273	1.40951	6.611187	21.84149	27.70396
Combined	74	21.2973	.6725511	5.785503	19.9569	22.63769
diff		-4.945804	1.362162		-7.661225	-2.230384

```
    diff = mean(Domestic) - mean(Foreign)                     t =  -3.6308
H0: diff = 0                     Degrees of freedom =       72
    Ha: diff < 0                 Ha: diff != 0                 Ha: diff > 0
 Pr(T < t) = 0.0003         Pr(|T| > |t|) = 0.0005          Pr(T > t) = 0.9997
```

結果の表から、外国製の車の平均燃費と国内製の車の平均燃費は異なると結論付けることができます。本来ならデータ分析を始める前にこの検定を行うほうが良いでしょう。「ttest mpg, by(foreign)」コマンドは簡単なので覚えておくと便利です。不均一な分散の場合は異なる *t* 値を求めるオプションや自由度の近似計算を行うオプションがありますのでご自由にお試しください。

構文メモ：by()オプションは先程使用した by プレフィックスとは異なるものです。似たようなコンセプトを使用していますが、用法が違います。by()オプションは t 検定の専用オプションです。

1.6 記述統計量–相関行列

ここで少し話題を変え、カテゴリー間の関係から数値間の関係に焦点をあてます。例えば燃費と車重に相関があるか調べてみます。メニューから、統計 > 要約/表/検定 > 要約と記述統計量 > 相関と共分散を選択します。mpg と weight を入力またはリストから選択し、適用をクリックします。結果ウィンドウに mpg と weight の相関行列を表示します。

```
. correlate mpg weight
(obs=74)
             |      mpg   weight
-------------+------------------
         mpg |   1.0000
      weight |  -0.8072   1.0000
```

コマンド入力の場合は「correlate mpg weight」です。相関は負の相関を示しています。これは重い車ほど多くの力を必要とするので、納得できる結果です。

では国内製と外国製の車で燃費と車重の相関を比較するために、今までに学んだ by プレフィックスの知識を使います。**correlate** ダイアログをアクティブにします。閉じた場合は先程と同じようにダイアログを開きます。**by/if/in** タブをクリックし、グループごとにコマンドを実行するのチェックボックスにチェックをつけ、グループ変数に foreign を入力します。適用を押すと Domestic と Foreign に分かれた相関を表示します。記述統計量セクションで使用した「by foreign, sort:」を「correlate mpg weight」コマンドの前に入力しても同じものが出力されます。

```
. by foreign, sort: correlate mpg weight
-------------------------------------------------------------------------------
-> foreign = Domestic
(obs=52)
             |      mpg   weight
-------------+------------------
         mpg |   1.0000
      weight |  -0.8759   1.0000

-------------------------------------------------------------------------------
-> foreign = Foreign
(obs=22)
             |      mpg   weight
-------------+------------------
         mpg |   1.0000
      weight |  -0.6829   1.0000
```

結果の表より、国内製(Domestic) の方が強い相関があることが分かります。

構文メモ：この例では correlate コマンドを使用して 2 つの変数の相関を確認しました。Stata は任意の変数の数で相関行列を作成します。例えば、5 つの変数を使用すると以下のような出力になります。

```
. correlate mpg weight length turn displacement
(obs=74)
             |      mpg   weight   length     turn displa~t
-------------+---------------------------------------------
         mpg |   1.0000
      weight |  -0.8072   1.0000
      length |  -0.7958   0.9460   1.0000
        turn |  -0.7192   0.8574   0.8643   1.0000
displacement |  -0.7056   0.8949   0.8351   0.7768   1.0000
```

これは、予測変数間の共線性などを確認する際に便利です。

1.7 データの作図

今までの作業から分かったことがいくつかあります。まず国内製と外国製の車では平均燃費 (MPG) が異なります。修理回数記録も異なることが分かりました。最後に燃費と車重で負の相関を予想通り見つけ、国内製の方がより強く相関していました。

これから回帰モデル作成を見据えて燃費 (MPG) と車重 (weight) について確認していきます。まずは相関グラフを作図しましょう。mpg 対 weight の散布図から始めます。コマンドを使用して作図するには単純に「scatter mpg weight」と入力します。グラフをカスタマイズする場合はメニューを使って次のように操作します。

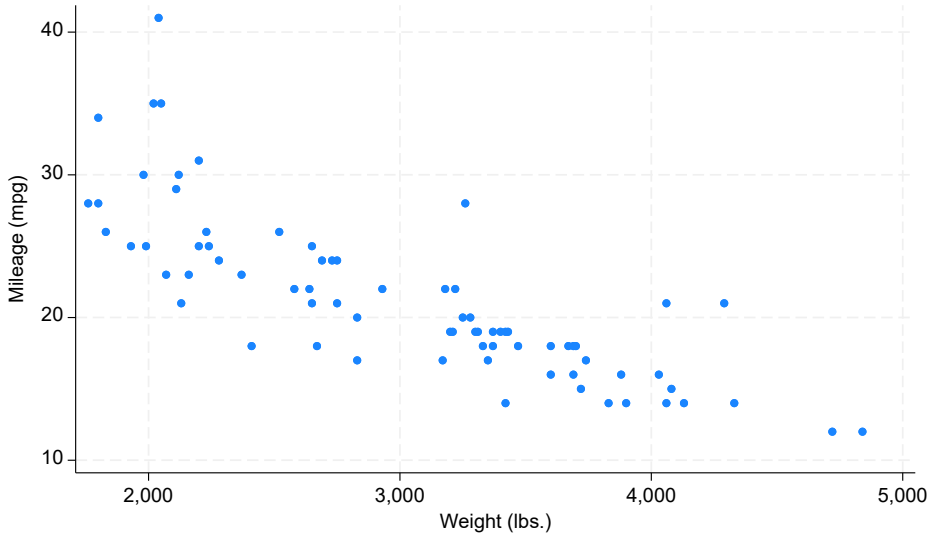
1. グラフィックス > 二元グラフ (散布図/折れ線など) を選択します。

2. 作成**...** ボタンをクリックします。
3. プロットカテゴリとタイプを選択するの枠にある、基本的なグラフのラジオボタンを選択します。(未選択の場合)
4. 基本的なグラフ**: (**タイプを選択**)** の中からマーカー（散布図）を選択します。 (未選択の場合)
5. プロットタイプ**: (**散布図**)** 枠の **y** 変数に mpg を、**x** 変数に weight をそれぞれドロップダウンリストから選択します。
6. 適用ボタンをクリックします。

メニューで実行した操作のコマンドを結果ウィンドウに表示します。

```
. twoway (scatter mpg weight)
```

実行したコマンドは初めに紹介したコマンドより少し複雑です。複雑なのには理由があり、複雑なコ マンドのほうがグラフの統合やグラフの重ね合わせも行えるからです。これから実際に操作する中で確認してください。それでは作成したグラフを見てみましょう。



グラフから、mpg と weight には非線形かつ負の相関 (右下がりの分布) があると分かります。

メモ：グラフを作図すると、結果ウィンドウの上に Graph ウィンドウが表示されます。Stata のメインウィンドウをクリックすると結果ウィンドウを最前面に配置します。グラフをもう一度確認したい場合は、グラフウィンドウを前面にボタン をクリックすると、再び Graph ウィンドウが最前面になります。グラフウィンドウを前面にボタンについての詳細は [GSM] **14 Graphing data (**データを作図する**)** をご覧ください。

国内製と外国製、それぞれの相関関係がどのように異なるのか散布図で見てみましょう。それぞれのカテゴリーの散布図と全体の散布図を同時に表示します。

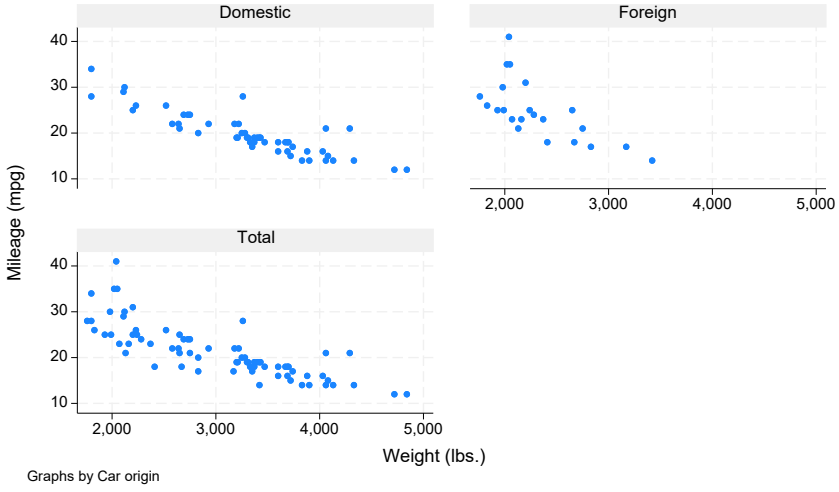
構文メモ：現在サブグループを見ているので、by プレフィックスで作図できそうです。実際に試してみましょう。

先程と同じように操作します。

1. グラフィックス > 二元グラフ (散布図/折れ線など) をメニューから選択します。
2. プロット 1 のダイアログ (先程グラフを作成したダイアログ) がまだ開いている場合、OK ボタンをクリックしてステップ 4 から操作してください。
3. 前のページに示した手順にしたがいグラフを作成します。
4. twoway - 二元グラフダイアログにある by 条件タブをクリックします。
5. 変数のユニーク値ごとのサブグラフを作成するのチェックボックスにチェックを付けます。
6. 変数欄に foreign を入力します。
7. 合計を含むグラフを追加するのチェックボックスにチェックを付けます。
8. 適用ボタンをクリックします。

作成したコマンドとグラフは次の通りです。

```
. twoway (scatter mpg weight), by(foreign, total)
```



どちらのカテゴリーも非線形な関係が成り立っている事が分かります。

構文メモ：サブグループごとのグラフを統合するとき (統合グラフ)、by プレフィックスではなく by() オプションを使用しました。by プレフィックスを使用すると、統合グラフではなく別々のグラフを作成します。

1.8 フィットモデル：線形回帰

グラフで特徴をつかんだので、車重とカテゴリーで燃費を予測する回帰モデルを作成します。変数の関係は非線形だと分かります。よって、車重の二次式として燃費をモデリングしてみます。Domestic と Foreign からは燃費と車重の関係が若干異なることが分かります。ダミー変数として foreign を加え、後でこの変数が正しく違いを表しているのか確認します。では、次のモデルをフィットしてみましょう。

$$\text{mpg} = \beta_0 + \beta_1 \text{weight} + \beta_2 \text{weight}^2 + \beta_3 \text{foreign} + \epsilon$$

foreign は既にダミー変数（0 か 1）ですが、weight の二乗値を作成する必要があります。メニュー操作でも新しい変数を作成できますが、コマンド入力の方が簡単です。次のようにコマンドウィンドウに入力しましょう。

```
. generate wtsq = weight^2
```

必要な変数がすべて揃ったので、線形回帰を行います。メニュー操作で実行し、後でコマンドを確認しましょう。統計 > 線形モデル他 > 線形回帰をメニューから選びます。ダイアログの従属変数欄に mpg を、独立変数欄に weight, wtsq, foreign を入力し、適用をクリックします。regress のコマンド文と、分散分析表が表示されます。

```
. regress mpg weight wtsq foreign
      Source |       SS       df       MS              Number of obs =      74
-------------+------------------------------           F(3, 70)      =   52.25
       Model |  1689.15372     3  563.05124             Prob > F      =  0.0000
    Residual |   754.30574    70  10.7757963            R-squared     =  0.6913
-------------+------------------------------           Adj R-squared =  0.6781
       Total |  2443.45946    73  33.4720474            Root MSE      =  3.2827

------------------------------------------------------------------------------
         mpg |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      weight |  -.0165729   .0039692    -4.18   0.000    -.0244892   -.0086567
        wtsq |   1.59e-06   6.25e-07     2.55   0.013     3.45e-07    2.84e-06
     foreign |    -2.2035   1.059246    -2.08   0.041      -4.3161   -.0909002
       _cons |   56.53884   6.197383     9.12   0.000     44.17855    68.89913
------------------------------------------------------------------------------
```

出力に問題は無さそうなので、車のカテゴリーごとの散布図上に予測値を作図しましょう。そのためには予測、またはフィットした値が必要です。これはメニュー操作で実行できますが、簡単なのでコマンドウィンドウに入力しましょう。まずは予測値を格納する新規変数、mpghat を作りましょう。次のコマンドを入力します。

```
. predict mpghat
(option xb assumed; fitted values)
```

このコマンドを入力するとメッセージが　　1 行だけ表示されます。画面右上にある変数ウィンドウを下までスクロールすると変数 mpghat が確認できます。mpghat が作成された状態でこのコマンドをもう一度実行しても、既存データは上書きされません。

```
. predict mpghat
variable mpghat already defined
r(110);
```

predict のように、回帰実行後に続けて利用するコマンドのことをポスト推定（**postestimation**）コマンドと呼びます。新しい変数 mpghat には次の数式で計算した値が入ります。

$$-0.0165729\text{weight} + 1.59 \times 10^{-6}\text{wtsq} - 2.2035\text{foreign} + 56.53884$$

モデルの推定後には予測値の計算はもちろん、モデルを改良するための様々な機能が利用できるようになります。詳しくは **[U] 20 Estimation and postestimation commands** をご覧ください。

では、国内製と外国製のグラフの上に予測値を書き込んでダミー変数の適切さを確認しましょう。データと予測曲線を同じグラフ上に作図し、適切なシフトパラメータの判断をします。両方のグラフを 1 度に作図できるので、実際にやってみましょう。メニューとダイアログを使うには以下の手順で操作します。

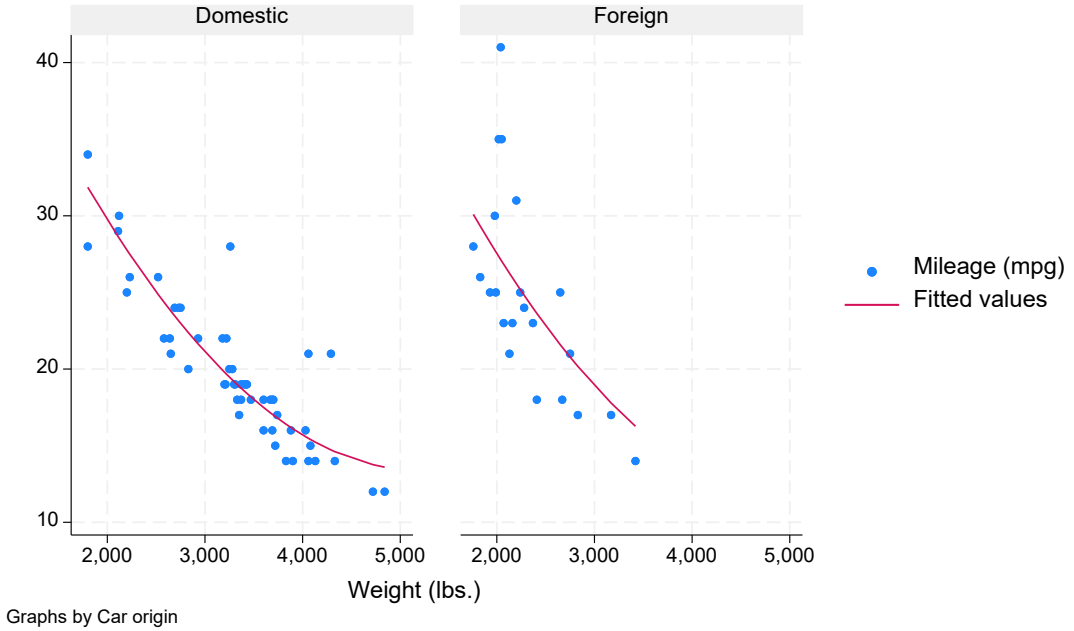
1. グラフィックス > 二元グラフ **(**散布図/折れ線など**)** をメニューから選択します。
2. プロットの定義に他のプロットが残っている場合、リセットボタン  をクリックします。
3. mpg 対 weight のグラフを作成します：
   (a) 作成**...** ボタンをクリックし、プロット **1** ダイアログを開きます。
   (b) 基本的なグラフと散布図が選択されていることを確認します。
   (c) プロットタイプの **y** 変数に mpg を、**x** 変数に weight をそれぞれ選択します。
   (d) **OK** をクリックします。
4. mpghat 対 weight のグラフを作成します：
   (a) 作成**...** ボタンをクリックします。
   (b) 基本的なグラフと線を選択します。
   (c) プロットタイプの **y** 変数に mpghat を、**x** 変数に weight を選択します。
   (d) **x** 変数でソートのチェックボックスをチェックします。これでデータ内の順番ではなく、weight の昇順で直線を作成します。
   (e) **OK** をクリックします。
5. 2 つのプロット、国内製と外国製を同じグラフ内に表示します：
   (a) **by** 条件タブをクリックします。
   (b) 変数のユニーク値ごとのサブグラフを作成するのチェックボックスにチェックを付けます。

(c) 変数欄に foreign を入力します。

6. 適用ボタンをクリックします。

コマンド文とグラフは次のようになります。

```
. twoway (scatter mpg weight) (line mpghat weight, sort), by(foreign)
```

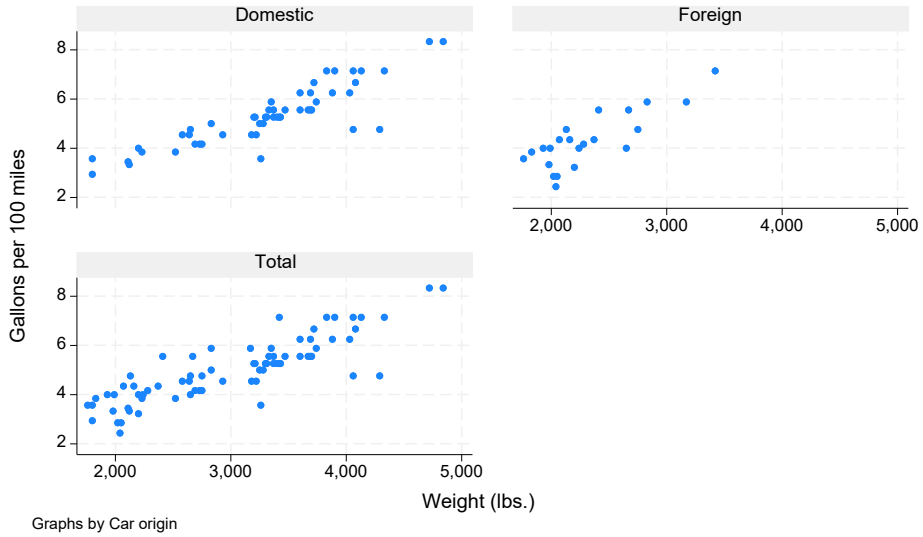


この例から、scatter と line のコマンドを別々のカッコに入れて同時に実行すると、重ね書きできることが分かります。このように、散布図と曲線を重ねる場合はカッコを使用してください。このグラフは良くフィットしています。良いグラフができたので、技術者である友人にこのグラフを見てもらうことにしましょう。今までの結果ではなくグラフだけを印刷するには、Graph ウィンドウ内でファイル > 印刷 > **Graph ―Graph** と操作して印刷します。

印刷したグラフを友人に見せたところ、どうやら間違いがあるようです。「違う。」と言われてしまいました。「2,000 ポンド (約 900kg) を 1 マイル (約 1.6km) 動かすのには 1,000 ポンド (約 450kg) を同じ距離動かすときの約 2 倍のエネルギーが必要。つまり、ガソリンの消費量も比例し、2 倍の量を消費するはず。mile/gallon は重さの二次式でなく、一次式になるはず。」とのことです。

友人が言ったことを検証してみましょう。まずは単位距離毎のエネルギー（gallon/mile）変数を作成し、散布図を作図します。以下が必要なコマンドです。このコマンドはセッション内で使用したものに似ています。この中に初めて見るコマンドが 1 つあります。「label variable」コマンドは変数に gp100m 変数ラベルを設定します。結果として次のようなグラフを作成します。

```
. generate gp100m = 100/mpg
. label variable gp100m "Gallons per 100 miles"
. twoway (scatter gp100m weight), by(foreign, total)
```



友人が正しいという結論が出たところで、回帰をやり直してみましょう。

```
. regress gp100m weight foreign
      Source |       SS           df       MS      Number of obs   =        74
-------------+----------------------------------   F(2, 71)        =    113.97
       Model |  91.1761744         2  45.5880872   Prob > F        =    0.0000
    Residual |  28.4000936        71  .400001319   R-squared       =    0.7625
-------------+----------------------------------   Adj R-squared   =    0.7558
       Total |  119.576268        73  1.63803107   Root MSE        =    .63246

------------------------------------------------------------------------------
      gp100m |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      weight |   .0016254   .0001183    13.74   0.000     .0013896    .0018612
     foreign |   .6220535   .1997381     3.11   0.003      .223787    1.02032
       _cons |  -.0734839   .4019932    -0.18   0.855    -.8750355    .7280677
------------------------------------------------------------------------------
```

記述統計量のセクションで 1978 年代の外国製の車の燃費が国内製の物より良かった理由は軽かったからだ、ということが分かります。このモデルによると、国内製の車と同じ重さの外国製の車は 100 マイル（約 160km）あたり、追加で 8 分の 5 ガロン（約 2.35L）のガソリンを消費することになります。

以上で、この一連の分析を終了します。

1.9 コマンドとメニューの違い

今までのセッションで Stata はメニュー操作とコマンドウィンドウのどちらからでも操作できることが分かりました。慣れてきたら、よく使用するコマンドはコマンドウィンドウで素早く実行し、グラフを作成するような複雑な操作はメニューとダイアログを利用すると効率的です。

Stata のコマンド構文 **(command syntax)** には一貫性があります。基本的にコマンドは次の構文を使用します。[ ] 内の項目はオプションであり、変数名は *varlist* に入力します。

[*prefix*:] *command* [*varlist*] [*if*] [*in*] [*weight*] [, *options*]

一般的なルール：

- ほとんどのコマンドでは機能を変化させる前置コマンドを併せて利用できます。詳しくは [U] **11.1.10 Prefix commands** をご覧ください。頻繁に使われる前置コマンドは by プレフィックスです。
- *varlist* の指定がない場合、全ての変数を使用します。
- *if* と *in* はコマンドの実行対象となるデータに条件を付加します。
- *options* (オプション) はコマンドの機能を修正します。
- 各コマンドの構文はシステムヘルプとマニュアルで確認できます。
- Stata のコマンド構文はここで紹介した以上に多くのものが存在しますが、とりあえずこれまでの知識で使い始めてみましょう。詳しくは [U] **11 Language syntax** と「help language」コマンドをご覧ください。

in と weight 以外のコマンドは、このセッション内で既に使用してきました。システムヘルプにはすべてのコマンド構文と例題が載っています。詳しくは [GSM] **4 Getting help (**ヘルプ・ヒントを見つける**)** をご覧ください。文法は基本的に同じですから、新しいコマンドの用法もすぐに分かりますし、他の研究者の分析内容を理解して読み解く際にも便利です。

以前使用した summarize コマンドをもとに構文を読んでみましょう。summarize の構文は Stata コマンドの典型例です。

summarize [*varlist*] [*if*] [*in*] [*weight*] [, *options*]

以下の内容を読み取ることができます。

コマンドのみ： summarize

コマンドと *varlist*(変数リスト)： summarize mpg

summarize mpg weight コマンド（と変数リスト）と *if* 条件文： summarize if mpg>20

summarize mpg weight if mpg>20 など

summarize の詳細は [R] **summarize** またはヘルプ **>Stata** のコマンド**...** と選択してから summarize と入力します。

1.10 作業内容を記録する

作業記録（ログ）を取るとそれまでの結果を見返し、変更内容を確認できるので便利です。ログの取り方は [GSM] **16 Saving and printing results by using logs (**ログを使い結果の保存や印刷を行う**)** で説明します。ログにはコマンドと出力結果が含まれるので、コマンド構文を学んでおけば自分が実行したコマンドをすぐに思い出すことができます。

結果ウィンドウが表示する内容を全てログとして記録するには、ノートのように見えるログボタン  をクリックします。普通のファイルと同じように、このログファイルを保存する場所を選び、ファイルに名前を付けます。ログ記録 (ロギング) を始めてから終えるまでの間、結果ウィンドウに表示されたすべてのものをログファイルは保存します。

1.11 ビデオチュートリアル

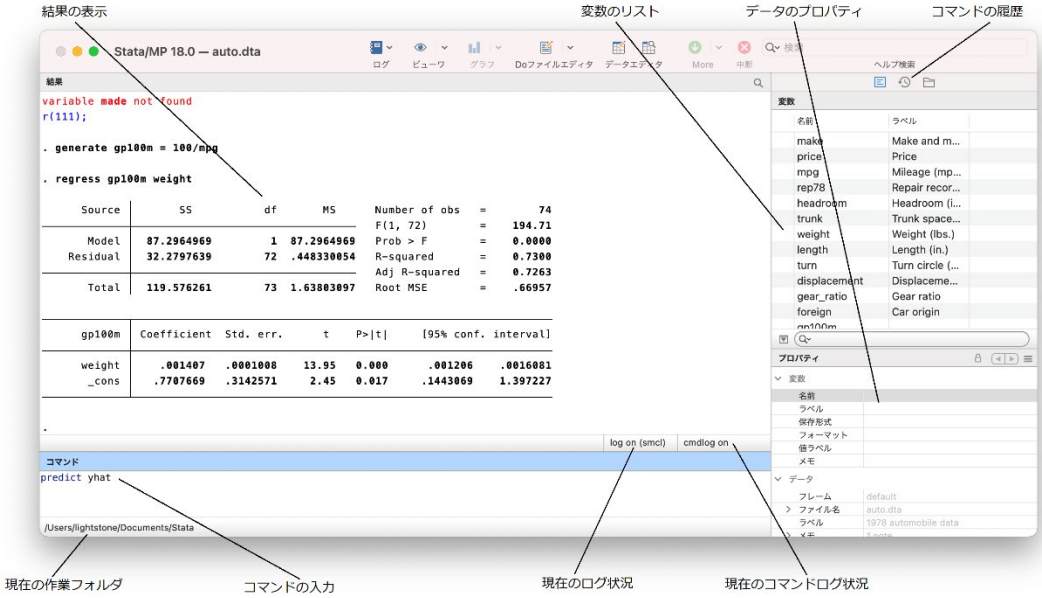
1.12 まとめ

この章では Stata の機能を簡単に紹介しました。このままマニュアルを読み進み、作業を続けてください。このマニュアルを一通り読み終えた上で、User’s Guide をご参照ください。

2 Stata のユーザインターフェイス

2.1 ウィンドウ

Stata インターフェイスの中心であるメインウィンドウ、ツールバー、メニューとダイアログについて紹介します。



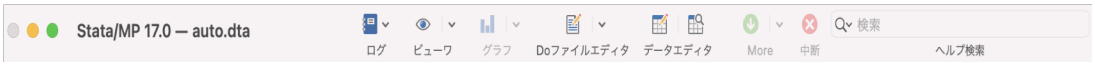
ワイドスクリーンレイアウトを選択すると、上記のようなスクリーン配置になります。Stata のメインウィンドウは履歴、結果、コマンド、変数、プロパティの 5 つのウィンドウで構成されます。各ウィンドウ独自の名前が上部のタイトルバーに表れます。この 5 つのウィンドウは、基本的に Stata の使用中は常に開いています。これらとは別に、ビューワ、データエディタ、変数マネージャ、 do ファイルエディタ、Graph、グラフエディタという、用途ごとに特化したウィンドウがあります。詳しくはマニュアルの後半に記します。

ウィンドウを開くまたは他ウィンドウの背面にあるウィンドウを表示するにはウィンドウメニューから対応するウィンドウを選択します。あるいは Mission Control を用います。あるいは Shift+Command+］キーで現在開いている Stata の全てのウィンドウを巡回できます。Stata の多くのウィンドウでは、右クリックを行うとコンテキストメニューが表示され、利用できる機能が表示されます。もしマウスにボタンが 1 つだけの場合、Control キーを押しながらクリックを行うと右クリックと同じ操作になります。ウィンドウ内での右クリックにより、利用できる機能がコンテキストメニュー

に表示されます。ウィンドウによってコンテキストメニューの内容は異なりますが、主にテキストのコピー、ウィンドウの設定変更、ウィンドウの印刷などのコマンドがあります。テキストのコピーや印刷を行う場合、対象となるウィンドウの取り違えを回避するためにも、メニューバーより右クリックを利用することをお勧めします。

2.2 ツールバー

メインツールバーを次に示します。



ツールバーには頻繁に利用するコマンドがボタンとして配置されています。ボタンが何の機能なのか忘れてしまった場合、マウスカーソルをボタン上に移動するとヒントが表れます。

矢印の付いたボタンでは、矢印をクリックするとさらに小さなメニューを表示します。ツールバーボタンと機能の概要は次の通りです。

ログ	新しいログの開始、現在のログの終了、中断、再開、のいずれかを選択して実行します。ログファイルについては [GSM] **16 Saving and printing results by using logs (**ログを使い結果の保存や印刷を行う**)** をご覧ください。あるいはビューワからもログファイルを選択して表示できます。
ビューワ	ビューワウィンドウを新たに開く、または既に開いているウィンドウを最前面にします。ボタンをクリックすると新規ビューワを開きます。詳しくは [GSM] **3 Using the Viewer (**ビューワを使う**)** をご覧ください。
グラフ	Graph ウィンドウを最前面にします。ボタンをクリックすると直近に選択したグラフを、ボタンを長押しすると選択したグラフを最前面に表示します。詳しくは [GSM]

14 Graphing data (データを作図する) 内にあるグラフボタンをご覧ください。

do ファイルエディタ

do ファイルエディタウィンドウを新たに開く、または既に開いているウィンドウを最前面にします。ボタンをクリックすると新規 do ファイルエディタが開き、ボタンを長押しすると、既に開いているウィンドウを選択して最前面にできます。詳しくは[GSM] 13 Using the Do-file Editor– automating Stata (do ファイルエディタを使用する–Stata の自動化) をご覧ください。

データエディタ

データエディタウィンドウを開くか、既に開いているウィンドウを最前面にします。詳しくは [GSM] **6 Using the Data Editor (データエディタを使用する)** をご覧ください。

データブラウザ

データエディタをブラウズモードで開きます。詳しくは [GSM] **6 Using the Data Editor (データエディタを使用する)** 内のブラウズモードをご覧ください。

More

長い出力表示の途中で一時停止した時に次の画面を表示します。クリックまたは長押ししてコマンドを最後まで実行させることもできます。詳しくは [GSM] **B.5 More** をご覧ください。

中断

実行中のタスクを中止します。詳しくは [GSM] **10 Listing data and basic command syntax (データのリストと基本コマンドの構文)** をご覧ください。

ヘルプ検索

公式なコマンドとユーザ作成コマンド、双方を検索します。詳細は、[GSM] 4 Getting help (ヘルプ・ヒントを見つける) をご覧ください。

2.3 コマンドウィンドウ

Stata はコマンドウィンドウに入力したコマンドを実行します。コマンドウィンドウでは基本的なテキスト編集、コピーと貼り付け、コマンド履歴、ファンクションキーのマッピング、フ

ファイル名と変数名コンプリート (variable-name completion) をサポートしています。コマンドウィンドウには、do ファイルエディタと同じく構文ハイライト機能があります。詳しくは、**[GSM] 13 Using the Do-file Editor—automating Stata (do ファイルエディタを使用する—Stata の自動化)** をご覧ください。

コマンドウィンドウをアクティブにして次のように操作してみましょう。

Page Up コマンド履歴を遡ります

Page Down コマンド履歴を進めます

Tab 途中まで入力した変数名を自動で記入します。候補が複数あるときはそのリストを表示します入力を二重引用符（”）で始めたときはファイル名についての自動補完を行います。

コマンドウィンドウ用のキーボードショートカットについては **[U] 10 Keyboard use** をご覧ください。

コマンド履歴は同一セッション内で実行したコマンドを振り返るのに有効で、編集を加えて再実行もできます。Stata のダイアログで実行されたコマンドについてもコマンド履歴に含まれているので、ダイアログを再び開くことなくコマンドを振り返り、必要に応じて再実行できます。

2.4 結果ウィンドウ

結果ウィンドウはセッション中のすべてのコマンドとテキスト形式の結果を表示します。結果ウィンドウをスクロールして操作やコマンドを確認するより、結果ウィンドウの検索バーを利用する方が簡単に操作できます。デフォルトでは検索バーは非表示になっており、表示するには結果ウィンドウの上部にある検索ボタン  をクリックします。

結果ウィンドウのウィンドウ内で右クリックをしてコンテキストメニューから結果のクリアを選択すると、結果ウィンドウのコンテンツを削除できます。このアクションはやり直しができません。

2.5 サイドバー

メインウィンドウにあるサイドバーでは、次の 2 つから 3 つタブによって表示内容を変更できます。

変数は変数ウィンドウとプロパティウィンドウを表示します履歴は履歴ウィンドウを表示します

履歴はウィンドウレイアウトがサイドバーに設定されている場合、履歴を表示します。うぃんどウィンドウレイアウトがワイドスクリーンの場合、履歴ウィンドウは Stata ウィンドウの反対側に表示されます

プロジェクトマネージャはプロジェクトマネージャウィンドウを表示します

デフォルトでは、変数が選択されていて、変数ウィンドウとプロパティウィンドウが表示されています。変数ウィンドウと履歴ウィンドウの機能については後述します。プロジェクトマネージャについては、[P] Project Manager をご覧ください。

2.6 変数ウィンドウ

変数ウィンドウは設定したプロパティとデータセット内の変数をリスト形式で表示します。デフォルトではすべての変数と変数ラベルを表示します。

変数ウィンドウ内で一度クリックすると変数を選択できます。複数の変数を選択する場合、それが隣り合っていない時は Command キー +マウスクリックを使い、隣り合っている時は Shift キー +マウスクリックを使って 1 度に選択します。

変数ウィンドウで変数をダブルクリックすると、コマンドウィンドウのカーソル位置にその変数を表示します。複数の変数が選択された状態で、いずれかの変数をダブルクリックすると、選択されていたすべての変数をコマンドウィンドウに表示します。

変数ウィンドウの左端の列を「ワンクリック貼り付け列」と呼びます。ワンクリック貼り付け列の上にマウスを移動し、矢印が出たらクリックします。この方法でもコマンドウィンドウに変数名を入力できます。

変数ウィンドウは変数の並び替えやフィルタリングの機能をサポートしています。虫眼鏡マークの右側の入力欄に入力したテキストで変数ウィンドウ内の変数にフィルタをかけることができます。フィルタは表示されている列全てに適用され、1 つでも条件に当てはまる変数を抽出します。デフォルトでは大文字・小文字を区別しません。虫眼鏡マークの右端にある矢印をクリックすると、この挙動を変更できます。

変数ウィンドウのヘッダーをクリックすると変数の並び替えができます。1 回クリックで昇順、2 回クリックで降順、そして 3 回クリックすると元のデータセット順に並べます。変数ヘッダーにある項目をクリックすると変数ウィンドウの表示のみがその項目を使ってアルファベット順になります。変数ウィンドウの並べ替えはリアルタイムに行われます。よって、変数ウィンドウをあるプロパティで並べ替えた後に変更を加えると、自動的に並び順を更新します。変数ウィンドウの変数の表示順を並べ替ても、データセット内の順番は変わりません。

変数ウィンドウ内の変数を右クリックするとコンテキストメニューを表示します。

- 変数"*vername*"のみ維持/選択した変数のみ維持　選択した変数だけをデータセットのメモリに残します。この操作はユーザに承認を求めます。メモリ内のデータセットのみ変更するので、ディスクに保存したデータセットには影響しません。詳しくは [GSM] **12 Using the Variables Manager (**変数マネージャを使用する**)** をご覧ください。
- 変数"*vername*"を削除/選択した変数を削除　データセットメモリから、選択した変数を削除します。この操作はユーザに承認を求めます。上記のように、これはメモリ内のデータセットのみ変更するので、ディスクに保存したデータセットには影響しません。詳しくは [GSM] **12 Using the Variables Manager (**変数マネージャを使用する**)** をご覧ください。
- 変数リストをコピー　選択した変数名をクリップボードにコピーします。
- すべて選択　フィルタの条件に当てはまる全ての変数を選択します。フィルタの指定がない場合、すべての変数を選択します。
- コマンドウィンドウに変数リストを送る　選択した変数を全てコマンドウィンドウに表示します。
- ユーザ設定**...** 変数ウィンドウの設定を編集します。
- 小さな変数リストを出力する　変数リストの表示を変数名のみに切り替えます。

コンテキストメニューの項目は通常の Stata コマンドと同じです。つまり、右クリックから操作を行うのはコマンドウィンドウに直接コマンドを入力するのと同じです。

変数ウィンドウを非表示にしたい場合、変数ウィンドウとの間にあるディバイダー（分割線）を掴み、右端に向かってドラッグします。これは、変数ウィンドウの幅をゼロにするイメージです。変数ウィンドウを非表示にすると、同時にプロパティウィンドウも非表示になります。

非表示の変数ウィンドウを再び表示するには、ウィンドウ > 変数ウィンドウと操作します。

2.7 プロパティウィンドウ

プロパティウィンドウは変数とデータセットのプロパティを表示します。変数ウィンドウで変数を選択すると、そのプロパティを表示します。変数ウィンドウで複数の変数を選択する場合、その選択したすべての変数に共通するプロパティをプロパティウィンドウに表示します。

プロパティウィンドウのタイトルバーにあるロックアイコンをクリックすると、選択した変数のプロパティを編集できます。デフォルトでは編集できません。プロパティのロックを解除すると、変数またはデータセットのプロパティを自由に編集できます。編集操作はコマンドとして結果と履歴ウィンドウに直接表示されます。当然コマンドログにも記録します。メモ

(Notes) の管理、変数と値ラベルの変更、表示形式も変更を行うにはプロパティウィンドウを使用するのが最も簡単です。詳しくは [D] **notes** 、 [D] **label** 、 [D] **format** をご覧ください。

ロックアイコンの隣にある矢印ボタンをクリックすると、変数ウィンドウ内の上または下にある変数が選択され、それらのプロパティを表示します。プロパティウィンドウを非表示にするには変数ウィンドウの  ボタンをクリックします。再びプロパティウィンドウを表示するには、  ボタンをクリックしてください。

より細かく変数の管理を行う場合は、優れたユーザーインターフェイスを備えている変数マネージャを利用してください。変数マネージャの詳細は [GSM] **7 Using the Variables Manager (**変数マネージャを使用する**)** を参照してください。

2.8 履歴ウィンドウ

履歴ウィンドウは入力したコマンドの履歴を表示します。成功したコマンドは黒で、失敗したコマンドはエラーコードと共に赤で表示します。履歴ウィンドウはデフォルトでは、サイドバーの履歴タブ内にあります。サイドバーにある履歴ボタン  をクリックして表示させます。

履歴ウィンドウの下部には、テキストの入力欄があり、ここに入力するテキストで表示されるコマンドをフィルタリングできます。デフォルトでは大文字/小文字を区別せずにテキストを検索します。虫眼鏡マークの右端にある矢印をクリックすると、フィルタリングの設定を変更できます。ここでエラーを非表示を選択すると、エラーが起きたコマンド履歴を非表示にできます。
履歴ウィンドウからコマンドを入力する時は次のようにします。

- リスト上のコマンドを 1 回クリックすると、再びコマンドウィンドウに表示します。
- リスト上のコマンドをダブルクリックすると再実行します。再実行したコマンドは履歴ウィンドウのコマンド履歴に再度登録されます。

履歴ウィンドウで右クリックをすると、コンテキストメニューを表示します。

- 切り取り 選択したコマンドを履歴ウィンドウから切り取り、クリップボードに移します。
- コピー 選択したコマンドをクリップボードにコピーします。
- 削除 選択したコマンドを履歴ウィンドウから削除します。
- すべて選択 履歴ウィンドウ内のコマンドをすべて選択します。
- すべてクリア 履歴ウィンドウ内のコマンドをすべて削除します。

- 選択範囲を実行　選択したすべてのコマンドを再実行し、コマンド履歴に追加します。エラーコマンドも含め、選択したコマンドはすべて再実行します。エラーが発生しても途中で止まりません。
- 選択範囲を **do** ファイルエディタへ送る　選択したすべてのコマンドを do ファイルエディタウィンドウに表示します。
- すべて保存**...** 履歴の内容を保存ダイアログを開き、履歴ウィンドウにあるすべてのコマンドを do ファイルとして保存します。(do ファイルに関する詳細は **[GSM] 13 Using the Do-file Editor—automating Stata (do** ファイルエディタを使用する**—Stata** の自動化**)** をご覧ください。
- 選択範囲を保存**...** 履歴の内容を保存ダイアログを開きます。履歴ウィンドウで選択したコマンドを do ファイルに保存します。
- ユーザ設定**...** 履歴ウィンドウの設定を変更します。

履歴ウィンドウを結果ウィンドウの左側のサイドバーに移動して、常に表示させることができます、表示 > レイアウト > ワイドスクリーン、と操作します。メインのウィンドウを大きくして、結果が折りたたまれないようにしましょう。カーソルをウィンドウの右・左端まで移動させて、リサイズシンボルになったら、外側にドラッグします。

履歴ウィンドウをサイドバー内に戻すには、表示 > レイアウト > サイドバー、と操作します。

2.9 タブ

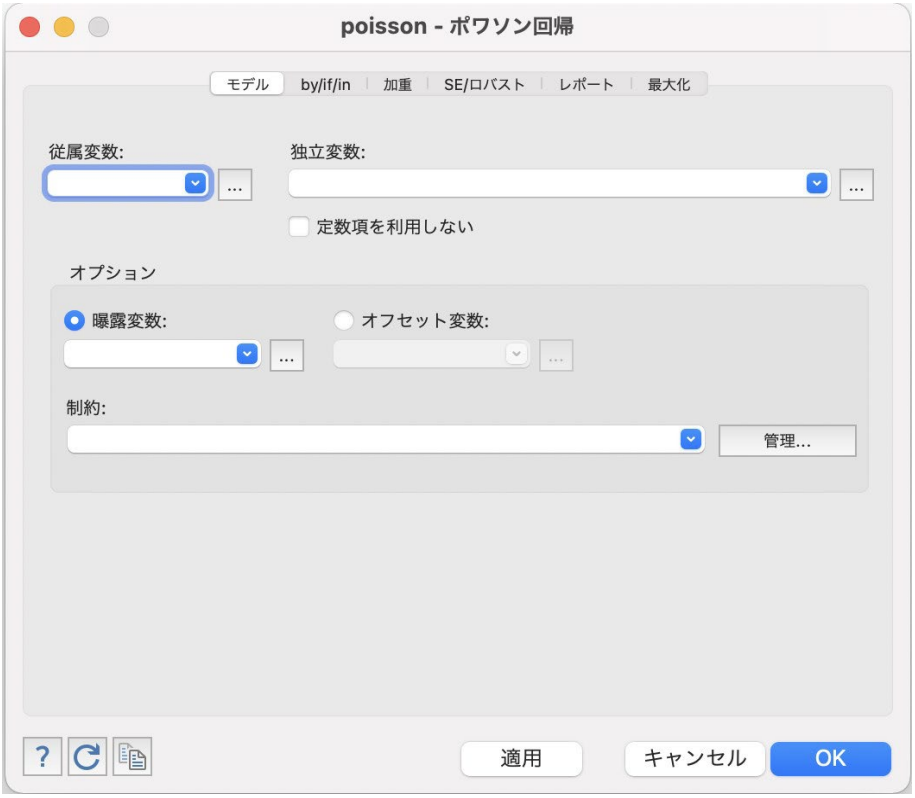
Stata のほとんどのウィンドウには複数のタブがあります。タブを領域外までドラッグすると、新しいウィンドウとして開きます。ウィンドウからタブをドラッグして、別のウィンドウにタブを移動することができます。ウィンドウ内に 1 つしかタブがなく、タブバーが表示されていない場合、ウィンドウ > タブバーを表示と操作してタブを選択します。デフォルトでは同じタイプのウィンドウがタブ付けされています。例えば、新規に Do ファイルエディタを開くと他の Do ファイルエディタと一緒になります。タブの新規作成ボタン  をクリックすると、アクティブになっているタブと同様のタブが新しく作成されます。タブの順番はタブバー内でドラッグして並び替えることができます。

タブの設定を変更するには、Stata > ユーザ設定 > 一般的なユーザ設定...、と操作してウィンドウタブを選択、変更したウィンドウタイプをを選択します。

2. 10 メニューとダイアログ

Stata を操作する方法は 2 通りあります。1 つはメニューを使用する方法で、もう 1 つはコマンドウィンドウを使用する方法です。 **[GSM] 1 Introducing Stata—sample session (Stata の紹介—サンプルセッション)** の中にあるサンプルセッションを操作して分かるように、どちらの方法にも便利なところはありました。このセクションではメニューとダイアログについてより詳しく紹介をします。

データ、グラフィックス、統計のメニューに Stata のほぼすべてのコマンドがあります。Stata はプログラミング機能を備えているので、ユーザは独自のダイアログやメニューを作成できます。作成したメニュー項目はユーザメニューに入ります。初期状態では空欄のサブメニューがあるだけです。例として、Poisson 回帰を行う場合を紹介します。Stata の「poisson」コマンドを入力するか、統計 > アウトカム (カウント) > ポアソン回帰とメニュー操作をすると、次のダイアログを表示します。

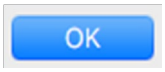


このダイアログは poisson コマンドで実行できるすべての機能を提供します。従属および独立変数は数値である必要があるため、数値が入力されている変数のみがコンボボックスで選択可能になります。ダイアログのタブを変更することで poisson コマンドが備える数多くのオプションを選ぶことができます。選択したコマンドのダイアログを見ると、機能の概略が分かります。

多くのダイアログには **by/if/in** と加重タブがあります。推定に利用する標本をコントロールしたり、データに重み付けを行うときに利用します。Stata のプログラミング言語に関する詳細は **[U] 11 Language syntax** をご覧ください。

推定を行うコマンドの多くには最大化タブがあり、そこでは最大化に関するオプションを設定します。([R] **maximize** をご覧ください。) 例えば、最適化のための反復回数の上限を設定できます。

ほとんどのダイアログには上記 **poisson** ダイアログの下部にある 6 つのボタンと同じものがあります。
ダイアログを通して入力したコマンドも手入力したものと全く同じです。コマンドは結果ウィンドウに表示され、実行後は履歴ウィンドウで確認できます。メニュー操作で実行したコマンドの出力を詳しく見ることで Stata のコマンド構文を学ぶことができます。



OK

ダイアログの設定に応じてコマンドを実行します。実行後、ダイアログを閉じます。

キャンセル

何もしないでダイアログを閉じます。赤い x 印のボタンと同じです。

適用

OK と同じようにコマンドを実行しますが、ダイアログは開いたままです。そのままダイアログに変更を加え、再適用すると新たにコマンドを実行します。この機能は、不慣れなコマンドの利用や、複雑なグラフの作成中に役立ちます。

ヘルプ

ヘルプシステムを起動します。ボタンをクリックするとそのダイアログに関連のあるヘルプファイルを表示します。画面上でこのボタンを押すと poisson のヘルプファイルにつながります。ヘルプファイルの Options の項目ではタブごとにオプションを解説します。

リセット

ダイアログをデフォルトの状態にリセットします。ダイアログは開く度に最後の状態を覚えています。ダイアログの画面をデフォルトに戻したい場合、このボタンをクリックしてください。

コマンドをクリップ

適用ボタンのように機能しますが、コマンドを実行する代わりにボードにコピーする クリップボードにコピーします。このコマンドは他のところ (例えば do ファイルエディタ) で使用できます。

メニューから Stata ダイアログを開く他に、2 つの方法でダイアログを開くことができます。目的のコマンドがメニュー内のどこにあるのか忘れてしまったときには次の方法を試してください。コマンドウィンドウに「db コマンド名」の形で入力します。

```
. db poisson
```

また、ダイアログを開くためのメニューコマンドはヘルプファイルからも調べられます。詳しくは

[GSM] **4 Getting help (**ヘルプ・ヒントを見つける**)** をご覧ください。

このマニュアルを読み進めて行くと Stata コマンドを直接入力する例が出てきます。例に書かれている通りに入力してコマンドを実行します。しかし、ダイアログを使用して同じ操作を行うこともできるので db コマンドを試してみてください。db コマンドはコマンドに対応するダイアログを素早く開きます。

2.11 現在の作業フォルダ

「ウィンドウ」セクションのスクリーンショットを見ると、Stata のメインウィンドウ一番下にあるステータスバーに、現在の作業フォルダがあります。このパス名が示す /Users/Stata/Documents/Stata が現在の作業フォルダの位置です。各セッションの終了時の作業フォルダは次のセッションの開始時の作業フォルダとなります。グラフやデータセットを「save ファイル名」のコマンドで保存すると、保存先フォルダは現在の作業フォルダになります。これはメニューから操作するファイルのアクション、例えばファイル > 保存またはファイル > 開く... などには影響を与えません。Stata の起動後には現在の作業フォルダを cd コマンドで変更できます。詳細は [D] **cd** をご覧ください。メインウィンドウには常に現在の作業フォルダ名を表示するので、グラフやデータセットの保存先を簡単に確認できます。

3 ビューワを使う

3.1 ビューワの目的

ビューワは Stata 内で多岐にわたって活用できる便利なツールです。Stata でヒントを得るために利用しますが、ビューワが提供するものはヘルプシステムだけではありません。ビューワでは、第 3 者が作成した拡張プログラムであるユーザ作成コマンド (user-written command) の追加・削除・管理ができます。また、現在あるいは過去の Stata ログ、および Stata 形式 (SMCL) またはテキスト形式のファイルも表示・印刷できます。更にウェブブラウザを使ってハイパーリンク先を表示できます。

この章は一般的なビューワの利用方法、ボタン、ビューワに入力できるコマンドについて説明します。ビューワに関する詳しい情報は **[GSM] 4 Getting help (ヘルプ・ヒントを見つける)** を、ユーザ作成コマンドのインストールについては **[GSM] 19 Updating and extending Stata—Internet functionality (Stata のアップデートと拡張—インターネットでの機能)** をご覧ください。

新しいビューワウィンドウを開く、または新しいビューワタグを開くにはビューワボタン  をクリックするか、ウィンドウ > ビューワ > 新規ビューワと選択します。

3.2 ビューワのボタン

ビューワのツールバーには複数のボタンに加え、コマンドボックスと検索ボックスがあります。

 戻る 画面を 1 つ戻ります。

 次へ 画面を 1 つ進みます。これは、最低でも 1 度は戻ったことを前提にします。

検索 ビューワ内でヘルプ検索の対象範囲を選択します。

検索バーは現在のビューワ内でのテキスト検索に使用します。検索バーをウィンドウ下部に表示するには編集 > 検索 > 検索... をクリックします。

検索バーには専用のボタンやエリアがあります。

前へ 検索テキストの直前の候補に移動します。ビューワの先頭まで戻って次の検索対象が無い時、自動的に最後の候補に戻ります。

次へ 検索テキストの次の候補に移動します。ビューワの最後まで検索を行って次の検索対象が無い時、自動的に先頭に戻ります。

完了 検索バーを閉じます。

3.3 ビューワの機能

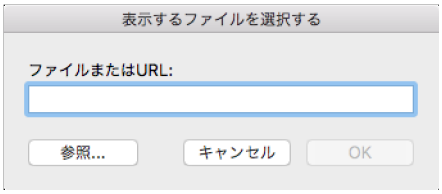
ビューワはウェブのブラウザと似ており、リンクがあります (デフォルトでは、青文字で表示)。リンクをクリックすると関連のあるヘルプトピックを表示し、第三者が作成したソフトをインストールおよび管理できます。マウスカーソルをリンク上に移動すると、そのリンクで実行される動作をウィンドウ下部にあるステータスバーに表示します。リンクが「help logistic」の時、このリンクをクリックすると logistic コマンドのヘルプファイルを表示します。ビューワのリンク上で Command+ クリックすると、リンク先を新しいビューワウィンドウに表示します。新しいビューワでリンク先を表示するには Shift+ クリックと操作します。

新しいビューワを開くにはウィンドウ > ビューワ > 新規ビューワと選択するか、メインウィンドウのツールバーにあるビューワボタンをクリックします。コマンドウィンドウに「help」コマンドを入力しても同じように新しいビューワを開きます。

複数のビューワウィンドウが開いている時に、1 つのビューワを最前面に移動するにはウィンドウ > ビューワと選択し、そのリストから該当するビューワを選びます。すべてのビューワを閉じるを選択すると開いているすべてのビューワウィンドウとタブを閉じます。

3.4 SMCL ファイルを含むローカルテキストファイルを表示する

Stata のヘルプファイルを表示する以外にも、ビューワを使用して Stata Markup and Control Language (SMCL) ファイルを見ることができます。SMCL は通常、作業のログを取るときに作成します。(詳しくは **[GSM] 16 Saving and printing results by using logs (ログを使い結果の保存や印刷を行う)** をご覧ください。) また、ビューワではテキストファイルも表示できます。ファイル > 表示... と操作すると、ファイルを開いて内容を確認できます。



開きたいファイル名を入力して **OK** を押すか、参照**...** ボタンを押してファイルダイアログを開き、ファイル名を選択します。

記録中のログもビューワで表示できます。結果ウィンドウでスクロールバックする方法と違い、新たな出力が追加されても現在の表示が固定される、という利点があります。現在のログファイルを参照する場合、ファイル > ログ > 表示**...** と選択し、ファイルを選択するダイアログを開きます。ログファイルのパスとファイル名がすでに選択されています。**OK** をクリックすると、ログをビューワに表示します。詳しくは **[GSM] 16 Saving and printing results by using logs (**ログを使い結果の保存や印刷を行う**)** をご覧ください。

3.5 インターネット上のファイルを閲覧する

インターネット上のファイルを閲覧する時は、ローカルファイルを表示する場合と同様に操作します。ただし、参照**...** ボタンでファイルでなく閲覧したい URL を、例えば https://www.stata.com/man/readme.smcl と入力します。ウェブ上のテキストか SMCL ファイルを見る時のみビューワで参照ができます。HTML ファイルのアドレスを入力すると、HTML ソースを表示します。

3.6 ビューワ内をナビゲートする

ウィンドウのスクロールバーの他に、キーボードの矢印上下キーまたは Page Up/Page Down キーでもウィンドウのナビゲートができます。矢印 (上下) キーを使用すると、1 行ごとにウィンドウがスクロールします。Page Up/Page Down キーを使用すると、1 画面ごとにウィンドウがスクロールします。

3.7 印刷

ビューワを印刷するには目的のビューワウィンドウ内で右クリックをし、コンテキストメニューから印刷**...** を選びます。または、メインウィンドウでメニューからファイル > 印刷 > ビューワタイトルと選択します。

3.8 ビューワウィンドウで右クリックする

ビューワウィンドウで右クリックすると以下のオプションを含むコンテキストメニューを表示します。

- すべて選択 ウィンドウ上にあるすべてのテキストを選択します。
- ユーザ設定... ビューワの設定を編集します。
- 印刷... ビューワの内容を印刷します。

コンテクストメニューには上記以外の項目も表示されます。

3.9 ビューワでヘルプを検索する

ビューワの検索ボックスでは文書 (ヘルプ文書を含む) を検索できます。虫眼鏡をクリックします。そしてすべて検索、ドキュメントと FAQ を検索、インターネットリソースを検索の中から 1 つを選びます。そして検索語句を検索ボックスに入力し、Return を押します。ビューワをヘルプとして使用するための詳細は [GSM] **4 Getting help (**ヘルプ・ヒントを見つける**)** をご覧ください。

3.10 ビューワのコマンド

ビューワの中でリンクやボタンをクリックして行える全ての操作は、ビューワのコマンドボックス (ウィンドウの上部) か Stata のコマンドラインにコマンドを入力して実行することも可能です。ビューワで実行できるコマンドをいくつかご紹介します。

- ヘルプを参照する (詳しくは [GSM] **4 Getting help (**ヘルプ・ヒントを見つける**)** をご覧ください。)

  「contents」と打ち込んで Stata のヘルプシステムを表示します。
  「コマンド名」を入力して Stata のコマンドヘルプを参照します。
  「キーワード」を入力して、ドキュメント、FAQ、やインターネットリソースを検索します。
- 検索する (詳しくは [GSM] **4 Getting help (**ヘルプ・ヒントを見つける**)** をご覧ください)。

  「search キーワード」と打ち込んでそのトピックに関する文書、FAQ、インターネット上のリソースを検索します。
  「search キーワード, local」と入力すると、そのトピックに関する文書と FAQ のみを検索します。
  「search キーワード, net」と入力すると、インターネット上の資料のみを検索します。

- ユーザ作成プログラムを検索し、インストールする (詳しくは [GSM] **4 Getting help (**ヘルプ・ヒントを見つける**)** と [GSM] **19 Updating and extending Stata—Internet functionality (Stata** のアップデートと拡張**—**インターネットでの機能**)** をご覧ください)。

  「net from https://www.stata.com/」を入力すると Stata Journal、ユーザ作成プログラムをインターネット上で検索してインストールできます。

  「ado」と打つとユーザ作成のインストール済みプログラムを確認できます。

  「ado uninstall」と入力すると、該当するコンピュータにインストールされているユーザ作成プログラムをアンインストールできます。

- ビューワでファイルを見る。

  「view ファイル名.smcl」で SMCL ファイルを表示します。

  「view ファイル名.txt」でテキストファイルを表示します。

  「view ファイル名.log」でテキストのログファイルを表示します。

- 結果ウィンドウでファイルを見る。

  「type ファイル名.smcl」とコマンドウィンドウに入力すると SMCL ファイルを結果ウィンドウに表示します。

  「type ファイル名.txt」とコマンドウィンドウに打ち込むとテキストファイルを結果ウィンドウに表示します。

  「type ファイル名.log」とコマンドウィンドウに入力するとテキストログファイルを結果ウィンドウに表示します。

- ブラウザを起動して HTML ファイルを表示する。

  「browse URL」を入力してブラウザを起動します。

3.11 コマンドウィンドウからビューワを使う

「help コマンド名」をコマンドウィンドウに打ち込むと新規ビューワに指示されたコマンドのヘルプを表示します。

4 ヘルプ・ヒントを見つける

4.1 システムヘルプ

Stata のヘルプシステムには分析上役立つ情報が豊富に用意されていますので是非ご利用ください。一般的に次の手順に沿って目的の統計およびデータ管理用のコマンドについて調べる事ができます。

1. ヘルプ > 検索... の中から全てを検索を選び、トピックまたはキーワードを入力します。このコマンドは新しいビューワウィンドウを開き、Stata のコマンド、Stata Journal 内のリファレンス記事、Stata のウェブ上にある FAQ (Frequently Asked Questions) へのリンク、Stata の YouTube チャンネルにある動画へのリンク、厳選した外部ウェブサイトへのリンク、ユーザ作成コマンドへのリンク等の情報を表示します。
2. これらの結果を確認してください。結果の中に参考になりそうなコマンドがあった場合、そのコマンド名のリンクをクリックし、ヘルプファイルを開きます。
3. 表示されたヘルプファイルをご一読ください。
4. より詳しいヘルプが必要な場合は、コマンド名のリンクをクリックして PDF 文書を開いてください。
5. 検索したファイルが目的のものでなかった場合、ビューワ右上の関連項目ボタンをクリックして関連するヘルプファイルのリストから他のリンクを選択します。あるいは、戻るボタンをクリックして直前の文書に戻り、他のヘルプファイルを検索します。
6. ヘルプファイルを開いた状態でもコマンドウィンドウにコマンドを入力して目的のコマンドを実行できます。ビューワ右上のダイアログボタンをクリックすると表示しているヘルプのコマンドダイアログを開きますので直接ダイアログからコマンドを実行できます。
7. 検索をやり直す場合は、新たなキーワードをビューワウィンドウの検索ボックスに入力します。
8. ドキュメントと FAQ の検索でキーワード検索を行った場合、Stata はコマンド、Stata Journal 記事とソフトウェア、FAQ、動画を検索します。インターネットリソースの検索を選択すると、Stata はユーザ作成コマンド (Stata Journal またはそれ以外の場所) を検索します。詳細は **[GSM] 19 Updating and extending Stata—Internet functionality (Stata** のアップデートと拡張**—**インターネットでの機能**)** をご覧ください。

では例題を使用してヘルプシステムの説明を行います。実際にコンピュータを動かしながら読み進めてみましょう。

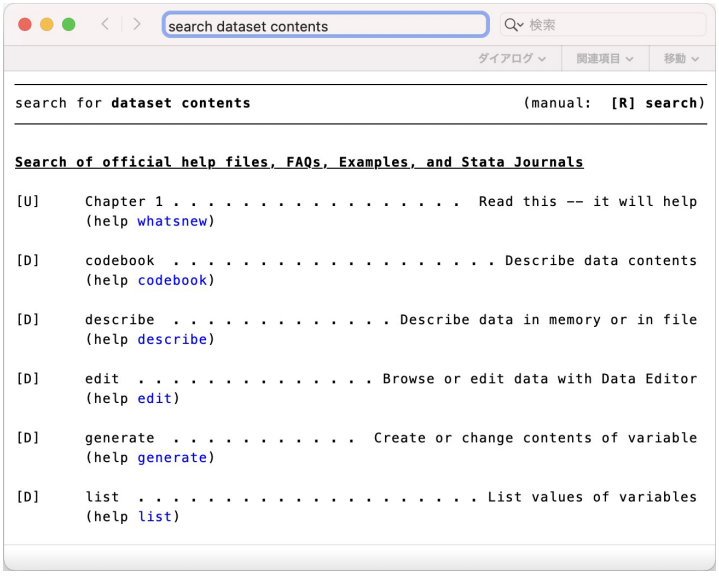
例えば、与えられたアンティークカーデータセットの内容を確認します。似たようなことを **[GSM] 1 Introducing Stata—sample session (Stata** の紹介**—**サンプルセッション**)** のサンプルセッションで行ったような気はしますが詳しいコマンドを覚えていないことにしましょう。

そこで、まずは「sysuse auto, clear」とコマンドウィンドウに入力してデータセットを開きます。(**clear** オプションについての詳細は [GSM] **5 Opening and saving Stata datasets (Stata** のデータセットを開く・保存する**)** をご覧ください。)

上記手法で調べていきます。

1. メニューからヘルプ > 検索**...** と選択します。
2. 全てを検索のラジオボタンが選択してあることを確認してください。
3. 「dataset contents」と検索ボックスに打ち込み、**OK** をクリックするか Return を押します。Return キーを押す前のウィンドウは次のようになります。

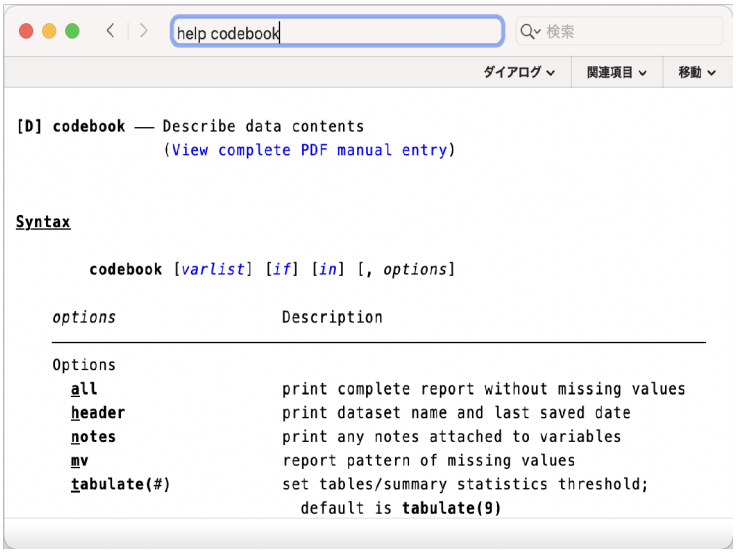
4. Stata はコマンド、リファレンスマニュアル、Stata Journal、Stata のウェブ上の FAQ、ユーザ作成コマンドの中を“dataset contents”のキーワードで検索します。結果は次の通りです。



5. 検索結果を見ると、「codebook」と「describe」コマンドが使えそうです。データセットの中身を知りたいので、「codebook」コマンドを確認しましょう。[D] は Data Management Reference Manual で codebook コマンドについて調べることを意味しています。(help

codebook) の中にある codebook リンクは codebook コマンドのヘルプファイルがシステムヘルプにあることを示しています。これが今探していたものです。

codebook のリンクをクリックします。リンクは多様なリソース、例えば Stata コマンド、ダイアログ、ウェブページ等に繋がっています。目的のリンクから codebook コマンドのヘルプファイルに移動します。

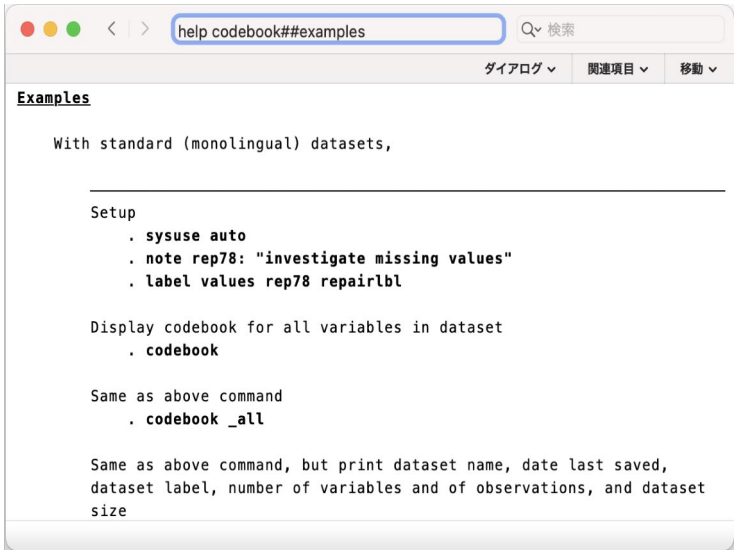


1. 開いた画面はコマンドのヘルプの典型的な例です。コマンドのヘルプファイルは、上から順に次のような内容のセクションに分かれています。
   (a) クイックアクセスツールバー (タブのすぐ下にあるバー) には 3 つのボタンがあります。
      i. ダイアログボタンはコマンドに関連するダイアログへのリンクを表示します。
      ii. 関連項目ボタンは関連する PDF 文書またはヘルプファイルのリンクを表示します。
      iii. 移動ボタンは現在のヘルプファイル内のセクションへのリンクを表示します。
   (b) ヘルプファイルの 2 行目の View Complete PDF Manual entry のリンクが表示されています。このリンクをクリックすると、コマンドのドキュメントを PDF ビューワで開きます、ここでは、codebook です。
   (c) コマンド構文、つまり Stata が読み取るコマンド構成のルールについて [ ] は、codebook コマンドに対してカッコ内の引数が任意であることを表します。引数を指定する場合は *varlist*、if 条件、in 条件の他にいくつかのオプションを選択できます。(使用可能なオプションはコマンドにより大きく異なります。) オプションはコマンドのすぐ下に記載してあり、詳しい説明はヘルプファイルの後半にあります。コマンド構文は、 **[GSM] 10 Listing data and basic command syntax (データのリストと基本コマンドの構文)** で詳しく学びます。
   (d) コマンドの概略を示します。codebook はデータセットの各変数の情報を示すだけの簡単なものなので、短い説明しかありません。

(e) このコマンドで使用できるオプションです。ここのリストは構文における解説と比べると、各オプションについてさらに詳しく説明しています。例えば、mv オプションは欠損値がある時に、欠損値の出現には何らかのパターンがあるか調べるものです。このオプションはデータ整理や補完計算 (imputation) に役立ちます。

(f) コマンド使用例があります。この codebook の例題は、実際に 1 ステップずつ操作していく例題です。必要なデータは Stata と共にシステムにインストールされているかインターネットから無料でダウンロードできます。(g) コマンド実行後にメモリ上に確保される情報です。この内容は基本的にプログラマー向けです。

では、移動ボタンをクリックして **Examples** を選ぶか、Examples までスクロールダウンします。例題に挑戦してみましょう。例題の先頭部分スクリーンショットは次の通りです。



4.2 ヘルプを探す

検索... では統計、グラフ、データ管理、プログラミングに関する機能を検索できます。これらは公式的なリリースに組み込まれているかユーザ定義コマンドとして使用できます。検索するトピックを入力するときは、統計の正しい表現を使用してください。例えば、「Mann-Whitney」と入力します。また、複数のキーワードを「regression residuals」のように入力することも可能です。

検索... を使用するときは「正しい英語」と「正しい統計用語 (英語)」を使用してください。コマンド名が分かっていて、直接ヘルプファイルを表示したい場合はメニューからヘルプ > **Stata** のコマンド... と選び、コマンド名を入力します。あるいはビューワの上部の検索欄にコマンド名を入力し Return キーを押します。

ヘルプはキーワードがコマンドがトピックかを区別します。これはいくつかのコマンド名はそのままトピックとしても使用できるからです。例えば「logistic」は Stata のコマンドです。ここでヘルプメ

ニューから **Stata** のコマンド**...** を選び「logistic」と入力すると、直接 logistic コマンドのヘルプファイルに移動します。一方、検索**...** を選択し、「logistic」と打ち込むと、検索結果としてロジスティック回帰に関する多くのコマンドを表示します。

ビューワ内部からでも検索できることを忘れないでください。検索する場合はコマンドをビューワのコマンドボックスに直接入力します。または検索ボックスの左にある虫眼鏡をクリックし、検索の範囲を設定した後にキーワードを検索ボックスに入力して Return キーを押します。

4.3 ヘルプと検索のコマンド

ヘルプシステムはコマンドウィンドウからも利用可能です。ある特定のコマンドのヘルプを見たい時などに特に便利です。次に使用できるコマンドをリストで記します。

- 「help コマンド名」を入力すると、ヘルプ **>Stata** のコマンド**...** と選択してコマンド名を入力するのと同じになります。このヘルプファイルは新しいビューワウィンドウに表示します。
- 「search 項目」をコマンドウィンドウに入力すると、ヘルプ > 検索**...** の後に全てを検索を選んで項目を入力した場合と同じになります。検索結果は新しいビューワウィンドウに表示します。
- 「search 項目, local」をコマンドウィンドウに入力すると、ヘルプ > 検索**...** の後にドキュメントと **FAQ** の検索を選んで項目を入力した場合と同じになります。ビューワウィンドウではなく結果ウィンドウに検索結果を表示します。
- 「search 項目, net」をコマンドウィンドウに入力すると、ヘルプ > 検索**...** の後にインターネットリソースの検索を選択して項目を入力した場合と同じになります。ビューワウィンドウではなく結果ウィンドウに検索結果を表示します。

詳しくはユーザガイド内の **[U] 4 Stata's help and search facilities** と **[U] 4.8 search:All the details** でコマンド言語版のヘルプシステムについての情報をご覧ください。search コマンドは、特にここで説明していない機能 (たとえば作者検索) を備えています。

4.4Stata リファレンスマニュアルと User's Guide

全ての Stata リファレンスマニュアルは PDF ファイルでソフト内に含まれています。マニュアル自体は内部にクロスリファレンスが多くあり、クリックで関連するページを移動できます。順序に沿って文書を読む必要はありません。

ヘルプファイル内の多くのリンクは PDF マニュアルにつながっています。リンクをクリックすると、マニュアル内にある多くの情報をご利用いただけます。Stata のヘルプシステムは多くの内容を含みますが、マニュアルの中にある情報のほんの一部分でしかありません。

Stata のリファレンスマニュアルはアルファベット順に並んでいます。Getting Started マニュアルの各 OS 版は、それぞれ索引がついています。全マニュアルの統合的な索引は、Stata Index に記載されています。コマンドについて調べたい時に、まずこの統合索引を確認してみると良いでしょう。

索引語の **collapse**, **egen**, **summarize** のようにそれ自身も Stata のコマンドとなるものも多くあります。

検索結果およびヘルプファイルの [R] **ci** 、 [R] **regress** 、 [R] **ttest** などという表記は Base Reference Manual を参照します。この他にも [P] **pyStata Integration** があり、これは Programming Reference Manual を、 [U] **12 Data** は User’s Guide を参照します。マニュアルのリストとそれぞれの略語表記についてはこのマニュアルの目次直後にある「**Stata** の他のマニュアル参照について」をご
覧ください。

これらのリファレンスマニュアルの使用に関するアドバイスは [GSM] **18 Learning more about Stata (Stata** についてもっと詳しく学ぶ**)** 、または [U] **1.2 The User’s Guide and the Reference manuals** をご覧ください。

4.5 Stata 動画

Stata YouTube channel は Stata を知る上で優れたリソースです。各トピックが短めの動画の中で実際に Stata を操作しながら紹介されます。扱っているトピックは、データ管理、グラフィックス、要約統計、仮説検定から、マルチレベルモデル、構造方程式モデルなどの高度な話題にまで及びます。

トピックに沿って一連の動画を再生するプレイリストも用意されています。例えば、”Power and sample size calculations”のプレイリストには 2 つの独立した群や対応のある 2 群における検出力、標本の大きさ、効果量の計算方法を紹介する動画がまとまっています。”Survival analysis?”では生存分析のためのデータ設定処理、基本的な記述分析、グラフ化、生存関数や生命表の計算という一連の流れが閲覧できます。”Time series?”では時系列分析用のデータ設定処理、時系列グラフの作成、推定式での演算子(オペレータ)の使用法、ARMA や ARIMA モデルでのフィットの操作法がご覧になれます。
また、”Back-to-school video?”では学生、Stata を初めてまたは久しぶりに使う方のための動画を集めています。

https://www.stata.com/links/video-tutorials/には、最新のものを含む動画がトピックごとに紹介されています。https://www.youtube.com/user/statacorp/からは、プレイリストへ直接アクセスできます。

4.6 The Stata Journal

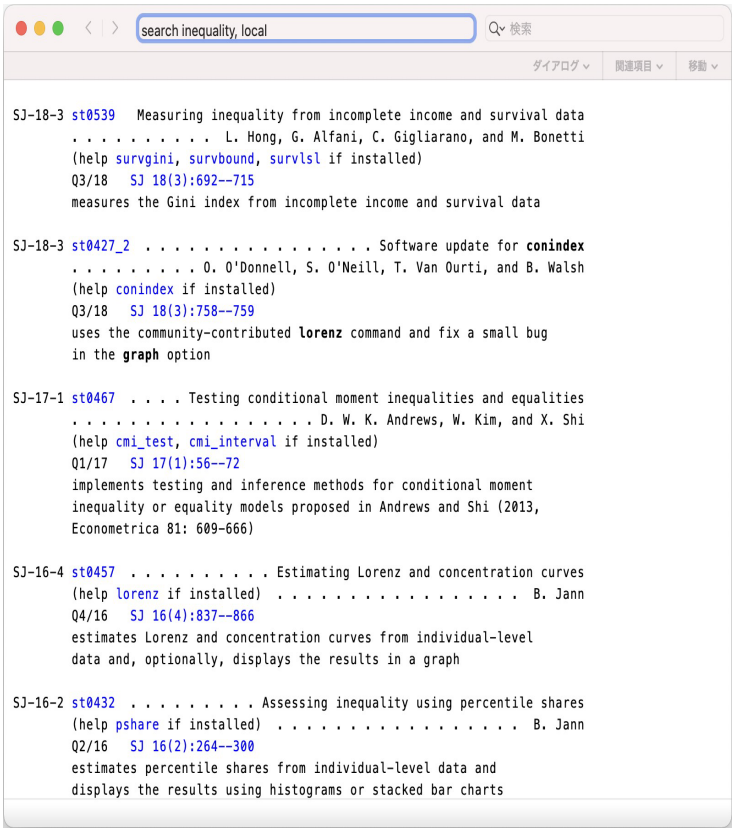
Stata で検索を行っている時、Stata Journal へのリンクをよく見かけます。

Stata Journal は印刷および電子版の機関誌で、年 4 回発行しています。内容は統計、データ分析、教授方法、Stata 言語の有効活用法などにわたります。Stata Journal は査読付きの論文ですが、短い注釈やコメント、コラム、ヒント、書評など、統計を様々な分野で利用する人に有益な情報も合わせて掲載しています。初心者から熟練者までのすべての Stata ユーザを対象にした機関誌です。Stata の熟練度は問いません。統計、実験計画、データ管理、グラフ作成、結果発表などを、Stata で行う研究者のための論文誌です。詳しくは https://www.stata-journal.com をご覧ください。

この Stata Journal には論文で説明しているプログラムとデータセットが対になって用意されています。プログラムとデータセットは購読者だけでなく Stata ユーザならばインターネット経由でダウンロードしインストールできます。詳しくは [R] **net** と [R] **sj** をご覧ください。

Stata Journal には不均一性の計測についての記事が複数あります。ヘルプ > 検索**...** と選択し、ドキュメントと **FAQ** の検索を選んだ後、「inequality」と入力してから少しスクロールします。

4.6 The Stata Journal



SJ-18-3 は volume 18 の 3 番目に発行した Stata Journal です。st0539 はパッケージ番号を表します。st はこのパッケージが Stata Journal の“statistics（統計）”カテゴリに分類されることを意味

します。続く記述はソフトウェアパッケージのタイトルと著者名です。続くカッコ内にはパッケージ内で使用されるユーザ作成コマンドが記述されます。SJ のリンク (たとえば、SJ 18(3):692--715 をクリックするとブラウザが起動して Stata Journal のウェブサイトに移動します。ここで要約や記事をダウンロードできます。各検索結果は、ユーザ作成コマンドの簡単な説明で記述が終了します。

Stata Journal のウェブサイトでは 3 年以上経過した記事は全て無料でダウンロードできます。このソフトウェアのインストールについての詳細は [GSM] **19 Updating and extending Stata—Internet functionality (Stata** のアップデートと拡張**—**インターネットでの機能**)** の中の「ユーザ作成のプログラムをダウンロードする」を参照してください。そして便利なインターフェイスやリソースは Statistical Software Components (SSC) Archive から利用できます。詳細は [R] **ssc** をご覧ください。

ユーザには Stata Journal を購読することをお勧めします。詳しくは [U] **3.4 The Stata Journal** をご覧ください。

他のプログラムやデータセットを無料でダウンロードできるサイトへのリンクは Stata のウェブサイト (https://www.stata.com/links/) に掲載されています。

5 Stata のデータセットを開く・保存する

5.1 データセットをロードし、保存する

Stata のデータセットを開き、保存するには他のコンピュータアプリケーションと同じように操作します。しかし、2 つの相違点があります。1 つ目はコマンドウィンドウからファイルを開き、保存できる点です。2 つ目は、Stata は一度に 1 つのデータセットしかアクティブにすることができません。一度に複数のデータセットをメモリに読み込みことはできますが ([D] frames intro)、アクティブにできるデータセットは 1 つのみです。新しいデータセットを開くときには必ず既存のデータセットを閉じてください。この章ではデータセットを開き、保存する全ての方法を紹介します。

Stata のデータセットは複数の方法で開くことができます。ほとんどの方法は他のアプリケーションでなじみ深いものでしょう。

- Stata のデータファイル (拡張子が.dta) をダブルクリックします。メモ：拡張子はコンピュータのシステム設定によっては非表示になっている場合もあります。
- ファイル > 開く... とメニュー操作するか、開くボタンを押してファイルを選択します。
- ファイル > データのサブセットを開く... と操作し、ファイルを選択し、データの範囲またはデータセット内の変数を指定します。
- ファイル > 最近のデータセット > ファイル名、と選択します。
- 「use ファイル名」とコマンドウィンドウに入力します。このコマンドは、現在の作業フォルダ (working directory) 内を入力した「ファイル名」で検索します。ファイルが他のフォルダにあるなら、目的のフォルダのパスを指定しなければなりません。もしパスやファイル名のどこかにスペース (空白) があるなら、そのファイル名をダブルクオテーション（“ ”）の中に入れるよう注意してください。詳しくは [U] **11.6 Filenaming conventions** をご覧ください。
- 「sysuse ファイル名」とコマンドウィンドウに入力します。このコマンドは、adopath と呼ばれるディレクトリ内にあるファイル名を検索します。通常、このコマンドは Stata をインストールする時にソフトウェアと共にインストールした例題のデータセットを検索しますが、それ以外でも、自分のデータセットに簡単にアクセスする手段として使用できます。adopath についての詳細は [P] **sysdir** をご覧ください。
- 「webuse ファイル名」とコマンドウィンドウに入力します。webuse コマンドは Stata マニュアルで使用するデータセットにアクセスする際に使用します。例えば、「webuse lbw」は logistic コマンドのマニュアル文書内で使用している lbw データセットをロードします。詳細は [D] **webuse** をご覧ください。

現在のフレームでデータセットを開く時、すでに他のデータセットがフレームにあれば上書きされます。Stata は一度に 1 つのデータセットしか開けないので、データセットを新たに開くとメモリ内にあるデータセットを破棄します。他のフレームのデータセットは影響を受けません。Stata は現在開いているデータセットに変更を加えた場合、操作を通じて強制しない限りデータセットの破棄を拒否します。コマンドウィンドウ以外の方法でファイルを開こうとするとポップアップを表示します。以下のエラーメッセージを表示します。

```
. sysuse auto
no; dataset in memory has changed since last saved
r(4);
```

この機能は誤ってデータを消す可能性を低くします。

名前を付けていないデータセットを保存する (または既存のデータセットを新しい名前で保存する) には次のように操作します。

1. ファイル > 名前を付けて保存**...** と選択します。
2. 「save ファイル名」とコマンドウィンドウに打ち込みます。

Stata 13 の形式でデータセットを保存するには、次のようにします。

1. ファイル > 名前を付けて保存**...** を選び、ファイルの種類のドロップダウンリストから **Stata 13** データ **(*.dta)** を選びます。
2. 「saveold ファイル名」とコマンドウィンドウに打ち込みます。

変更したデータセットを保存する（または元のデータセットを上書きする）には次のように操作します。

1. ファイル > 保存**...** と選択します。
2. 保存ボタンをクリックします。
3. 「save ファイル名」とコマンドウィンドウに入力します。

データセットを上書きするとデータセットを元に戻すことはできません。重要なデータセットに関しては元のデータのコピーをバックアップとしておくか、変更後の内容を新しい名前で保存します。これはワープロのドキュメント (Microsoft Word など) と同じですが、うかつに上書き保存をするとデータセットの回復はほぼ不可能です。

重要：データセットに施した変更は保存するまで完了しません。この間の作業はデータセットのメモリで行われており、データファイルそのものではありません。ほぼすべてのコンピュータアプリケーションはこのように動作しています。

既存のデータセットを保存しない場合、そのデータセットをクリアすれば新たなデータセットを開けます。他のファイルを開くにはコマンドウィンドウに「use（開きたい）ファイル名, clear」と入力します。

5.2 ディスクからフレームのセットを開く、またはディスクに保存する

複数のフレームまたはフレームセットは frames save を使用し、単一の.dtas 形式で保存できます。フレームセットを開くには、frames describe を使用します。

6 データエディタを使用する

6.1 データエディタ

データエディタは現在メモリ内にあるデータをスプレッドシート形式で表示します。この画面を使用してデータの入力、編集、データセットの属性編集を行えます。変数名、ラベル、表示形式の設定や値ラベルを作成する時は、この属性を編集します。

データエディタはデータの表示だけでなく、変数の変更とプロパティの変更を行う、変数ウィンドウとプロパティウィンドウも表示します。これはメインウィンドウにある、同名のウィンドウと同じように機能します。

データエディタで行った操作も、あたかもコマンドウィンドウに直接入力したかのように結果ウィンドウへ表示されます。つまり、何を行ったのか記録できるのでデータエディタを通じてコマンドを学ぶこともできます。

データエディタは作業中表示しておくことができ、データの様子を見ながら作業できます。データを誤って変更しないようにデータエディタには 2 つの表示モードがあります。編集モードで編集を加え、ブラウズモードで表示のみを行います。ブラウズモードではデータエディタウィンドウ内でデータを編集できません。データをチェックする時などはブラウズモードを使い、編集が必要な場合にのみ編集モードに切り替えて使用することをお勧めします。

データエディタ上のデータを印刷するには、ファイルメニューの印刷を選択します。

この章ではデータ入力および編集を実際に行います。また変数とプロパティのウィンドウを使用して値の変更も行います。ではデータエディタボタン  をクリックして編集モードのデータエディタを開きましょう。

6.2 データエディタ内のボタン

データエディタのツールバーには標準ツールバーボタンに加え、いくつか新しいボタンがあります。


編集モード　データエディタを編集モードに変更します。


ブラウズモード　データエディタを閲覧専用のブラウズモードに変更します。


観測値フィルタ データエディタで表示する観測値にフィルタをかけます。このボタンはデータセット内のサブセットを表示するのに便利です。


変数 変数ウィンドウとプロパティウィンドウの表示/非表示を切り替えます。


プロパティ プロパティウィンドウの表示/非表示を切り替えます。

データエディタのサイドバーには、サイドバーに表示する内容を変更するボタンがあります。これは実際にはタブですが、外見上はボタンなので、このマニュアルではボタンとして扱います。


変数 変数とプロパティウィンドウを正面に表示します。。


スナップショット スナップショットウィンドウを正面に表示します。詳細は下記スナップショットで作業するをご覧ください。


フィルタ 観測値フィルタウィンドウを正面に表示します。このウィンドウはデータセット内のサブセットを表示するのに便利です。

データエディタでアクティブなセルを動かす時は次のようにします。

- 右に移動するには Tab キーか、右矢印キーを押します。
- 左に移動するには Shift+Tab キーか、左矢印キーを押します。
- 下に移動するには Return キーか、下矢印キーを押します。
- 上に移動するには Shift+Return キーか、上矢印キーを押します。

目的のセルをクリックすると直接選択できます。

データエディタ内で右クリックをするとコンテキストメニューを表示します。このメニューを使ってデータや表示内容を変更できます。このメニューを使ってデータや表示内容を変更できます。詳しくは章の後半で説明します。実際にデータエディタを右クリックすると、次に示すようなコマンドを表示します。

- コピー クリップボードにデータをコピーします。
- 貼り付け クリップボードのデータを貼り付けます。

- 特殊な貼り付け... より繊細な区切り文字をコントロールし、データのプレビューを確認しながらクリップボードのデータを貼り付ける事ができます。
  - すべて選択 データエディタ内に表示しているデータをすべて選択します。これはデータにフィルタがかかるか、変数が非表示になっている場合はデータセット全体とは異なる可能性もあります。
- データ 以下の項目を含むサブメニューを表示します。
  - 変数を挿入... 新規の変数を作成するダイアログを表示し、現在のカーソル位置に挿入します。
  - 変数を追加... 新規の変数を作成するダイアログを表示し、データセットの開始または終了位置に追加します。
  - 変数の内容を変更... 選択した変数の値を置き換えるダイアログを表示します。
  - 観測値を挿入... 新規の空の観測値を作成するダイアログを表示し、現在のカーソル位置に挿入します。
  - 観測値を追加... 新規の空の観測値を作成するダイアログを表示し、データセットの開始または終了位置に追加します。
  - データをソート... 選択された変数でデータをソートします。
  - 値ラベル 値ラベルの管理のサブメニューを開きます。
  - 選択した変数のみ維持 選択したデータのみをデータセットに残します。これ以外のデータはデータセットからドロップします (取り除きます)。この操作はメモリ内のデータにのみ適用され、ディスク上のデータには影響はしません。
  - 選択範囲を削除 選択したデータをドロップします。これは選択範囲が変数全体 (列全体) または観測値全体 (行全体) である時のみドロップできます。
  - 変数を文字列から数値に変換... 文字列変数を数値変数に変換する際に使用します。なお、文字列変数に数値形式を指定する記号が含まれていると、より変換しやすくなります。
  - 数値変数から文字列変数に変換... 数値変数を文字列変数に変換する際に使用します。
  - 文字列をラベル付数値にエンコード... 文字列を値とするカテゴリー変数を数値を値とするよう符号化 (エンコード) します。その際、カテゴリーは表やグラフ内の表示は文字列のままになります。
  - ラベル付数値を文字列にデコード... エンコードされたカテゴリー変数を、元の文字列変数に変換します。
- 選択した観測値または変数を固定する 行または列をピン止めします。1 つ以上の列・変数が選択されていると、選択した変数を固定すると表示されます。1 つ以上の行・観測値が選択されている選択した観測値を固定するが表示されます。
- 選択した列の幅をリセットする 選択した列をデフォルトの幅に変更します。
- 選択した変数を非表示 選択した変数を非表示にします。
- 選択した変数のみ表示 選択した変数以外を非表示にします。
- フィルタ解除 全てのフィルタを解除し、すべての変数を表示します。

- ユーザ設定... データエディタの設定を編集します。
- 印刷... データを印刷します。

6.3 データ入力

データエディタにデータを入力するのはスプレッドシートにデータを入力することと似ています。しかし、相違点の 1 つとしてデータエディタは観測値というコンセプトがあり、データ入力をスマートに行えます。例題を使い、この内容を確認しましょう。お手元のコンピュータで操作しながら進んでください。作業を始めるには空のデータセットが必要です。必要に応じてデータセットを保存し、コマンドウィンドウに「clear」と入力してください。

メモ：コマンドウィンドウに「describe, short」(または省略して「d, s」) と入力してデータが変更されたことを確認します。データが変更されている場合は、Stata が適切なメッセージを表示します。

次に示すデータを、これから説明する方法に従って入力してみましょう。

Make	Price	MPG	Weight	Gear ratio
VW Rabbit	4697	25	1930	3.78
Olds 98	8814	21	4060	2.41
Chev. Monza	3667		2750	2.73
AMC Concord	4099	22	2930	3.58
Datsun 510	5079	24	2280	3.54
	5189	20	3280	2.93
Datsun 810	8129	21	2750	3.55

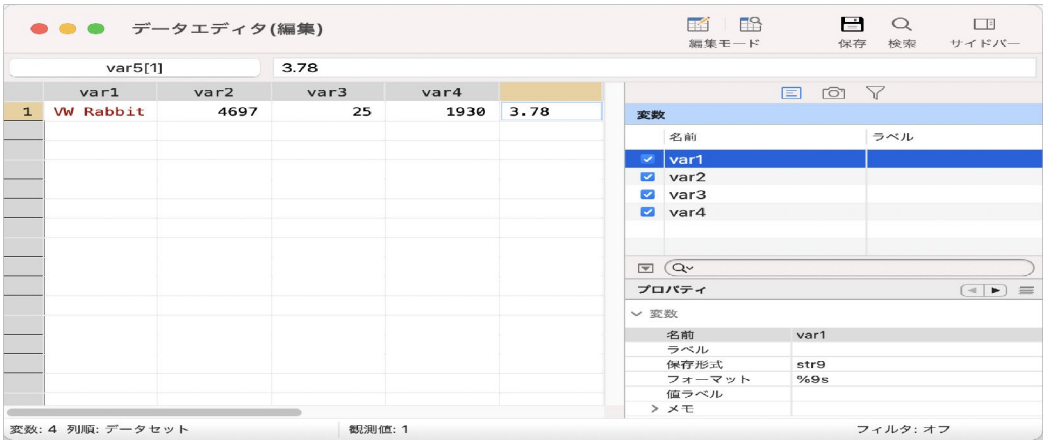
3 番目の車の MPG と 6 番目の車の Make(車種) はわかりません。

まず、データエディタを編集モードで開きましょう。データエディタボタン  をクリックするか、コマンドウィンドウに「edit」と入力すると、空のデータエディタウィンドウが開きます。既にデータが存在する場合は、コマンドウィンドウに「clear」と入力します。Stata はアクティブなセルをハイライトして表し、カーソル位置ボックス (ウィンドウ上部、左側のボックス) の隣にある入力ボックスに *varname*[*obsnum*]と表示します。後ほど詳しく見ていきますが、このセル参照機能を使ってデータセット内を移動することもできます。データエディタはデフォルトで 1 行 1 列目から始まります。まだデータが無く、変数名もないので位置ボックスには var1[1]と表示されます。

データは横 (観測値ごと) か縦 (変数ごと) に入力します。観測値ごとにデータを入力する場合、各値を入力後に Tab を押すと左から順に移動しながら入力できます。この例題の場合、「VW Rabbit」と打

った後に Tab キーを押します。そして「4697」と入力して Tab キーを押します。このように 1 番目の観測値 (行) を全て入力します。

1 番目の観測値 (行) の入力が終わったら、2 行 1 列目のセルを選択します。クリックして選択するか、キーボードを使用して移動しましょう。この時点では次のような画面を表示します。



2 行目 (観測 2) も 1 行目と同じ要領で入力します。最後の列 (var5) まで入力した後に Tab キーを押すと自動的にカーソルが 3 行 1 列目に移動します。これは 1 行目の入力の後、変数の数を Stata が認識した事によるものです。

残りのデータの入力は各セル間を Tab キーで移動しながら行い、欠損値 (空欄) に関してはデータを入力せずに、そのまま Tab で移動します。

変数 (列) ごとにデータを入力する場合、値を入力するごとに Return キーを押して下方向に移動します。欠損値の入力は 2 回連続で Return を押します。初めの変数の入力が終わったら、1 行 2 列目のセルを選んで次のデータ Price (価格) を入力します。全てのデータが入力できるまで繰り返します。

6.4 データ入力に関するメモ

データ入力に関する注意点を説明します。

- データセット内に空の行や列は作成できません。

  新しい変数や観測値を入力する際は向かって左上角から順番に入力します。もし列または行を飛ばしてしまった場合、そこは自動的に欠損値になります。

- 文字列 **(string)** と値ラベルは数値とは異なる色を表示します。

  データエディタ内の様々な変数タイプを区別するために、文字列、値ラベル (詳しくは **[GSM] 9 Labeling data (データのラベリング) をご覧ください)**、それ以外の値はそれぞれ異なる色で表示します。文字列と値ラベルの色を変更するにはデータエディタウィンドウで右クリックをしてユーザ設定**...** を選びます。

- ピリオド **(.)** は欠損値を示します。

  Stata にはシステム欠損値 (.) があり、拡張欠損値が '.a' から '.z' まであります。デフォルトではこのシステム欠損値 (.) を使用します。

- Tab キーについて。

  先の例で見たように、1 行目の観測値を全て入力すると、Stata は変数の数を認識します。2 行目の最後の観測値で Tab キーを押すと自動的に 3 行 1 列目に戻ります。

- カーソル位置ボックスはカーソル位置の表示とカーソル移動に使用できます。

  カーソル位置ボックスは現在選択しているセルの位置を表示します。例えば、var3[4]のようになっているとき、選択セルは変数 var3 の 4 行目にあることを示します。特定のセルへ移動するには、カーソル位置ボックスに変数名と行番号を入力すれば移動できます。例えば、変数 var1 の 2 行目のセルを選択する場合は「var1 2」とカーソル位置ボックスに入力して Return を押します。

- 文字列をダブルクオテーションで囲む必要はありません。

  Stata が一度でもその変数を string (文字列) であると認識すれば、たとえその値が数値に見えても、ダブルクオテーション (" ") で囲む必要はありません。例えば、郵便番号 (ZIP code) をテキストとして入力するには、1 行目は郵便番号をダブルクオテーションで囲みます ("02173") が、2 行目以降では不要です。

- 矢印キーは内容に依存します。

  セルを選択して新しいデータ入力後に矢印キーを使用すると、この変更を反映して次のセルに移

動します。セルをダブルクリックすると内容を編集できます。この場合、左右の矢印キーはセル内のデータ上でカーソルを移動させます。

- セルに施した変更点は破棄できます。

データ入力中に変更をキャンセルするときは、Esc キーを押すか他のセルを 1 回クリックしてください。

- 文字列変数のセルエディタのサイズは変更できます。

文字列変数を編集する際、セルエディタのサイズを変更し、より多くの文章を表示できます。

6.5 変数の名前およびフォーマットの変更

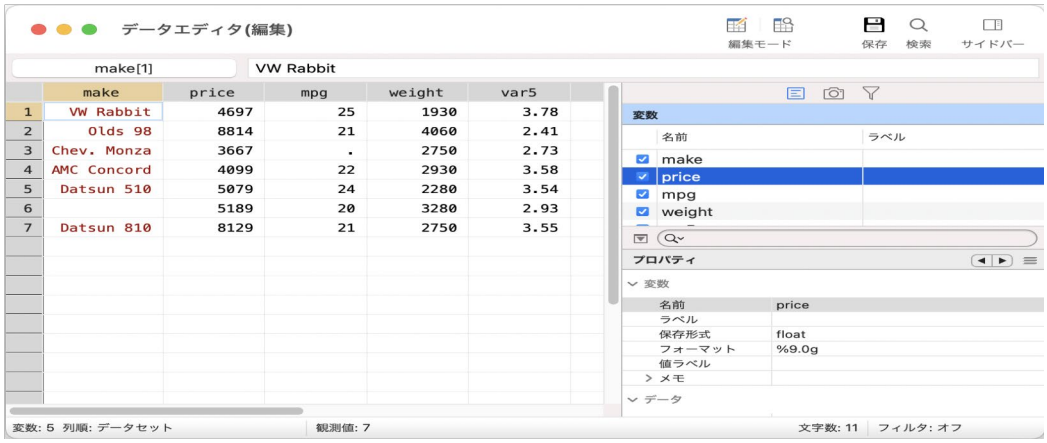
「データ入力」セクションでデータを入力しましたが、変数名がデフォルトの var1, var2, ..., ver5 となっています。これらの変数名を変更し、データセットの列タイトルに対応するようにしましょう。そして変数に説明を付け加え、さらに変数のフォーマットも変更したいと思います。

変数名、ラベル、そしてフォーマットの変更については price を例にしながら手順に沿って操作します。それから変数にメモを付けます。操作を始めるには、サイドバーで変数タブがアクティブになっていることを確認します。サイドバーに変数の一覧が表示されない場合、変数ボタンをクリックします。以上の準備ができたら、変数 var2 を変数ウィンドウでクリックします。すると、変数 var2 のプロパティがプロパティウィンドウに表示され、編集可能になります。これから順に変数 var2 のプロパティを変更しましょう。

1. 名前欄で var2 をクリックします。選択後「price」と打ち込んで名前を上書きします。
2. price の下のラベル欄をクリックします。
3. 変数を説明するラベルを設定します。例えば「Price in dollars」と入力します。
4. フォーマット欄の%9.0g をクリックします。
5. クリックすると表示される省略符号 **(...)** をクリックします。書式作成ダイアログが開きます。
6. ここには多くのフォーマットがあり、中でも時間に関係したものが大多数を占めます。数字の中にカンマ **(,)** を入れたいので書式のプロパティボックス内の出力する数値をカンマで区切るのチェックボックスにチェックを付け、**OK** ボタンを押します。
7. メモ欄をクリックします。
8. クリックすると表示される省略符号 **(...)** をクリックします。**price** のメモというダイアログが開きます。
9. 追加ボタンをクリックし、何かメモを入力します。
10. 入力が終わったら、適用ボタンに続けて閉じるボタンをクリックします。変数 price にメモが付きました。
11. メモの前に表示される三角ボタンをクリックするとプロパティウィンドウに先程作成したメモを表示します。

他の変数を編集するには変数ウィンドウで目的の変数をクリックするか、またはプロパティウィンドウでナビゲーション矢印をクリックして目的の変数を表示します。var1 を「make」、var3 を「mpg」、

var4 を「weight」、var5 を「gear ratio」に変更します。var5 を「gear ratio」に変更する直前の画面の様子は次の通りです。



ここで、変数名に関するルールをいくつか確認しましょう。

- 大文字と小文字は区別します。

  Make, make, MAKE はすべて異なる名前になります。変数名を Make、Price、MPG 等のように大文字も使用して入力した場合、今後、同じように大文字で入力する必要があります。全て小文字で入力する方が簡単でしょう。
- 変数名は 1-32 文字の長さで入力してください。
- 使用できる記号は半角英数 (A-Z, a-z, 0-9)、アンダースコア (_)、記号以外の Unicode 文字です。
- スペースや他の記号は使用できません。
- 変数名の 1 文字目はアルファベット、アンダースコア、Unicode 文字のいずれかにします。変数名の 1 文字目にアンダースコアを使用出来ますが、これはお勧めしません。Stata では一時的に利用する変数名には先頭にアンダースコアを付けるという流儀があります。したがって普通の変数の先頭にアンダースコアを付けることは避けるべきでしょう。

変数名と値ラベルに関しての詳細は **[GSM] 9 Labeling data (データのラベリング) をご覧ください)** を、表示形式に関しての詳細は **[U] 12.5 Formats:Controlling how data are displayed** を参照してください。

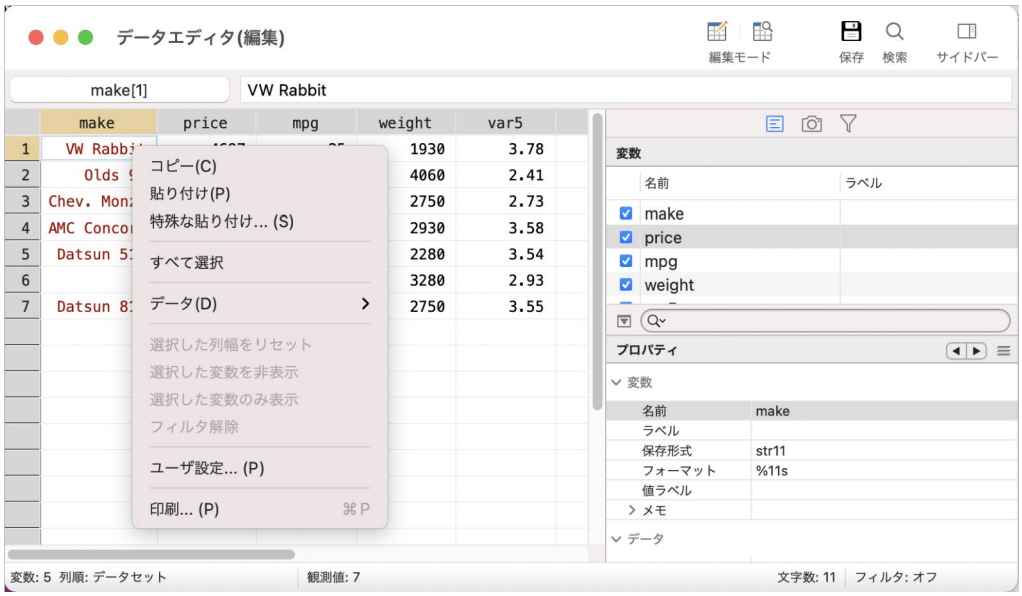
6.6 データのコピーと貼り付け

データのコピーと貼り付けにはデータエディタを使用します。他のスプレッドシートやデータベースからデータを移動する場合はこの方法を利用すると簡単です。

1. 以下の方法のうち 1 つを使い、Stata にコピーする内容を選択します。
   - 変数名または列ヘッダーを 1 回クリックして列を選択します。
   - 行番号または行ヘッダーを 1 回クリックして行を選択します。
   - マウスをクリック＆ドラッグしてセルの範囲を選択します。
2. 選択した範囲内で右クリックを行い、コピーを選んでクリップボードにコピーします。
3. クリップボードのデータを貼り付けるセル範囲の左上角を選択します。そのセル内で右クリックして貼り付けを選びます。

コピーと貼り付けを説明するために先程の例で作成したデータの 1 行目をコピーし、データセットの最後に貼り付けましょう。

まずは行番号 (一番左側の番号) をクリックします。これで行のデータを全て選択します。続けて右クリックし (マウスを動かす必要はありません) 、コピーを選択します。



8 行 1 列目を選択し、右クリックします。そして表示されたメニューから貼り付けを選びます。1 行目がうまく複製できたことが分かります。

6.7 コピーと貼り付けに関するメモ

- 先ほどの例はデータエディタ内でのコピーと貼り付けに関するものです。この方法とほぼ同じ方法で Stata と他のアプリケーション間でコピーと貼り付けを行えます。Stata と他のアプリケーションの間でコピーと貼り付けができるかどうか確認するには、実際にやってみると良いでしょう。表計算ソフト、データベースソフト、ワープロソフトなど、コピー元のデータに何か明確な区切りがある場合はうまくコピーできるはずです。編集 > 特殊な貼り付け**...** を使うと、他のフォーマットにも対応するオプションが提供されます。単純な貼り付け (paste) コマンドでうまくいかない時はこの、編集 > 特殊な貼り付け**...** を試してください。ファイルをインポートする場合についての詳細は **[GSM] 8 Importing data (**データのインポート**)** をご覧ください。
- 値ラベルがついているデータのコピーと貼り付けには選択肢があります。つまり、値ラベルをテキストと貼り付けることができますし、元の数値データを貼り付ける事ができます。値ラベルをテキストとしてコピーするのがデフォルトの設定です。他のオプションを使用するにはデータエディタ内で右クリックをしてデータ > 値ラベル > すべての値ラベルを非表示と操作するか、メインメニューからツール > 値ラベル > すべての値ラベルを非表示と操作してください。

6.8 データを変更する

名前から予想できるようにデータエディタはデータセットの編集を行うツールです。先の例で示したように、データの編集、変数の説明や表示オプションの編集を行えます。

実際に auto のデータセットに変更を加えて、データエディタとそのコマンド履歴の使用方法を見てみたいと思います。このセクションではテキストのスナップショットも取りながら作業します。これで何かを間違えた場合は、間違える前のデータセットに戻すことができます。

これからデータセットの調査、値ラベルの作成、変数 trunk の削除、100 マイル (約 160km) あたりのガソリン消費量を示す新しい変数の作成を行います。この一連の操作でデータエディタを使用する時の基礎を学べます。

はじめに、コマンドウィンドウに「sysuse auto」と入力します。このセクションの前に例題を行っている場合、エラーが表示されてメモリ内のデータは失われると表示されます。このメッセージはメモリ上にある編集済みのデータを削除する前に、ユーザーに保存の確認を求めるものです。既存のデータセットを保存する場合はファイル > 保存と操作してファイルを目的のフォルダに保存します。保存しない場合はコマンドウィンドウに「clear」と入力して Return を押し、データを削除します。その後、 auto データセットをロードしてください。

auto のデータセットをロードしたらデータエディタを表示してください。

1. 祖父の乗っていた車が Toronado でした。お洒落な車である一方、大量にガソリンを必要とした気がします。この車がデータセットに掲載があるか調べるとします。編集 > 検索 > 検索... を選択し、Toronado と入力して Return を押します。その結果、この車の情報を発見し、燃費が 16 mile/gallon であったことが分かります。
2. この中で最も燃費が良い車と悪い車を調べましょう。まず、mpg 列の列ヘッダーを右クリックします。そして、コンテキストメニューからデータ > データをソート... を選択します。どの様にソートを行うのか尋ねるダイアログが表示されます。デフォルトは昇順です。**OK** をクリックします。(Stata にはリサンプリング機能があるので、並び替えを行う場合は注意が必要です。) データが mpg の昇順で並べ替えられます。燃費が最も悪い車 (mpg の数値が小さい車) がスクリーンの一番上にあります。データをスクロールダウンするとデータセットの最後に燃費が最も良い車があります。
3. 修理記録も比較したいので、次は変数 rep78 でソートしてみましょう。 (今行ってください。) Starfire と Firebird の修理記録は良くありません。ですが今は修理記録が良い車を調べたいので、この画面では分かりません。データセットの下までスクロールダウンもできますが、今回はカーソル位置ボックスを使う方が速いです。「rep78 74」と入力し Return を押すと、rep78[74] セルがアクティブになります。rep78 の最後の 5 つのセルは点 (.) になっており、これは欠損値を表します。ここで、いくつか注意点があります。
    - ソート結果から分かるように、Stata は欠損値を数値より大きいものとして扱います。技術的に書くと、「rep78 >= . 」は missing(rep78) と同じことになります。
    - このデータを見ただけでは分からないこととして、Stata は複数の欠損値指標を使用します。(.) は Stata のデフォルトまたはシステム欠損値指標で、.a, .b, ..., .z は Stata の拡張欠損値です。拡張欠損値は欠損の理由を区別するのに役立ちます。
    - 各欠損値指標は欠損値内でソートし、(. < .a < .b < ... < .z) のようになります。 [U] **12.2.1 Missing values** をご覧ください。
4. この変数 rep78 の値に意味を持たせたいと思います。変数ウィンドウの rep78 をクリックします。
5. プロパティウィンドウの値ラベル欄をクリックし、省略符号 (...) のボタンをクリックします。これで値ラベルの管理ダイアログを開きます。では新しい値ラベルを定義しましょう。
    (a) ラベルを作成するボタンをクリックします。するとラベルを作成するダイアログが開きます。
    (b) ラベル名、例えば「repairs」をラベル名ボックスに入力します。
    (c) Tab キーを押すか値ボックス内をクリックします。
    (d) 値に 1 と打ち込んで、Tab キーを押します。そして「atrocious」をラベルに入力します。
    (e) Return キーを押して値とラベルのペアを作成します。
    (f) ステップ d と e を繰り返してすべてのペアを作成します。2 を「bad」、3 を「OK」、4 を「good」、5 を「stupendous」として作成しましょう。
    (g) **OK** をクリックして値ラベルを作成するのを終了します。

(h) 折りたたみ解除ボタン ▶ をクリックして作成したラベルを確認します。
間違いがある場合、ラベルを編集するボタンを押してラベルを編集します。

(i) 閉じるボタンを押して値ラベルの管理ダイアログを閉じます。

ラベルを作成したので、変数 rep78 にラベルを付けます。値ラベル欄の右側にある下矢印ボタンをクリックして **repairs** ラベルを選択してください。すると値の代わりにラベルを表示します。

6. 仮に欠損値ではなく、元のデータが分かった時は rep78 にその値を入力できます。その場合、直接セルに値を入力すれば大丈夫です。または欠損値のセル内を右クリックし、データ > 値ラベル > 変数 **'repairs'** に値ラベルを割当と選択してラベルを選択します。値ラベルに複数の選択肢があるときに便利です。
7. 次に変数 trunk を削除します。変数 trunk の変数名 (列の一番上) で右クリックを行い、表示するコンテキストメニューでデータ > 選択範囲を削除を選びます。データをメモリ上から削除する場合は、確認のダイアログを表示します。そこで、はいボタンをクリックします。
8. 最後に、100 マイル (約 160km) を走るのに必要なガソリンの量を表す変数を 1 つ作成しましょう。
   (a) どこのセルでも構わないので右クリックをし、データ > 変数を追加...を選択して **generate** ダイアログを開きます。
   (b) 変数名欄に「gp100m」と打ち込みます。
   (c) 値または式を指定するのラジオボタンが選択されていることを確認してから「100/mpg」と入力します。右隣りにある作成... ボタンを押すと数式ビルダダイアログを表示しますが、この数式は簡単なので直接入力します。(ここでちょっと休憩をして数式ビルダの機能を確認するのも良いでしょう。)
   (d) 新たな変数の位置欄には、データセットの最後に追加するとあるのを確認します。
   (e) **OK** をクリックします。右にスクロールすると新しく作成した変数を確認できます。

このデータ編集セッションではデータエディタを使用してデータを編集してきました。結果ウィンドウを見てみると、コマンドとその出力結果を確認できます。また、履歴ウィンドウではデータエディタで実行したコマンドも見ることが可能です。編集コマンドを後で使用する場合は、次の操作で保存します。

1. データエディタが最後に出力したコマンドを履歴ウィンドウ内でクリックします。
2. データエディタを開いてからすぐに実行したコマンド「sort mpg」をスクロールで探し、Shift+クリックしてこれまでの内容を選択します。
3. 選択したコマンド上で右クリックします。
4. 選択範囲を **do** ファイルエディタへ送るを選びます。

このように操作することによって、選択したコマンドを do ファイルエディタに保存できます。この後、再実行できます。do ファイルエディタについては **[GSM] 13 Using the Do-file Editor—automating Stata (do ファイルエディタを使用する—Stata の自動化)** で説明します。do ファイルについてのヘルプは **[U] 16 Do-files** をご覧ください。

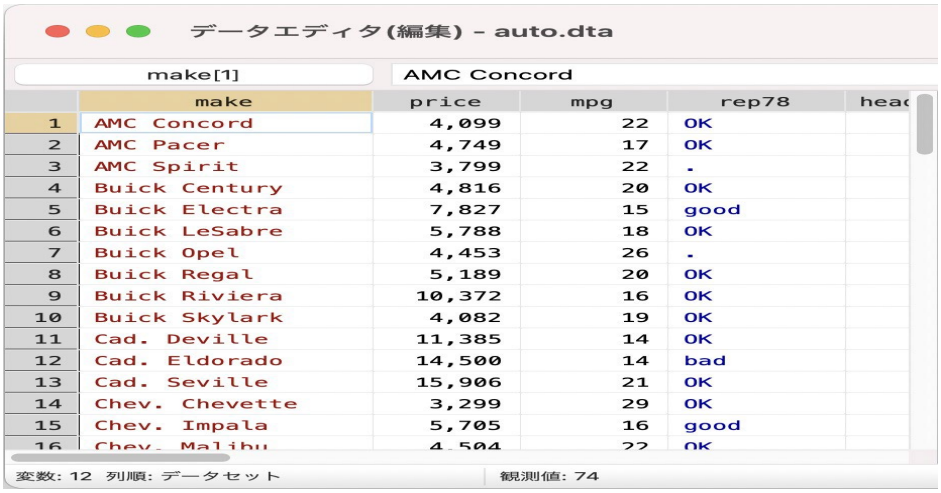
もしこのデータセットを保存する場合はメインメニューでファイル > 名前を付けて保存**...** と操作し、新しい名前を付けて保存します。元のデータセットを上書きしないように注意しましょう。

6.9 スナップショットで作業する

データエディタでは作業中のデータセットならいつでもスナップショットを残すことができます。スナップショットは Stata を閉じる時に削除されるので、一時ファイルの扱いになります。それでもスナップショットの用途は幅広く、次のような使用例があります。

- データを一時的なコピーとしてメモリに残して、他のデータセットを開いて見る事ができます。
- 作業内容を保存できるので、万一データを失くした時でも再現できます。
- 分析中のデータセットを自由に加工して保存できます。

引き続き auto データセットを使用します。このセクションから作業を開始する場合はコマンドウィンドウに「sysuse auto」と入力してデータセットを開きます。(メモリ内のデータが失われることになりますというエラーメッセージを表示した場合、「clear」をするか既存データを保存してください。詳しくは **[GSM] 5 Opening and saving Stata datasets (Stata のデータセットを開く・保存する)** をご覧ください。) データエディタを開いてサイドバーのスナップショットボタンをクリックすると、次のようなウィンドウが開きます。データセットを開くところから作業を始めている場合は、変数 rep78 は値ラベルではなく数値を表示します。



開始直後、スナップショットツールバーにはアクティブなボタンは 1 つしかありません。アクティブなボタン、追加ボタン  をクリックします。するとダイアログが開きますので、スナップショット用の名前 (ラベル) の入力を行います。今回は「Start」というラベルを付けましょう。そして Return を押します。スナップショットはリストに追加され、ツールバーにある他のボタンもアクティブになります。スナップショットウィンドウには次のボタンが表示されます。

 追加 時間スタンプとラベルの付いた新しいスナップショットを保存します。

 はずす スナップショットを削除します。これは一時的なスナップショットファイルを削除しますがメモリ内のデータには影響しません。

 ラベルの変更 選択したスナップショットのラベルを編集します。

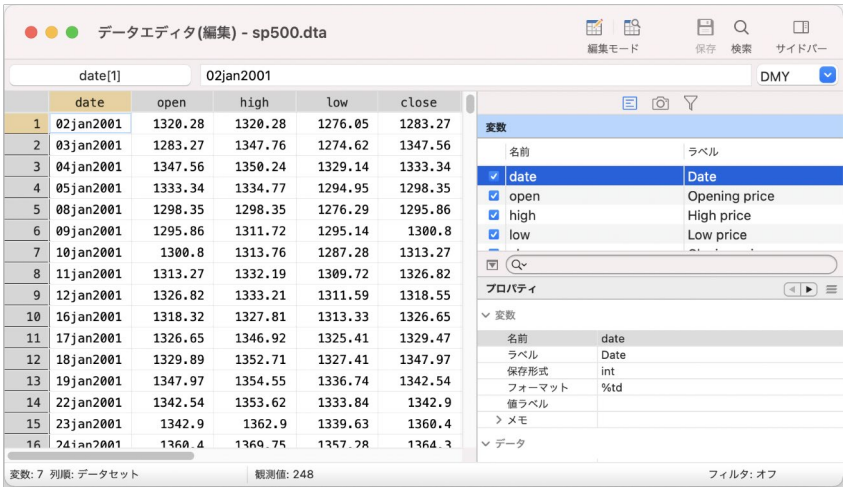
 リストア メモリ内のデータを選択したスナップショットのデータで置き換えます。データを置き換える場合、操作を確認するダイアログを表示します。

ではいま紹介したデータエディタのツールを実際に使用してデータセットの内容を変更してください。変更後に新たなスナップショットを作成し、「Changed」とラベルを付けます。スナップショットウィンドウを開いて「Start」を選択し、ダブルクリックをするかリストアボタンを押して「Start」を復元します。そして先程の変更したファイルに戻るには「Changed」スナップショットを復元します。

これらのスナップショットは、削除するか Stata の該当セッションを閉じない限り使用できます。つまり、1 つのデータセットで作業しながら、別のデータのスナップショットを開くことができます。スナップショットの用途は千差万別なので、ご自身に合ったものを探してみてください。ただし、スナップショットは一時的なファイルなので、後から利用するデータはしっかりと保存してください。

6.10 データエディタと日付

データエディタには日付を取り扱う 2 つの特殊なツール (フォーマットと日付マスク) があります。これらの動作を確認するには他のデータセットを開く必要があります。データセットを保存するか clear をして、コマンドウィンドウに「sysuse sp500」と入力します。データエディタを開いて、内容を見てみましょう。



	date	open	high	low	close
1	02jan2001	1320.28	1320.28	1276.05	1283.27
2	03jan2001	1283.27	1347.76	1274.62	1347.56
3	04jan2001	1347.56	1350.24	1329.14	1333.34
4	05jan2001	1333.34	1334.77	1294.95	1298.35
5	08jan2001	1298.35	1298.35	1276.29	1295.86
6	09jan2001	1295.86	1311.72	1295.14	1300.8
7	10jan2001	1300.8	1313.76	1287.28	1313.27
8	11jan2001	1313.27	1332.19	1309.72	1326.82
9	12jan2001	1326.82	1333.21	1311.59	1318.55
10	16jan2001	1318.32	1327.81	1313.33	1326.65
11	17jan2001	1326.65	1346.92	1325.41	1329.47
12	18jan2001	1329.89	1352.71	1327.41	1347.97
13	19jan2001	1347.97	1354.55	1336.74	1342.54
14	22jan2001	1342.54	1353.62	1333.84	1342.9
15	23jan2001	1342.9	1362.9	1339.63	1360.4
16	24jan2001	1360.4	1369.75	1357.28	1364.3

データの最初の日付として 2001 年 1 月 2 日 (January 2, 2001) と記載してある日付変数 date があります。この表示形式は Stata のデフォルト形式です。

まずは日付形式を変更することから始めましょう。

1. データエディタウィンドウの右側にある変数ウィンドウ内で変数 date を選びます。
2. プロパティウィンドウでフォーマットを選び、省略記号 (...) ボタンをクリックします。
3. 書式作成ダイアログからこの日付形式について 3 つの情報を読み取れます。
    - このデータは日次 (daily) の時系列データです。ダイアログを見ても分かるように、Stata は金融業界等で使用する日付の種類にも対応します。
    - ダイアログの下には、Stata のデフォルト日付形式として「%td」を表示します。このフォーマット記号のついている変数は日次の時系列データであると解釈します。
    - このデフォルト形式は、例えばの 07apr2021 ように表示されます。(これはデータエディタで見ると一目でわかります。)
4. 書式作成ダイアログの右上にあるサンプルにはさまざまな日付形式が用意されています。ここではその中から「April 07, 2021」をクリックします。どのようにこの形式が記号で記述されるのかをダイアログの下で見ることができます。

5. **OK** をクリックして書式作成ダイアログを閉じます。日付の表示形式が変更されました。

この方法で日付形式を簡単に変更できます。日付と日付形式に関する情報は [D] **datetime** をご覧ください。

それでは、日付をいくつか変更し、形式の変更を簡単に行える様子を確認しましょう。これは実際の表示形式がどのような場合でも同じです。データエディタの右上角を見ると、日付マスク (date mask) エリアがあり、そこには DMY と記載されています。データ編集中の日付表示形式はここで変更します。

デフォルトでは、このマスクは DMY となっています。これは日付の入力順番が日・月・年ならば、日付は様々な様式で入力できることを示しています。実際にやってみましょう。

1. 変数 date の 1 行目の観測値をクリックします。するとカーソル位置ボックスには date[1]と表示します。
2. 「18jan2021」と入力して Return キーを押します。Stata は DMY 日付マスクを理解し、選択したセルに新しい日付を入力します。
3. 「30042021」と入力して Return キーを押します。今回、デリミタ (区切り) はありませんが、日付マスクは入力した情報を正確に認識して表示します。
4. 時間／日付入力マスクエリア内をクリックし、ドロップダウンメニューから MDY を選びます。
5. 変数 date 内にある観測値のどれかをクリックします。
6. 「March 15, 2021」と入力して Return キーを押します。すると、他の観測値と同じ形式で表示します。

結果ウィンドウには、データ編集のコマンドを表示します。この場合、データエディタへの直接入力の方が効率的です。

次は他のデータセットを利用しますので「clear」と打ち込み、Return を押しましょう。

6.11 データエディタのアドバイス

先の例からも分かるように、データエディタ内の小さな間違いがデータセットに大きな問題を引き起こすこともあります。データ編集の方法には細心の注意を払わなければなりません

- 生データの管理に気を配る人はデータエディタのような編集機能は危険だと感じるはずです。なぜなら、うっかりデータを変更してしまう可能性があるからです。データを表示するだけなら、データエディタの編集モードを使わないでください。代わりにデータエディタのブラウズモードを使用してください (あるいは browse コマンドを使用してください)。

- データを編集する場合、データセットの表示領域を制限すれば誤ってデータを変更することはありません。例えば、rep78 の欠損値だけを直す必要がある場合、その欠損値だけを表示するように工夫してください。これで編集する変数および観測値以外の値は変更（破損）できなくなります。この方法は後で実際に試します。
- 注意をしていても間違いは起きてしまうことがあります。それでもデータエディタは結果ウィンドウにコマンドの記録があるので他のソフトよりもデータの破損等に関しては安心です。この機能を使って出力結果をログとして記録し、変更の記録を作ることができます。この記録から変更が意図したものと一致しているか確認できます。ログファイル作成についての情報は **[GSM] 16 Saving and printing results by using logs (**ログを使い結果の保存や印刷を行う**)** をご覧ください。

6.12 フィルタと非表示

これからデータエディタ内で表示制限する方法を紹介します。データエディタのアドバイスセクションで説明した表示制限に使用するだけでなく、大きなデータセットの内容を見る際にも役立ちます。どちらの場合でも、データ全てではなくデータの一部、例えば変数や観測値の一部を表示する時にとても便利です。更に、例題の中で変数の順序も変更します。この操作方法はグラフィカルなインターフェイスとコマンドの両方で紹介します。

コマンドウィンドウに「sysuse auto」と打ち込んで、データセット auto を開きます。もしエラーメッセージが出てきたら、clear を実行して再度試してください。データセット auto をうまく読み込んだら、データエディタを開きます。

例えば、変数 rep78 の欠損値のみを編集するとしましょう。どの観測値について作業をしているのか確認するために車のメーカー (make) も表示しますが、他の変数は必要ありません。ここではとても大きなデータセットで作業を行っていると仮定して操作します。

1. 作業を始める前にデータエディタウィンドウの変数ウィンドウで次の操作を試してください。
   (a) 変数をリスト上でドラッグ& ドロップして配置を変えます。するとデータエディタ内の変数列の順番が変わります。データセットの元の順番には変更ありません。
   (b) 1 列目のチェックボックスからチェックを数個外し、データエディタ内でその変数を非表示にします。
   (c) ウィンドウ上部の検索欄にキーワードを入力します。メインウィンドウの変数ウィンドウのように、デフォルトでは変数または変数ラベルについて、大文字と小文字の区別をつけずにフィルタ内の言葉について検索します。虫眼鏡マークをクリックすると、この挙動を変更できます。変数ウィンドウで行うフィルタリングは変数ウィンドウ上の表示を変更するもので、観測値の表示は制御しません。一通り終了したらフィルタのテキストを消去します。

2. 変数ウィンドウの変数（どれでも構いません）を右クリックしてコンテキストメニューからすべて選択を選びます。
3. 一度、チェックのついているチェックボックスの 1 つをクリックし、すべてのチェックを外します。
4. 変数 make をクリックして選択し、他の全ての変数の選択を外します。
5. 変数 make のチェックボックスをクリックします。
6. 変数 rep78 のチェックボックスをクリックします。

結果ウィンドウを見ると、コマンドとしては何も実行していません。データエディタで変数を非表示してもデータセットには影響しません。データエディタでの表示だけに影響します。

これで作業する変数だけを表示できたので、間違える可能性を減らせました。では変数 rep78 で欠損値となっている観測値を表示しましょう。これは簡単に行えます。

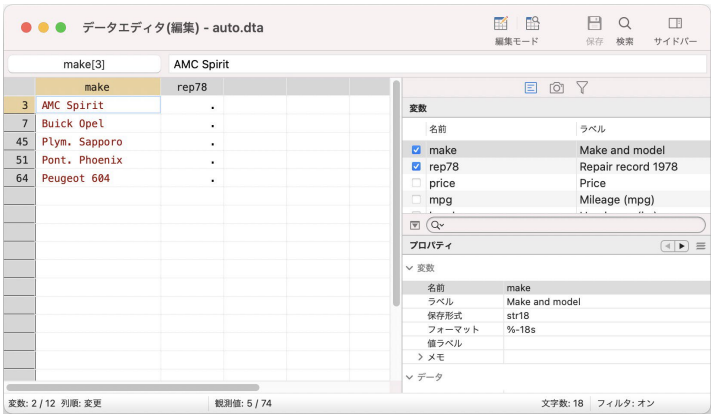
1. サイドバーの観測値フィルタタブ  をクリックします。
2. 式によるフィルタ欄に「missing(rep78)」と入力します。
3. フィルタを適用するボタンをクリックします。
4. 関数の表記がよくわからない時は式によるフィルタ欄右側にある省略記号（...）をクリックしてください。数式ビルドダイアログを開きます。このダイアログは Stata で使用できる関数のリストを表示します。詳しくは Stata Functions Reference マニュアルをご覧下さい。

これで作業を行う部分だけを表示しました。些細なキーボード操作のミスなどで誤ってデータを削除または変更する可能性も無くなりました。

コマンドウィンドウからデータエディタ内の変数を非表示にし、観測値にフィルタをかける方法も確認しましょう。このコマンドを使用すると先程行ったような表示制限を簡単に行えるようになります。コマンドウィンドウから作業を行う場合は、edit コマンド、*varlist*（変数リスト）、if と in 条件を使用してコマンド文を入力します。*varlist* で表示する変数を制限し、if と in 条件で観測値の制限を行います（**[GSM] 10 Listing data and basic command syntax (データのリストと基本コマンドの構文)** にはコマンド文と共に *varlist* と if および in 条件の使用についての例が多くあります)。では変数 rep78 の欠損値を修正しましょう。最低限必要な情報は変数 make と変数 rep78 です。最小限の表示で操作を行い、間違える可能性をできるだけ小さくするには次のコマンド文を入力します。

```
. sysuse auto
(1978 automobile data)
. edit make rep78 if missing(rep78)
```

実行すると、次のような画面が表示されます。



これで間違える可能性を大幅に減らせました。

ここで学んだ内容を忘れずに、今後のデータ編集を行ってください。

6.13 ブラウズモード

誤ったキー操作等でデータを変更したくない時は、データエディタのブラウズモードを使用します。データエディタをモードで開くためにはデータエディタ **(**ブラウズ**)** ボタン  をクリックするか、コマンドウィンドウで「browse」を実行します。ブラウズモードで作業をしているとき、データを変更できるすべてのコンテキストメニュー、変数のラベル、またはディスプレイ形式に関する操作はできなくなっています。変数のプロパティはプロパティウィンドウで見ることはできますが、変更はできません。しかし、観測値や変数を非表示にして表示領域を制限することはできます。この操作はデータセットの変更を伴わないのでブラウズモードでも実行できます。

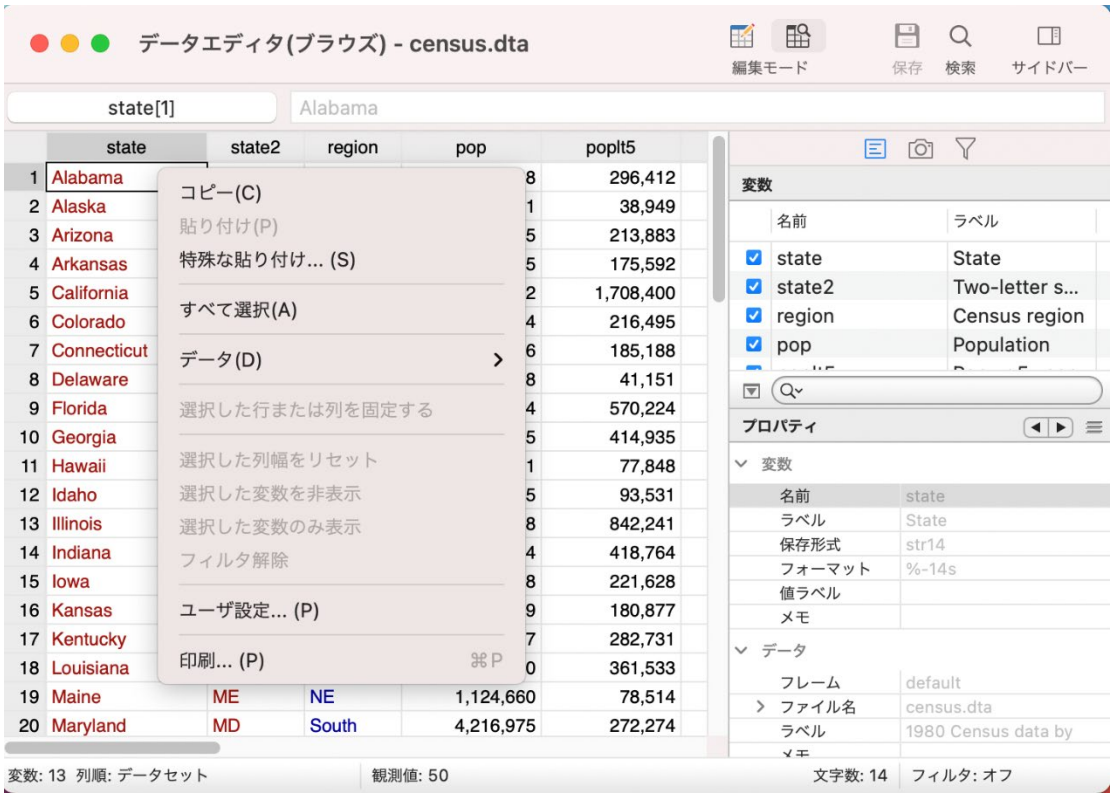
メモ：データエディタがブラウズモードでも、Stata のメニューやコマンドウィンドウを使用して操作すればデータは変更できます。実行したコマンドがどのようにデータセットに影響するのか、直接見ることができます。データエディタの編集モードは本当にデータを変更するときにのみ使用してください。

6.14 列ヘッダの変数ラベルと列幅の制御

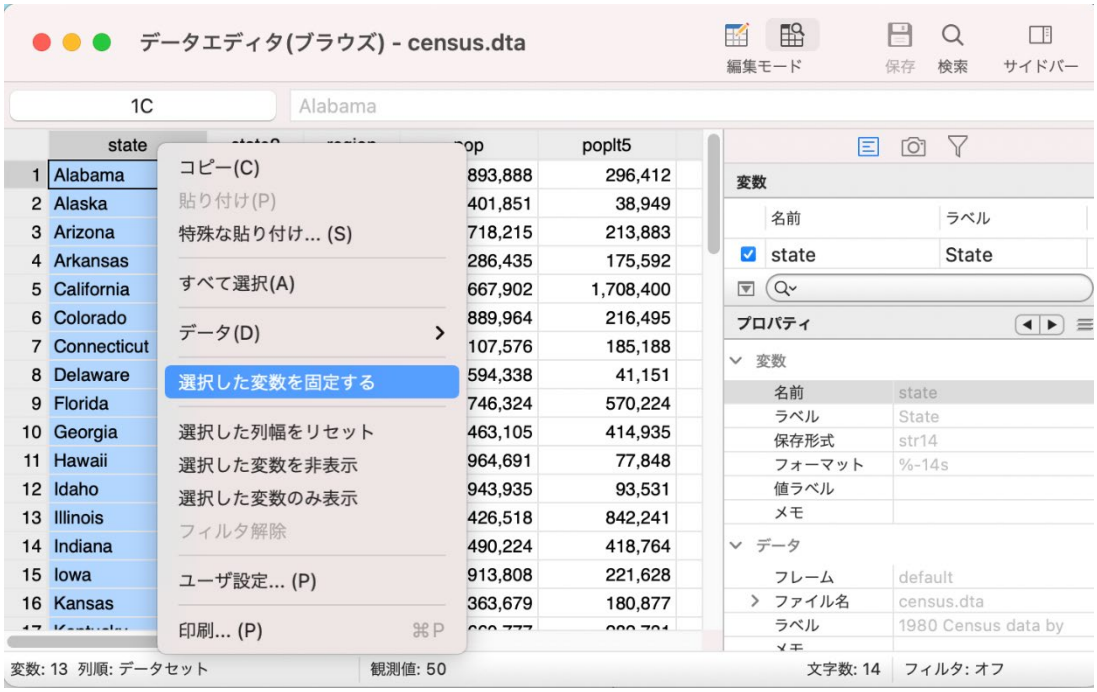
変数ラベルは列ヘッダにも表示できます。表示 ＞ データエディタと操作し、列ヘッダに変数ラベルを表示、にチェックを入れます、列間で左右の矢印をドラッグして列を広げることができます。
列を広げて、変数ラベルがすべて表示されるようにしましょう。列の幅はデータセットを保存する際に記録されます。列の幅より文字列変数の値が長い場合、セルに表示される値は省略されます。データエディタでは省略されたセルの表示のためのツールチップがあります。マウスポインタをセル上に移動すると、省略されている全文を確認できます。または、セルをダブルクリックすることでサイズ変更が可能なセルエディタを表示できます。

6.15 行と列のピン止め

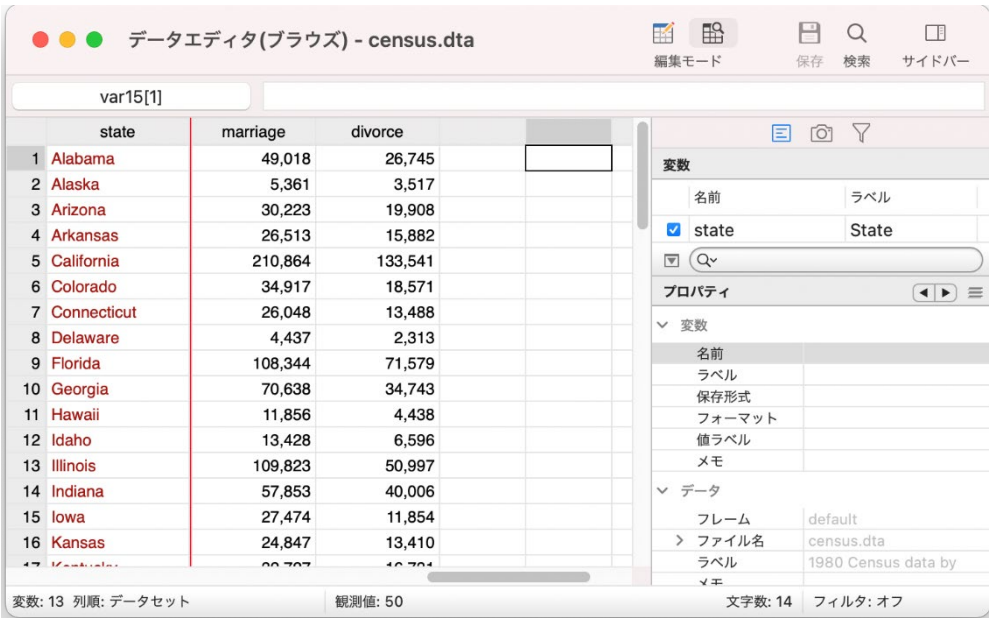
データエディタでは行と列をピン止めできます。ピン止めされた行と列は、データを移動してもスクロールされず、表示され続けます。sysuse census を実行し、データセットを変更しましょう。これは 1980 年における、米国の州ごとの国勢調査のデータです。データエディタを閲覧モードで起動します。データには 50 州がありスクリーンサイズは制限されるので、全観測値と変数を一度に表示できないことがあります。さらに、データを閲覧する際、いくつかの州または観測値をピン止めし、違いを確認したいことがあるでしょう。次でいくつかの例を紹介します。
いずれかのセルを右クリックすると、選択した行または列を固定する、がグレーアウトしていることがわかります。



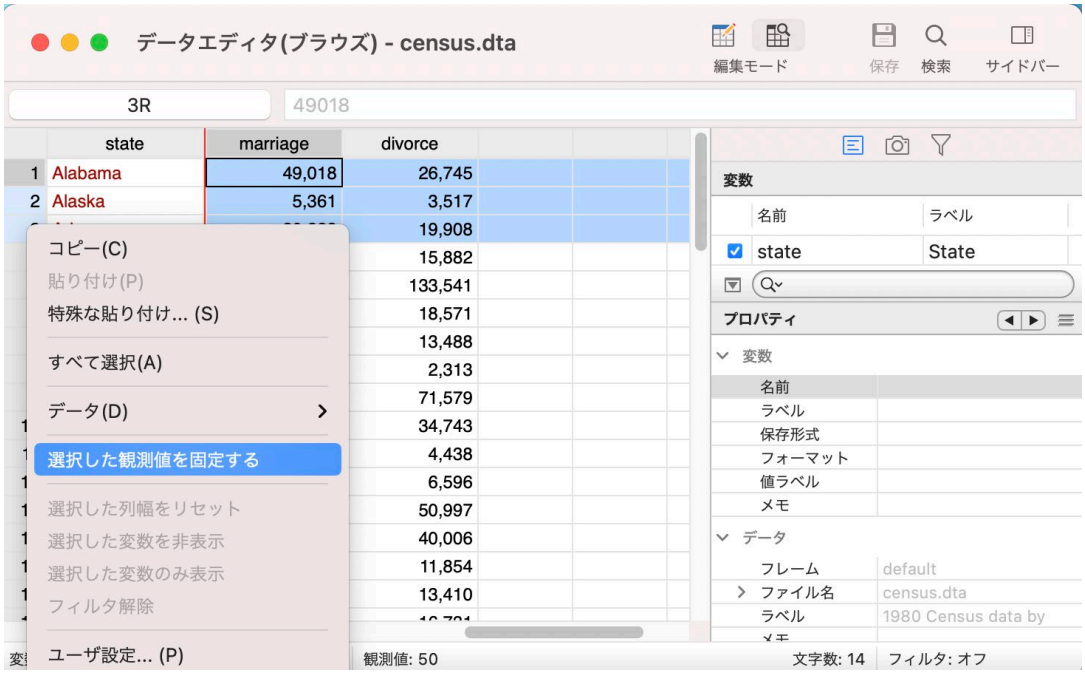
例えば state といった、特定の列をハイライトするために列ヘッダをクリックすると、同じようなメニューが表示されますが、選択された変数を固定、が有効となっています。1 つ以上の変数をまとめて選択し、ピン止めすることもできます。



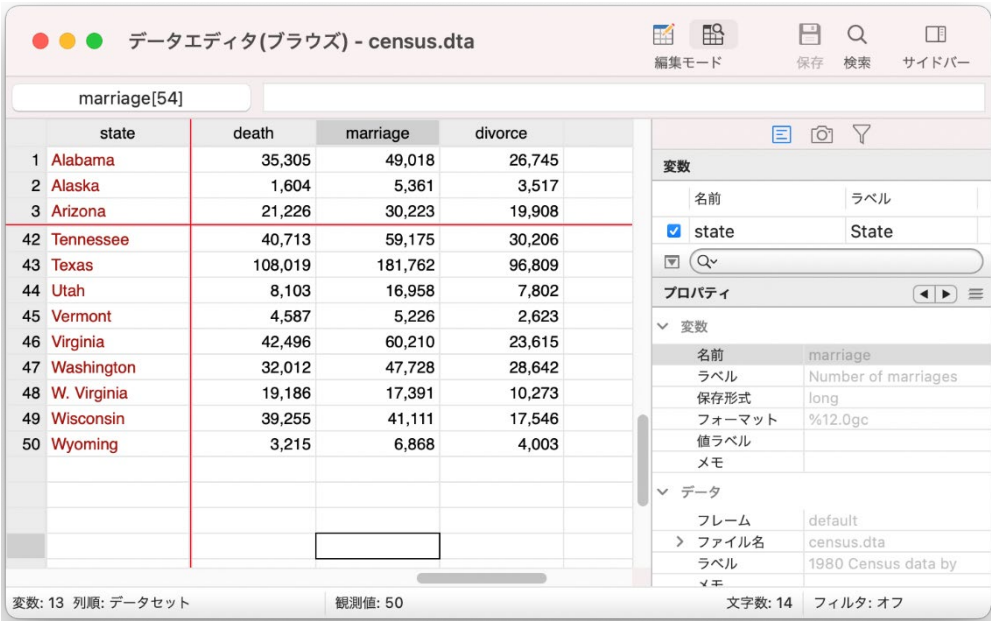
州の情報をピン止めした後、ウィンドウに表示されていない、marriage や divorce といった変数を確認するため右方向にスクロールしても、州名は左側にピン止めされたままです。



同様に、1 つ以上の観測値をハイライトし、右クリックして特定の列をピン止めすることもできます。



1 から 3 行目までの観測値をピン止めし、垂直にスクロールして他の州を比較してみましょう。

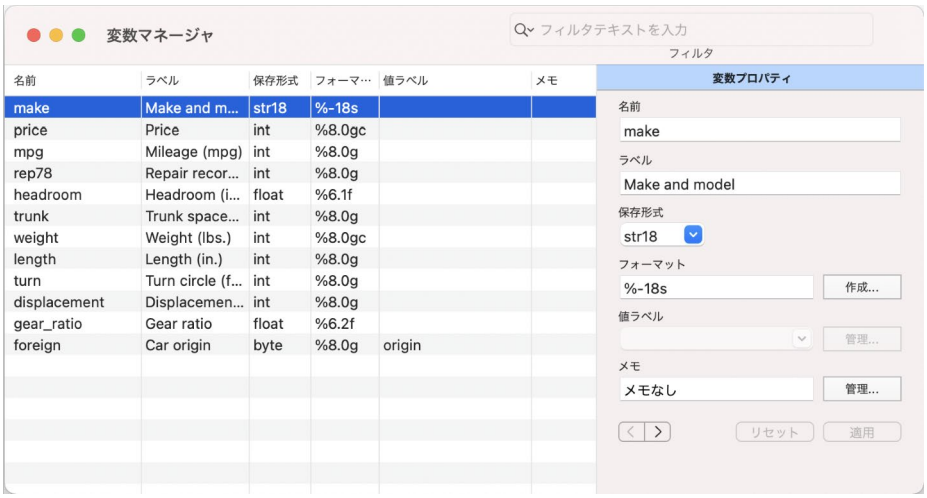


ピン止め機能は閲覧モードと編集モードのどちらでも利用できます。行と列は同時に固定することができます。いずれかの行または列右クリックし、固定した変数または観測値をクリア、を選択しピン止めを解除します。

7 変数マネージャを使用する

7.1 変数マネージャ

この章では Stata の変数マネージャについて説明します。まずは、コマンドウィンドウで sysuse auto, clear を入力して automobile データセットを開きます。変数マネージャを開くにはメインメニューからデータ > 変数マネージャと操作します。



変数マネージャは変数のプロパティを管理するもので、1 つまたは複数の変数を同時に管理できます。変数マネージャでは変数および値ラベルの作成、変数名の変更、表示形式の変更、そしてメモの管理を行えます。また変数をフィルタにかけてグループにまとめることも、変数リストを作成することもできます。この機能は大きなデータセットを管理する時に便利です。

変数マネージャで行った操作は、コマンドウィンドウに直接入力したように結果ウィンドウに表示します。つまり、操作の記録を取ると同時に、変数マネージャを利用することによってコマンドを学べます。

7.2 変数ペイン

変数マネージャの左側ペインは、スクリーン上では特に記載されていませんが、変数ペインと呼ばれます。このペインはデータセット内の変数リストを表示します。このリストは様々な方法で表示できます。

- 左上角にあるフィルタボックスにテキストを入力すると変数にフィルタをかけます。これは似たような名前がついている変数および値ラベルのみを表示するのに役立ちます。
- リストは列タイトルをクリックするとソートできます。

1. (a) 列タイトルを 1 回クリックすると昇順で表示します。
   (b) 同じ列タイトルを 2 回クリックすると変数を降順でソートします。

(c) 同じ列タイトルを 3 回クリックすると、ソート順を元の変数の並び順に戻します。

- ソート順は変数マネージャウィンドウの表示のみ変更します。データセット自体の順番は変更出来ません。
- 列タイトルをドラッグ＆ドロップで移動すると列の順番を変更できます。デフォルトの順番に戻すには、列タイトルを右クリックして列をデフォルトに戻すを選びます。

7.3 変数ペインで右クリックする

変数ペインで右クリックすると、多くのタスクを実行できるコンテキストメニューを表示します。

- 選択した変数のみ維持 選択した変数のみデータセット内に残し、他の変数をドロップ（削除）します。
- 選択した変数を削除 選択した変数をデータセット内から削除します。
- 選択した変数のメモを管理... 1 つの変数にメモを付けたり削除するウィンドウを開きます。複数の変数を選択している場合、この機能は使用できません。
- データセットのメモを管理... データセット全体にメモを付けたり削除するウィンドウを開きます。
- 変数リストをコピー 選択した変数名をクリップボードにコピーします。
- すべて選択 表示しているすべての変数を選択します。フィルタの影響で変数が非表示である場合、この操作では選択されません。
- コマンドウィンドウに変数を送る 選択した変数名をコマンドウィンドウに入力します。これはソートやグループ化と組み合わせると、大きなデータセット内で変数リストを作成するのに便利です。
- 印刷... 変数ペインを印刷します。変数マネージャで列幅を変えると、印刷される列幅を変更できます。

7.4 変数プロパティペイン

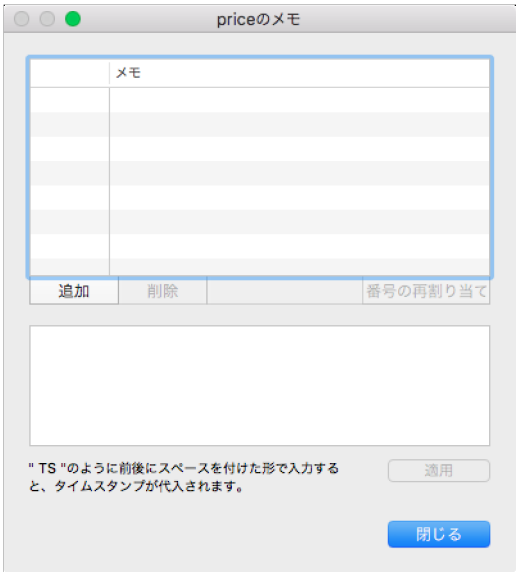
変数ペインで選択した変数（行）のプロパティを変更する時は変数プロパティペインを利用します。変数を 1 つ選択した場合、その変数のプロパティをすべて変更できます。複数の変数を選択している場合、それらの形式、タイプ、値ラベルを一度に変更できます。このペインは **[GSM] 6 Using the Data Editor**

(データエディタを使用する) 内の、「変数の名前およびフォーマットの変更」では説明したダイアログと同じように操作できます。また次のセクションで示すように Stata では変数およびデータセットにメモを追加できます。

7.5 メモを管理する

Stata では各変数またはデータセット全体にメモを追加できます。このメモはシンプルなテキストメモで、自由に記入できます。例えばデータセットの出所、変数に関連したデータ採取時の備考、変数の分析手法についてなど、どのようなものでも構いません。

まずは変数ペインで変数を選択することから始めましょう。例として変数 price を使用します。変数プロパティペイン内のメモ欄の右隣りにある、管理... ボタンをクリックし、以下のダイアログを開きます。



ではメモを追加しましょう。

1. 追加ボタンをクリックしてメモを追加します。
2. 「TS - started working」と入力します。「TS」とそれに続くスペースで時間スタンプをメモに挿入します。適用をクリックするとメモを作成します。
3. あと 2 つメモを入力します。下図のように価格についてのメモを追加しましょう。

ではメモの追加・削除・編集をやってみましょう。メモは長期プロジェクト中などに書き残しておくと便利です。メモマネージャ（メモを入力するダイアログ）でメモを変更すると、その度に結果ウィンドウにコマンドを出力します。

8 データをインポートする

8.1 コピーと貼り付け

データをインポートする最も簡単な方法は、表作成のできるアプリケーションからデータをコピーし、データエディタに直接貼り付けることです。これはすべてのスプレッドシート型のアプリケーションからのインポートで使用できます。また、多くのデータベース、複数のワープロやウェブページのアプリケーションにも対応しています。データ範囲すべてをコピーし、それをデータエディタに貼り付ければうまくいくはずです。さらにカンマ（,）で区切られているテキストファイルも、コピーしてデータエディタ内に貼り付けることができます。

例えば、友人が古い車に関する小さなデータセットを持っているとします。

```
VW Rabbit,4697,25,1930,3.78
Olds 98,8814,21,4060,2.41
Chev.Monza,3667,,2750,2.73
,4099,22,2930,3.58
Datsun 510,5079,24,2280,3.54
Buick Regal,5189,20,3280,2.93
Datsun 810,8129,,2750,3.55
```

このデータを Stata に取り込みましょう。この操作は簡単に行えます。

1. 現在のデータセットに「clear」と打ち込んでメモリ内からデータを消します。
2. 上記のデータをコピーします。
3. データエディタを編集モードで開きます。
4. 変数 > 特殊な貼り付けを選びます。
5. Stata は列の区切り文字がカンマ（comma）であることを認識し、データの見え方をプレビューします
6. **OK** ボタンをクリックします。

Stata はうまくデータをインポートしました。

同じデータセットを章の後半でコピーと貼り付け以外の方法でインポートするので、テキストファイルとして保存します。メインの Stata ウィンドウに戻り、**do** ファイルエディタボタン  を押して、新しい do ファイルエディタウィンドウを開きます。do ファイルエディタにデータを貼り付けて保存ボタンをクリックします。作業中のフォルダを参照し、このファイルを「a few cars.csv」として保存します。もし作業中のフォルダ（working directory）が分からない場合は Stata のメインウィンドウの下にある、ステータスバーの左端を見てください。

スプレッドシートからデータをコピーする場合、特殊なフォーマットで入力されていることもあります。そのままコピーすると、フォーマットが矩形にならずに崩れてしまう場合もあります。スプレッドシートに空の行や列、重複しているヘッダ、結合したセルがないことを確認してください。これらが残っていると貼り付ける時にトラブルを起こす可能性があります。スプレッドシートが表のように見えるなら基本的に問題はないはずです。

8.2 データをインポートするコマンド

コピーと貼り付けは Stata 内にデータをインポートする簡単な方法ですが、はっきりとした操作記録（履歴）が必要な場合、別の方法でデータをインポートしなければなりません。章の残りを使い、コピーと貼り付け以外のインポートコマンドを確認します。また、繰り返し行う作業に適したインポート方法と、様々な形式のデータをインポートする方法も紹介します。

Stata にはデータをインポートするコマンドが複数あります。一般的なテキストファイルのデータセットをインポートする時に使用するコマンドの内、特によく使用する 3 つのコマンドは次の通りです。

- import delimited スプレッドシートやデータベースプログラムで作成したテキストファイルを読み取ります。また、さらに全般的に、列を区切っているデリミタ、例えばカンマ、タブ、セミコロン、スペースが定義されているテキストファイルも読み取れます。
- infile スペース区切りのファイルとデータが固定列で定義されているファイルを読み取ります。
- infix 固定列になっているファイルを読み取ります。1 つのデータが途中で分断されても読み取れます。

Stata はこの他にもさまざまなタイプのファイルを読み込むコマンドや、外部のデータベースから直接データを取得するコマンドなどがあります。

- import excel Microsoft Excel のファイルを直接読み込みます。.xls と.xslx ファイルを読み込めます。
- import sas SAS ファイルを読み込むことができます。
- import spss IBM SPSS Statistics ファイルを読み込むことができます。
- import sasxport5 SAS V5 Transport ファイルを読み込みことができます。import sasxport8 は SAS V8 Transport ファイルを読み込みます。

- odbc　ODBC （Open Database Connectivity）ドライバに対応しているデータベースからインポートできます。
- jdbc　JDBC (Java Database Connectivity) ドライバに対応しているデータベースからインポート、SQL を実行できます

他の形式のインポートも可能です。インポート可能な形式の一覧は [D] **import** をご覧ください。

各コマンドはそれぞれのファイルが特定の形式であることを前提にしています。この章ではファイル形式についてサンプルを用いて説明します。詳細に関しては Data Management Reference Manual を import delimited コマンド参照してください。

8.3 import delimited コマンド

import delimited コマンドはスプレッドシートやデータベースのプログラムによって作成されたテキストファイルを読み取るコマンドです。この形式のファイルは、インターネット上でデータをやり取りする中で最も一般的なデータ形式です。全てのスプレッドシートプログラムと多くのデータベースアプリケーションではデータセットをテキストファイルとして保存できます。この時、行をタブまたはカンマで区切って保存します。プログラムによっては列タイトル（Stata では変数名）をテキストファイル内に保存できます。

このようなファイルを読み込むにはコマンドウィンドウに「import delimited ファイル名」を入力します。この場合ファイル名はインポートするテキストファイルの名前です。import delimited コマンドが自動で検出するデリミタ（区切り）はタブかカンマです。また、各列のデータの種類も自動で判別します。ファイル名にスペースが使用されている場合、ダブルクオテーション（″ ″）内にファイル名を入力します。目的のファイルが現在の作業フォルダ内にない時はそのファイルへのパスも忘れずに入力してください。

import delimited コマンドはデリミタがタブかカンマである場合、自動的に認識します。他の記号をデリミタとして使用しているファイルの場合、import delimited コマンドの delimiters() オプションを使用してください。

章の冒頭（「コピーと貼り付け」セクション）で「a few cars.csv」というファイルを保存しました。このデータには古い車の make, price, MPG, weight, gear ratio が入っています。変数名はファイルに含まれていないので import delimited コマンドは独自の名前を割り振ります。そして、カンマでフィールドが区切られています。現在開いているデータを一度 clear コマンドで消し、import delimited コマンドを使用してこのファイルのデータを読み込みましょう。このファイル名にはスペースを使用しているのでダブルクオテーション（″ ″) を使用します。

```
. clear
. import delimited "a few cars.csv"
(encoding automatically selected: ISO-8859-9)
(5 vars, 7 obs)
```

データエディタで見ると、章の冒頭で紹介したコピーと貼り付けのデータと全く同じものだと分かります。list コマンドを使用して結果ウィンドウにデータを表示します。separator(0) オプションはデフォルトで 5 行ごとに挿入する水平線を無効にします。

```
. list, separator(0)

     +-------------------------------------------+
     |          v1     v2   v3     v4     v5 |
     |-------------------------------------------|
  1. |   VW Rabbit   4697   25   1930   3.78 |
  2. |     Olds 98   8814   21   4060   2.41 |
  3. | Chev. Monza   3667    .   2750   2.73 |
  4. |               4099   22   2930   3.58 |
  5. |  Datsun 510   5079   24   2280   3.54 |
  6. | Buick Regal   5189   20   3280   2.93 |
  7. |  Datsun 810   8129    .   2750   3.55 |
     +-------------------------------------------+
```

ファイルを取り込む時に変数名を指定する場合、次のように import delimited コマンドを実行する時に使用する変数名も入力します。変数名を指定する場合、ファイル名の前に using キーワードを入力しなければなりません。

```
. import delimited make price mpg weight gear_ratio using "a few cars.csv"
(encoding automatically selected: ISO-8859-9)
(5 vars, 7 obs)
. list, separator(0)

     +-------------------------------------------------+
     |          make   price   mpg   weight   gear_r~o |
     |-------------------------------------------------|
  1. |   VW Rabbit    4697    25     1930       3.78 |
  2. |     Olds 98    8814    21     4060       2.41 |
  3. | Chev. Monza    3667     .     2750       2.73 |
  4. |                4099    22     2930       3.58 |
  5. |  Datsun 510    5079    24     2280       3.54 |
  6. | Buick Regal    5189    20     3280       2.93 |
  7. |  Datsun 810    8129     .     2750       3.55 |
     +-------------------------------------------------+
```

上記から分かるように、Stata は gear ratio を gear r~o として表示します。gear r~o は gear ratio を意味する、固有の省略形です。デフォルトで変数名の文字数が 8 文字よりも大きい場合、省略形を表示します。

gear_ratio を省略形で表示しないようにするには abbreviate(10)オプションで指定できます。

```
. list, separator(0) abbreviate(10)
```

	make	price	mpg	weight	gear_ratio
1.	VW Rabbit	4697	25	1930	3.78
2.	Olds 98	8814	21	4060	2.41
3.	Chev. Monza	3667	.	2750	2.73
4.		4099	22	2930	3.58
5.	Datsun 510	5079	24	2280	3.54
6.	Buick Regal	5189	20	3280	2.93
7.	Datsun 810	8129	.	2750	3.55

list コマンドと「~」を使用する省略形の詳細は、**[GSM] 10 Listing data and basic command syntax (データのリストと基本コマンドの構文)** をご覧ください。

このデータセットは次の章でも使用するので保存します。コマンドウィンドウに「save afewcars」と入力し、Return キーを押します。

import delimited コマンドを使用する他に、テキストファイルの内容をコピーしてデータエディタに特殊な貼り付け**...** で貼り付けるとインポート可能です。その時に表示するダイアログでカンマ（comma）をデリミタとして選択します。

デリミタでうまく区切られていないファイルや 1 つの観測が複数行にまたがるデータのために infile と infix コマンドがあります。このようなファイルのインポートの詳細については **[D] import** をご覧ください。

8.4 他のソフトウェアからファイルをインポートする

Stata には他アプリケーションで作成したデータを読み取る特殊なコマンドがあります。

import excel コマンドは Microsoft Excel で作成したファイルを読み取るものです。import excel コマンドの詳細は **[D] import excel** をご覧ください。

import spss コマンドは IBM SPSS Statistics で作成したファイルを読み取るものです。詳細は [D] import spss をご覧ください。

import sas コマンドは SAS で作成したファイルの読み取りと作成を行います。詳細は **[D] import sas** をご覧ください。

import sasxport5 と import sasxport8 の両コマンドは SAS V5 および V8 Transport ファイルを読み取ります。詳細は [D] Import sasxport5 と [D] Import sasxport8 をご覧ください。

ODBC に対応しているソフトウェアからインポートする場合、odbc コマンドで中間ファイルを作成することなく読み取ります。詳細は **[D] odbc** をご覧ください。また、ODBC のセットアップに関しては、FAQ にも有用な情報があります (https://www.stata.com/support/faqs/datamanagement/configuring-odbc)。

ｊｄｂｃコマンドは JDBC を使用して、データと接続、読み込み、およびデータベースへのデータの挿入、クエリの実行を行います。詳細は、 [D] **jdbc** をご覧ください。

インポートする方法の一覧は次の通りです。

- Microsoft Excel の.xls と.xlsx 拡張子のファイルの場合、import excel コマンドを使用します。
- IBM SPSS Statistics の.sav 拡張子のファイルの場合、import spss コマンドを使用します。
- Windows マシンで作成された SAS の.sas7bdat 拡張子のファイルの場合、imports sas コマンドを使用します。
- スプレッドシートまたはデータベースからインポートするファイルがタブ区切りか CSV ファイルの場合、import delimited コマンドを使用します。
  - フィックス形式ファイルをインポートする場合、infile コマンドを定義ファイルと共に使用するか、infix コマンドを使用します。
- ODBC でアクセス可能なデータベースの場合、odbc コマンドを使用します。
- JDBC でアクセス可能なデータベースの場合、jdbc コマンドを使用します。
- SAS V5 Transport ファイルをインポートする場合、import sasxport5 コマンドを使用します。
- SAS V8 Transport ファイルをインポートする場合、import sasxport8 コマンドを使用します。
- 米連邦準備銀行経済データの場合、import fred を使用します。
- Haver Analytics データベースのデータを取り込む場合、import haver コマンドを使用します。
- dBASE ファイルの場合、import dbase コマンドを使用します。
- 表形式のデータがあれば、データエディタにコピー& ペーストを行うことができます。
- 最後に、サードパーティのファイル変換プログラムを購入し、ファイル形式を Stata 形式に変換するという方法もあります。

9 データのラベリング

9.1 データを見やすくするには

この章ではデータセット、変数、値のラベリングについて説明します。ラベリングはデータそのままでは分かりにくい変数や値に説明を加えることができます。0 と 1 のダミー変数を使用してデータをカテゴリー分けする時にそのままデータエディタで表形式にしてみても、この 0 と 1 がそれぞれ何を示すのか分かりません。このような場合、値ラベルを使用すると、0 と 1 に具体的なカテゴリー名を表示できます。例えば、afewcars.dta のデータセットにダミー変数 0 と 1 を表示します。これでは何を示しているのか分かりません。これらの様式は他の人とデータを共有する際に重要な役割を担うとともに、自分自身のためにもなります。結果ウィンドウに結果を表示する時にもラベルが使われるため、データセットを正しくラベリングすると結果も分かりやすくなります。では、例題を使いながらデータセット、変数、値のラベリングの作業を行いましょう。

```
. use afewcars
. list, separator(0)

     +-------------------------------------------------+
     |        make   price   mpg   weight   gear_r~o |
     |-------------------------------------------------|
  1. |   VW Rabbit    4697    25     1930       3.78 |
  2. |     Olds 98    8814    21     4060       2.41 |
  3. | Chev. Monza    3667     .     2750       2.73 |
  4. |                4099    22     2930       3.58 |
  5. |  Datsun 510    5079    24     2280       3.54 |
  6. | Buick Regal    5189    20     3280       2.93 |
  7. |  Datsun 810    8129     .     2750       3.55 |
     +-------------------------------------------------+
```

9.2 データセットの構造：describe コマンド

[GSM] **8 Importing data (**データのインポート**)** 内の「 import delimited コマンド」セクションの最後で「afewcars.dta」というデータセットを保存しました。このデータセットを整え、他の研究者が見たときに理解できるようにします。まずはデータエディタでデータを確認しましょう。

この表からではデータセット内の値についてはほとんど分かりません。それだけではなく、price(価格) や weight (車重) の単位も分かりません。また、「mpg」の表記はアメリカ以外の人には意味が通じない可能性もあります。データセットの構造を確認するともう少し何か分かるかもしれません。データセットの構造を確認するには describe コマンドを使います（ **[GSM] 1 Introducing Stata—sample session (Stata** の紹介**—**サンプルセッション**)** で 1 度使用しました)。

```
. describe
Contains data from afewcars.dta
  Observations:       7
     Variables:       5                          21 Jan 2021 16:24

------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------
make            str11   %11s
price           int     %8.0g
mpg             byte    %8.0g
weight          int     %8.0g
gear_ratio      double  %10.0g
------------------------------------------------------------------------------
Sorted by:
```

研究者がデータ分析に使用できそうな情報はこの中にはほとんどありません。しかし表の初めの 3 列から Stata のデータの扱い方が分かります。

1. 変数名（*Variable name*）は各変数を表す固有の名前です。
2. 保存タイプ（*Storage type*）、またはデータ型は変数内にデータを保存する形式を示します。Stata には 6 種類の保存タイプがあり、それぞれ必要メモリ量が異なります。
   (a) 整数（integers）:

   保存タイプ byte は最小 −127、最大 100 までの整数データを格納でき、1 観測につき 1 バイトのメモリを使用します。

   保存タイプ int は最小 −32,767、最大 32,740 までの整数データを格納でき、1 観測につき 2 バイトのメモリを使用します。

   整数に関する long は最小 −2,147,483,647、最大 2,147,483,620 までの整数データを格納でき、1 観測につき 4 バイトのメモリを使用します。

   (b) 実数（real numbers）:

   保存タイプ float は浮動小数で、8.5 桁の精度でデータを格納できます。1 観測につき 4 バイトのメモリを使用します。
   保存タイプ double は浮動小数で、16.5 桁の精度でデータを格納できます。1 観測につき 8 バイトのメモリを使用します。

   (c) 文字列（テキスト）: 1 から 2,045 文字までの文字列データを格納します。メモリは ASCII 文字の場合 1 文字につき 1 バイト、Unicode 文字の場合 1 文字につき最大 4 バイトを使用します。

   str1 は 1 バイト長の文字列を記入できます。
   str2 は 2 バイト長の文字列を記入できます。
   str3 は 3 バイト長の文字列を記入できます。

…

str2045 は 2,045 文字の長さの記号を記入できます。

(d) また、Stata は strL（スタール）を使うと、任意の長さの文字列を格納できます。利用できる文字列の長さは 2,000,000,000 文字までです。また、strL はバイナリデータ、よくデータベースで BLOB（binary large objects）と呼ばれる文字列も格納できます。これらの詳細はこのページでは紹介しません。

保存タイプは計算の精度とデータセットの容量に影響します。保存タイプに関する詳細は「help data types」のコマンドを実行するか、 [D] **Data types** をご覧ください。

3. 表示形式（*Display format*）は数値の表示形式をコントロールします。詳しくは [U] **12.5 Formats:Controlling how data are displayed** をご覧ください。デフォルトでは、保存タイプを元に表示形式を選択します。

このデータセットは必要な情報をすべて含めたものにしましょう。

丁寧にラベリングしてあるデータセットを参考にするために、Stata Press のアーカイブにあるデータセットを開いてみましょう。これはデータをロードして現在の作業内容を中断する必要も、データセットのコピーをコンピュータ上に準備する必要もありません。（インターネット上で行えることについての詳細は、 [GSM] **19 Updating and extending Stata—Internet functionality (Stata** のアップデートと拡張**—**インターネットでの機能**)** 実際に行うことは describe コマンドで正しいファイルを参照するだけです。「describe using ファイル名」とコマンドウィンドウに入力して実行してください。

```
. describe using https://www.stata-press.com/data/r18/auto
Contains data                                  1978 automobile data
 Observations:            74                   13 Apr 2022 17:45
    Variables:            12
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
make            str18   %-18s                 Make and model
price           int     %8.0gc                Price
mpg             int     %8.0g                 Mileage (mpg)
rep78           int     %8.0g                 Repair record 1978
headroom        float   %6.1f                 Headroom (in.)
trunk           int     %8.0g                 Trunk space (cu. ft.)
weight          int     %8.0gc                Weight (lbs.)
length          int     %8.0g                 Length (in.)
turn            int     %8.0g                 Turn circle (ft.)
displacement    int     %8.0g                 Displacement (cu. in.)
gear_ratio      float   %6.2f                 Gear ratio
foreign         byte    %8.0g       origin    Car origin
-------------------------------------------------------------------------------
Sorted by: foreign
```

この出力結果の方がより有益な情報を含んでいます。この中でもデータセットを理解する時に必要な情報がラベルに記載してある箇所が 3 か所あります。

1. 1 行目の「1978 automobile Data」はデータラベルです。データセットの情報が分かります。データセットをラベリングするには、メインメニューからデータ > データユーティリティ > ラベルユーティリティ > データセットのラベルと操作するか、「label data」コマンドを使用するか、もしくはプロパティウィンドウのラベル欄で編集を行います。プロパティウィンドウで編集する場合、プロパティウィンドウのロックが解除してあることを確認してください。
2. 各変数には変数ラベルがついています。変数ラベルは、一般的な会話で変数を呼称する際に使用する名称です。このラベルには、変数の単位情報も入っています。変数をラベリングするには、変数ウィンドウで変数を選択し、プロパティウィンドウのラベル欄で編集を行います。変数のラベリングは変数マネージャや「label variable」コマンドからも実行できます。
3. 変数 foreign には値ラベルが付いています。値ラベルは数値変数、例えば foreign の数値を言葉で置き換えます。describe コマンドの出力から、数値変数 foreign は値ラベル origin と対応することが確認できます。describe コマンドからは分かりませんが変数 foreign は数値 0 か 1 を持っています。値ラベル origin は、0 は“Domestic”、1 は“Foreign”と表示します。データを browse する（見る）と（方法については **[GSM] 6 Using the Data Editor (データエディタを使用する)** をご覧ください)、変数 foreign は値“Domestic” と “Foreign”を表示します。変数内の値をラベル付けするには 2 つの段階を踏む必要があります。まずは値ラベルの定義を行います。この操作はデータエディタまたは変数マネージャのダイアログで行うか、データ > データユーティリティ > ラベルユーティリティ > 値ラベルの管理と選択します。または「label define」と入力しても同じ操作を行えます。ラベルを定義した後に対応する変数と関連付けます。関連付けるにはデータ > データユーティリティ > ラベルユーティリティ > 変数に値ラベルを割り当てと操作するか、「label values」コマンドを使ってください。

メモ：値ラベルの名前は変数名と同じでも差支えありません。この場合、値ラベルの名前が foreign でも問題ありません。

9.3 データセットと変数のラベリング

データセット afewcars.dta をロードしてラベルを作成します。この操作は簡単なのでコマンドウィンドウから行いましょう。 **[GSM] 6 Using the Data Editor (データエディタを使用する)** の変数の名前および形式の変更では、同じ目的でデータエディタを使用しました。コマンドウィンドウを使用しても、データエディタを使用しても、結果ウィンドウに表示する結果は同じコマンドになります。
今回、直接コマンドウィンドウから操作する方法をご紹介します。次のように入力しましょう。

```
. use afewcars
. describe
Contains data from afewcars.dta
 Observations:             7
    Variables:             5                  29 Mar 2023 09:11
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
make            str18   %18s
price           float   %9.0g
mpg             float   %9.0g
weight          float   %9.0g
gear_ratio      float   %9.0g
-------------------------------------------------------------------------------
Sorted by:
. label data "A few 1978 cars"
. label variable make "Make and model"
. label variable price "Price (USD)"
. label variable mpg "Mileage (miles per gallon)"
. label variable weight "Vehicle weight (lbs.)"
. label variable gear_ratio "Gear ratio"
. describe
Contains data from afewcars.dta
 Observations:             7                  A few 1978 cars
    Variables:             5                  29 Mar 2023 09:11
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
make            str18   %18s                  Make and model
price           float   %9.0g                 Price (USD)
mpg             float   %9.0g                 Mileage (miles per gallon)
weight          float   %9.0g                 Vehicle weight (lbs.)
gear_ratio      float   %9.0g                 Gear ratio
-------------------------------------------------------------------------------
Sorted by:
     Note: Dataset has changed since last saved.
. save afewcars2
file afewcars2.dta saved
```

9.4 変数の値をラベリングする

これから新しいダミー変数をデータセットに加えます。ダミー変数の数値 0 はアメリカ国内メーカーの車を、1 は他国のメーカーの車をそれぞれ表します。データエディタを開き、今まで学んできた内容をもとに変数 foreign を追加しましょう。

```
. list, separator(0)

     +-------------------------------------------------------------+
     |       make   price   mpg   weight   gear_r~o   foreign |
     |-------------------------------------------------------------|
  1. |   VW Rabbit    4697    25     1930       3.78         1 |
  2. |     Olds 98    8814    21     4060       2.41         0 |
  3. | Chev. Monza    3667     .     2750       2.73         0 |
  4. |                4099    22     2930       3.58         0 |
  5. |  Datsun 510    5079    24     2280       3.54         1 |
  6. | Buick Regal    5189    20     3280       2.93         0 |
  7. |  Datsun 810    8129     .     2750       3.55         1 |
     +-------------------------------------------------------------+
```

もちろん、データエディタを使用して新しい変数を作成することもできます。（データエディタの使用法については [GSM] **6 Using the Data Editor (**データエディタを使用する**)** をご覧ください）。 “0”と“1”がそれぞれ定義しているカテゴリーはこの内容の中では分かりますが、値ラベルを使用してより明確にしましょう。車について何も知らない人が見ても何を表しているデータなのか分かりやすくするためです。よって、値ラベルを使って誰でもその意味が分かるようにします。データエディタ内でポイント&クリックインターフェイスを使って値ラベルを作成し添付する作業は、 [GSM] **6 Using the Data Editor (**データエディタを使用する**)** の「データを変更する」セクションで行いました。ここではコマンドウィンドウから直接値ラベルを作成しましょう。

```
. label define origin 0 "Domestic" 1 "Foreign"
. label values foreign origin
. describe
Contains data from afewcars2.dta
 Observations:             7                  A few 1978 cars
    Variables:             6                  29 Mar 2023 09:11
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
make            str18   %18s                  Make and model
price           float   %9.0g                 Price (USD)
mpg             float   %9.0g                 Mileage (miles per gallon)
weight          float   %9.0g                 Vehicle weight (lbs.)
gear_ratio      float   %9.0g                 Gear ratio
foreign         byte    %8.0g      origin
-------------------------------------------------------------------------------
Sorted by:
     Note: Dataset has changed since last saved.
. save afewcarslab
file afewcarslab.dta saved
```

値ラベルを定義するには「label define ラベル名値 1″カテゴリー名 1″ 値 2 ″カテゴリー名 2″ …」とコマンドウィンドウに入力します。変数に作成したラベルを付けるには「label values 変数名ラベル名」と入力します。

では、先ほど作成した変数とラベルの情報を一度保存しておきましょう。データセットに値ラベルを付けて整えたことと、次の章でもこのデータを使用することを踏まえて、今までのデータセットと混同しないように新しいファイル名で保存します。

値ラベルをポイント&クリックインターフェイスで定義する場合はメインウィンドウおよびデータエディタのプロパティウィンドウか変数マネージャを使用します。詳しくは **[GSM] 7 Using the Variables Manager (変数マネージャを使用する)** をご覧ください。

この他にも値ラベルには使用例や構文があります。具体例は **[U] 12.6.3 Value labels** をご覧ください。

変数およびデータにメモを付け足すこともできます。メモについては **[GSM] 6 Using the Data Editor (データエディタを使用する)** 内の「変数の名前および形式の変更」または **[GSM] 7 Using the Variables Manager (変数マネージャを使用する)** 内の「メモを管理する」セクションをご覧ください。さらに詳しく知りたい場合は「help notes」とコマンドを打ってシステムヘルプを参照したり、PDF マニュアル内の **[D] notes** の解説を参照してください。

10 データのリストと基本コマンドの構文

10.1 コマンド構文

この章では Stata のコマンド構文の基本を学ぶと共に、データリストの表示をコントロールする方法について学びます。

このマニュアルからも分かるように、Stata はメニューとダイアログで操作する方法と、コマンドウィンドウにコマンドを直接入力して操作する方法の 2 つを用意しています。最初は他のソフトウェアのようにメニューで操作していたとしても、コマンドウィンドウに慣れるとより速く、快適に操作できるようになります。コマンド文がシンプルで一貫性のある構文になっているので、コマンドウィンドウを使う方が効率的に操作できます。ここでコマンドウィンドウの使い方に慣れ、より快適に Stata を使えるようになりましょう。「help language」コマンドではオンラインヘルプで追加情報と例題を、PDF マニュアルの [U] **11 Language syntax** ではコマンド構文に関する詳細の確認をできます。

「help list」とコマンドウィンドウに打ち込んで、list コマンドの構文を確認しましょう。

list [*varlist*] [*if*] [*in*] [, *options*]

この構文の読み方は次の通りです。

- [ ] 内の引数はすべて任意です。list コマンドの場合、
  1. (a) *varlist* は任意です。*varlist* とは変数名のリストです。
     (b) *if* は任意です。*if* 条件で定めた内容に当てはまるデータにのみコマンドを実行します。 *if* 条件の使用例は [GSM] **6 Using the Data Editor (**データエディタを使用する**)** で一度紹介しています。
     (c) *in* は任意です。*in* 条件で指定した行番号にのみコマンドを実行します。
     (d) , と *options*（オプション）は任意です。コマンド文の他の部分とオプションはカンマで分けます。
- 特に指定がない限り、引数は 1 つのコマンド文で同時に使用できます。list コマンドに関しては *varlist* を *if* や *in* 条件と共に使用できます。
- 単語の一部に下線が引いてある場合、下線部は Stata が認識する最短の省略形です。下線部よりも長い省略形なら、Stata は正しいコマンドとして認識します。
  1. (a) list コマンドでは“l”に下線がついているので、Stata は l, li, lis を list コマンドの省略形として認識します。
- [ ] の外の項目はすべて必須です。list コマンドの場合、list のみが必須項目となります。

これらのルールを念頭に置いて、それぞれの引数でどのように list コマンドの出力が変化するのか見ていきましょう。前の章の最後に使用したデータセット、afewcarslab.dta を使用します。

10.2 変数リストを使用してデータの一部をリストする

データの一部を選んでリストするには変数リスト（*varlist*）を使います。このセクションで紹介する変数リストの入力方法は、煩わしい入力の手間を省きながら長い変数名でも快適にご利用いただけます。

- list コマンドでは *varlist* の入力は任意です。つまり、変数を特に指定しない場合は全ての変数を選択するのと同じです。もう 1 つの考え方は、list コマンドはデフォルトで全ての変数にコマンドを実行しますが、*varlist* を使用するとその変数にのみコマンドを実行します。
- 変数のサブセットだけをリストするには「list make mpg price」のように入力します。
- Stata はキーボード入力を少なくできる省略表記を複数用意しています。

  m*は m から始まるすべての変数を表します。
  price-weight はデータセットの順番で、変数 price から変数 weight までの全ての変数を表示します。
  ma?e は ma から始まり、最後の文字が e になる変数を表します。

- 「list gear r~o」のように、Stata が作成した変数固有の省略形でもコマンドを実行できます。
  省略形が固有の形式ではない場合、Stata はエラーメッセージを表示します。

```
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
2.	Olds 98	8814	21	4060	2.41	Domestic
3.	Chev. Monza	3667	.	2750	2.73	Domestic
4.		4099	22	2930	3.58	Domestic
5.	Datsun 510	5079	24	2280	3.54	Foreign
6.	Buick Regal	5189	20	3280	2.93	Domestic
7.	Datsun 810	8129	.	2750	3.55	Foreign

```
. l make mpg price
```

	make	mpg	price
1.	VW Rabbit	25	4697
2.	Olds 98	21	8814
3.	Chev. Monza	.	3667
4.		22	4099
5.	Datsun 510	24	5079
6.	Buick Regal	20	5189
7.	Datsun 810	.	8129

```
. list m*
```

	make	mpg
1.	VW Rabbit	25
2.	Olds 98	21
3.	Chev. Monza	.
4.		22
5.	Datsun 510	24
6.	Buick Regal	20
7.	Datsun 810	.

```
. li price-weight
```

	price	mpg	weight
1.	4697	25	1930
2.	8814	21	4060
3.	3667	.	2750
4.	4099	22	2930
5.	5079	24	2280
6.	5189	20	3280
7.	8129	.	2750

```
. list ma?e

     +-----------+
     |      make |
     |-----------|
  1. |  VW Rabbit |
  2. |    Olds 98 |
  3. | Chev. Monza |
  4. |            |
  5. |  Datsun 510 |
     |-----------|
  6. | Buick Regal |
  7. |  Datsun 810 |
     +-----------+

. l gear_r~o

     +-----------+
     | gear_r~o  |
     |-----------|
  1. |      3.78 |
  2. |      2.41 |
  3. |      2.73 |
  4. |      3.58 |
  5. |      3.54 |
     |-----------|
  6. |      2.93 |
  7. |      3.55 |
     +-----------+
```

10.3 if 条件でリストする

if 条件は論理表現からどのデータにコマンドを実行するのか決めます。if 条件が当てはまるデータにはコマンドを実行しますが、当てはまらないものには実行しません。真偽の判断を行う演算子は次の通りです。

<	よりも小さい
<=	以下
==	等しい
>	よりも大きい
>=	以上
!=	等しくない
&	and
|	or
!	not（論理的否定；~も使用できる）
()	カッコは評価の順番を指定するグループ分けに使用する

論理表現では、|（or）の前に&（and）を判断します（これは算数で足し算の前に掛け算を行うのと同じようなものです)。このルールを使用して if 表現を作成できますが、カッコを使用する方が確実になります。詳しくは [U] **13.2 Operators** をご覧ください。

```
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
2.	Olds 98	8814	21	4060	2.41	Domestic
3.	Chev. Monza	3667	.	2750	2.73	Domestic
4.		4099	22	2930	3.58	Domestic
5.	Datsun 510	5079	24	2280	3.54	Foreign
6.	Buick Regal	5189	20	3280	2.93	Domestic
7.	Datsun 810	8129	.	2750	3.55	Foreign

```
. list if mpg > 22
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
3.	Chev. Monza	3667	.	2750	2.73	Domestic
5.	Datsun 510	5079	24	2280	3.54	Foreign
7.	Datsun 810	8129	.	2750	3.55	Foreign

```
. list if (mpg > 22) & !missing(mpg)
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
5.	Datsun 510	5079	24	2280	3.54	Foreign

```
. list make mpg price gear if (mpg > 22) | (price > 8000 & gear < 3.5)
```

	make	mpg	price	gear_r~o
1.	VW Rabbit	25	4697	3.78
2.	Olds 98	21	8814	2.41
3.	Chev. Monza	.	3667	2.73
5.	Datsun 510	24	5079	3.54
7.	Datsun 810	.	8129	3.55

```
. list make mpg if mpg <= 22 in 2/4
```

	make	mpg
2.	Olds 98	21
4.		22

上記リストでは Stata が欠損値をデータ内の最大値より大きい値として扱うことを確認できます。よって、欠損値のある変数に if 表現を使用する場合は、注意が必要です。詳しくは [GSM] **6 Using the Data Editor (**データエディタを使用する**)**をご覧ください。

10.4 If コマンドでリストをする：よくある間違い

このセクションには list コマンドを使う時に良くある間違いとその修正例をまとめました。まず誤ったコマンド例を示し、その後に正しいコマンド文を載せます。正しいコマンド文を見る前に間違いが分かるか試してください。

```
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
2.	Olds 98	8814	21	4060	2.41	Domestic
3.	Chev. Monza	3667	.	2750	2.73	Domestic
4.		4099	22	2930	3.58	Domestic
5.	Datsun 510	5079	24	2280	3.54	Foreign
6.	Buick Regal	5189	20	3280	2.93	Domestic
7.	Datsun 810	8129	.	2750	3.55	Foreign

```
. list if mpg=21
=exp not allowed
r(101);
```

エラーはイコールが「==」で表現され、「=」ではありません。正しく入力すると次のようになります。

```
. list if mpg==21
```

	make	price	mpg	weight	gear_r~o	foreign
2.	Olds 98	8814	21	4060	2.41	Domestic

次は複数の if 表現を組み合わせる時によくある間違いです。

```
. list if mpg==21 if weight > 4000
invalid syntax
r(198);
. list if mpg==21 and weight > 4000
invalid 'and'
r(198);
```

2 つの if 表現を同じコマンド内で使う時は記号& を使用します。and や複数の if を使用しても Stata は認識しません。正しい if 表現は「if mpg==21 if weight>4000」ではなく「if mpg==21& weight>4000」になります。結果を出力すると次のようになります。

```
. list if mpg==21 & weight > 4000

     +-------------------------------------------------------------+
     |    make   price   mpg   weight   gear_r~o   foreign        |
     |-------------------------------------------------------------|
  2. | Olds 98    8814    21     4060       2.41   Domestic       |
     +-------------------------------------------------------------+
```

次の例は文字列変数の扱い方についてです。

```
. list if make==Datsun 510
Datsun not found
r(111);
```

文字列は make=="Datsun 510"のようにダブルクオテーション（" "）の中に入力してください。ダブルクオテーションを使用しないと、Stata は Datsun のみを存在しない変数だと認識します。よって、「Datsun not found」というエラーメッセージを表示します。修正すると次のようになります。

```
. list if make=="Datsun 510"

     +-------------------------------------------------------------+
     |    make   price   mpg   weight   gear_r~o   foreign        |
     |-------------------------------------------------------------|
  5. | Datsun 510  5079    24     2280       3.54   Foreign        |
     +-------------------------------------------------------------+
```

最後に文字列と値ラベルの混同についてです。

```
. list if foreign=="Domestic"
type mismatch
r(109);
```

値ラベルを使う変数は文字列変数のように見えますが、実際はダミー変数なので値として保存してあるのは数値です。変数 foreign には値 0 と 1 があり、0 に"Domestic"、1 に"Foreign"というラベルがついています。（詳しくは [GSM] **9 Labeling data (データのラベリング)** をご覧ください）をご覧ください。）値ラベルの元となっている数字を見るには「label list」コマンドを使用します。（詳しくは [D] **label** をご覧ください。）または「codebook 変数名」でその変数を確認します。エラーは if 表現を foreign==0 のように変更すれば直ります。

同じ内容を出力するのに別の表記方法を使うと、直接、値ラベル名を使用できます。

```
. list if foreign==0

     ┌──────────────────────────────────────────────────────────────┐
     │        make   price   mpg   weight   gear_r~o    foreign   │
     ├──────────────────────────────────────────────────────────────┤
  2. │     Olds 98    8814    21     4060       2.41   Domestic   │
  3. │ Chev. Monza    3667     .     2750       2.73   Domestic   │
  4. │                4099    22     2930       3.58   Domestic   │
  6. │ Buick Regal    5189    20     3280       2.93   Domestic   │
     └──────────────────────────────────────────────────────────────┘

. list if foreign=="domestic":origin

     ┌──────────────────────────────────────────────────────────────┐
     │        make   price   mpg   weight   gear_r~o    foreign   │
     ├──────────────────────────────────────────────────────────────┤
  2. │     Olds 98    8814    21     4060       2.41   Domestic   │
  3. │ Chev. Monza    3667     .     2750       2.73   Domestic   │
  4. │                4099    22     2930       3.58   Domestic   │
  6. │ Buick Regal    5189    20     3280       2.93   Domestic   │
     └──────────────────────────────────────────────────────────────┘
```

10.5 in でリストする

in 条件は *numlist*（数字リスト）を使い、リストするデータ範囲を指定します。正の数はデータセットの最初から数え始め、負の数はデータセットの最後から数え始めます。次の例題を参考にしてください。

```
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
2.	Olds 98	8814	21	4060	2.41	Domestic
3.	Chev. Monza	3667	.	2750	2.73	Domestic
4.		4099	22	2930	3.58	Domestic
5.	Datsun 510	5079	24	2280	3.54	Foreign
6.	Buick Regal	5189	20	3280	2.93	Domestic
7.	Datsun 810	8129	.	2750	3.55	Foreign

```
. list in 1
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign

```
. list in -1
```

	make	price	mpg	weight	gear_r~o	foreign
7.	Datsun 810	8129	.	2750	3.55	Foreign

```
. list in 2/4
```

	make	price	mpg	weight	gear_r~o	foreign
2.	Olds 98	8814	21	4060	2.41	Domestic
3.	Chev. Monza	3667	.	2750	2.73	Domestic
4.		4099	22	2930	3.58	Domestic

```
. list in -3/-2
```

	make	price	mpg	weight	gear_r~o	foreign
5.	Datsun 510	5079	24	2280	3.54	Foreign
6.	Buick Regal	5189	20	3280	2.93	Domestic

10.6 リストの出力をコントロールする

list コマンドの出力はオプションを使うことでコントロールできます。使用できるオプションとして、sepby( )オプションは変数のカテゴリー分類に応じてデータを分けて表示します。abbreviate( ) オプションは出力時に変数名省略を開始する最小文字数を指定できます。divider オプションは変数名の間に垂線を引きます。

```
. sort foreign make
. list ma p g f, sepby(foreign)

     +-------------------------------------------+
     |        make   price   gear_r~o   foreign  |
     |-------------------------------------------|
  1. |              4099       3.58   Domestic  |
  2. |  Buick Regal   5189       2.93   Domestic  |
  3. |  Chev. Monza   3667       2.73   Domestic  |
  4. |      Olds 98   8814       2.41   Domestic  |
     |-------------------------------------------|
  5. |   Datsun 510   5079       3.54    Foreign  |
  6. |   Datsun 810   8129       3.55    Foreign  |
  7. |    VW Rabbit   4697       3.78    Foreign  |
     +-------------------------------------------+

. list make weight gear, abbreviate(10)

     +-------------------------------+
     |        make   weight   gear_~o |
     |-------------------------------|
  1. |              2930       3.58 |
  2. |  Buick Regal   3280       2.93 |
  3. |  Chev. Monza   2750       2.73 |
  4. |      Olds 98   4060       2.41 |
  5. |   Datsun 510   2280       3.54 |
     |-------------------------------|
  6. |   Datsun 810   2750       3.55 |
  7. |    VW Rabbit   1930       3.78 |
     +-------------------------------+

. list, divider

     +-------------------------------------------------------------+
     |        make   price   mpg   weight   gear_r~o   foreign    |
     |-------------------------------------------------------------|
  1. |              4099    22     2930       3.58   Domestic    |
  2. |  Buick Regal   5189    20     3280       2.93   Domestic    |
  3. |  Chev. Monza   3667     .     2750       2.73   Domestic    |
  4. |      Olds 98   8814    21     4060       2.41   Domestic    |
  5. |   Datsun 510   5079    24     2280       3.54    Foreign    |
     |-------------------------------------------------------------|
  6. |   Datsun 810   8129     .     2750       3.55    Foreign    |
  7. |    VW Rabbit   4697    25     1930       3.78    Foreign    |
     +-------------------------------------------------------------+
```

separator()　オプションコマンドは水平線を指定された区切りで挿入します。特に指定がない場合、separator()オプションのデフォルト値は 5 です。

```
. list, separator(3)

     +-------------------------------------------------------------+
     |       make   price   mpg   weight   gear_r~o   foreign     |
     |-------------------------------------------------------------|
  1. |              4099    22     2930       3.58   Domestic    |
  2. | Buick Regal  5189    20     3280       2.93   Domestic    |
  3. | Chev. Monza  3667     .     2750       2.73   Domestic    |
     |-------------------------------------------------------------|
  4. |    Olds 98   8814    21     4060       2.41   Domestic    |
  5. | Datsun 510   5079    24     2280       3.54   Foreign     |
  6. |  VW Rabbit   4697    25     1930       3.78   Foreign     |
     |-------------------------------------------------------------|
  7. | Datsun 810   8129     .     2750       3.55   Foreign     |
     +-------------------------------------------------------------+
```

10.7 中断

Stata のコマンドを中断する時はメインウィンドウのツールバーにある中断ボタン  をクリックします。

中断ボタンをクリックした後の Stata の状態は、現在のコマンドを実行する前の状態に戻りますので安全です。

11 新しい変数を作成する

11.1 作成と置換

この章では Stata で変数の作成や値の変更を行う操作について説明します。データエディタを使う方法はすでに [GSM] **6 Using the Data Editor (**データエディタを使用する**)** で紹介しました。ここではコマンドウィンドウを使用する方法を見ていきます。この作業で使う 2 つのコマンドは次の通りです。

- generate 新しい変数を作成します。このコマンドの最小省略形は g です。
- replace 既存の変数の値を変更する時に使用します。このコマンドには省略形は存在しません。

  既存のデータを変更するコマンドなので、誤ってデータを変えてしまう事を未然に防ぐためです。

新しい変数を作成する最も基本的なコマンド文の形式は「generate *newvar* = *exp*」となります。この *exp* は表現（expression）を指します。もちろん、generate と replace のコマンドは、 if と in 条件と共に使用できます。Stata で表現というと、定数、既存の変数、演算子、関数を使って表す式のことです。データセット auto.dta 内にある変数を使って例を記すと、2 + price,weight^2 , sqrt(gear ratio)となります。

Stata が定義する演算子は次の表の通りです。

		大小関係
算術	論理	（数字および文字列）
+ 足し算	! not	> よりも大きい
- 引き算	| or	< よりも小さい
* 掛け算	& and	>= > または等しい
/ 割り算		<= < または等しい
^ べき乗		== 等しい
		!= 等しくない
+ 文字列の連結		

Stata は多くの統計、文字列、日付、時系列、プログラミングに関する関数を用意しています。オンラインヘルプを参照するには「help functions」コマンドを実行してください。また、PDF マニュアルを直接参照するには [D] **functions** をご覧ください。

メニューとダイアログから新しい変数を作成して既存の変数を変更することもできます。操作はメニューからデータ > データの作成または変更と行います。このダイアログから使用できる関数の確認もできます。しかし、今回はコマンドウィンドウの簡単な使い方とよくある間違いをお伝えするため、コマンド

ウィンドウを使って操作します。Stata にはいくつかのユーティリティコマンドがあり、それを使用して新しい変数を作成できます。

- egen コマンドは変数グループを横断して作業する時やデータのグループをまとめて操作する時に便利です。詳細は [D] **egen** をご覧ください。
- encode コマンドはカテゴリーごとの文字列をエンコードした数値に置き換えます。反対に、decode コマンドは数値のカテゴリーを文字列に直します。詳細は [D] **encode** をご覧ください。
- destring コマンドは数値であるべき文字列変数、例えば通貨記号のついた数字等を数値に変換します。数値から文字列に変換するには tostring コマンドが便利です。詳細は [D] **destring** をご覧ください。

この章では、generate と replace コマンドに集中します。

11.2 generate コマンド

generate コマンドはよく使うので、次に述べることはぜひ覚えておいてください。

- generate コマンドの基本形は「generate *newvar=exp*」です。*newvar* はこのコマンドで新たに作成する変数の名前で、*exp* はその変数を定義する表現です。既存の変数と同名の変数を generate コマンドで作成しようとすると、エラーメッセージが表示され、変数作成に失敗します。
- 欠損値を使って代数計算をすると、欠損値を算出します。これは、例えば 0 を使う割り算や負の数の平方根のように、計算が不可能な数式を計算した場合と同じです。
- 新しい変数（*newvar*）を作成中に欠損値ができた場合、Stata は変数内の欠損値の数を常に報告します。Stata が欠損値の数を表示しない場合、欠損値はできなかった事を示します。
- generate コマンドを使って新しい変数（*newvar*）を作成しながら、保存タイプの設定ができます。例えば、ダミー変数（0/1）は保存タイプ byte として作成するといいでしょう。これはデフォルトの保存タイプである float よりもデータ 1 つにつき 3 バイトの節約になるからです。

次の例題はデータセット afewcarslab からいくつかの新しい変数を作成できます。このデータセット afewcarslab は [GSM] **9 Labeling data (データのラベリング) をご覧ください)** の変数の値をラベリングするで作成したデータセットです。（実際に読み進めながら操作するにはデータセット

automobile を開きます。このデータセットを小さくして、リストを短くしています。）最後の例はダミー変数を作成する方法を示しています。この場合、3000 ポンド（約 1350kg）より重い車を識別します。Stata の論理表現では、1 は“真”、0 は“偽”となります。この例での if 条件は、作成するダミー変数の分類対象である weight（車重）が欠損値でないことを確認するために使用しています。

```
. use afewcarslab
(A few 1978 cars)
. list make mpg weight

     +-------------------------+
     |        make   mpg   weight |
     |-------------------------|
  1. |   VW Rabbit    25     1930 |
  2. |     Olds 98    21     4060 |
  3. | Chev. Monza     .     2750 |
  4. |                 22     2930 |
  5. |  Datsun 510    24     2280 |
     |-------------------------|
  6. | Buick Regal    20     3280 |
  7. |  Datsun 810     .     2750 |
     +-------------------------+

. * changing MPG to liters per 100km
. generate lphk = 3.7854 * (100 / 1.6093) / mpg
(2 missing values generated)
. label var lphk "Liters per 100km"
. * getting logarithms of price
. g lnprice = ln(price)
. * making an indicator of hugeness
. gen byte huge = weight >= 3000 if !missing(weight)
. l make mpg weight lphk lnprice huge

     +-------------------------------------------------+
     |        make   mpg   weight       lphk    lnprice   huge |
     |-------------------------------------------------|
  1. |   VW Rabbit    25     1930   9.408812   8.454679      0 |
  2. |     Olds 98    21     4060   11.20097   9.084097      1 |
  3. | Chev. Monza     .     2750          .   8.207129      0 |
  4. |                 22     2930   10.69183   8.318499      0 |
  5. |  Datsun 510    24     2280   9.800845   8.532869      0 |
     |-------------------------------------------------|
  6. | Buick Regal    20     3280   11.76101   8.554296      1 |
  7. |  Datsun 810     .     2750          .   9.003193      0 |
     +-------------------------------------------------+
```

11.3　replace コマンド

新しい変数を作成する時は generate コマンドを使い、既存の変数の値を変更する時は replace コマンドを使用します。ある変数 A を作成後に再び変数 A を作成しようとするとエラーになり、値は上書きされません。変数の値を間違えた時は replace コマンドを使って修正します。このように Stata は誤ってデータを上書きすることを防ぎますが、replace コマンドも省略できると他の r から始まるコマンドと

間違える可能性があります。それを防ぐために replace コマンドは省略できません。Stata ではデータ変更ができるコマンドを使う場合は全てのスペルを入力する必要があります。

```
. list make weight

     +-------------------+
     |        make   weight |
     |-------------------|
  1. |   VW Rabbit     1930 |
  2. |     Olds 98     4060 |
  3. | Chev. Monza     2750 |
  4. |                 2930 |
  5. |  Datsun 510     2280 |
     |-------------------|
  6. | Buick Regal     3280 |
  7. |  Datsun 810     2750 |
     +-------------------+

. * will give an error because weight already exists
. gen weight = weight/1000
variable weight already defined
r(110);

. * will replace weight in lbs by weight in 1000s of lbs
. replace weight = weight/1000
(7 real changes made)

. list make weight

     +-------------------+
     |        make   weight |
     |-------------------|
  1. |   VW Rabbit     1.93 |
  2. |     Olds 98     4.06 |
  3. | Chev. Monza     2.75 |
  4. |                 2.93 |
  5. |  Datsun 510     2.28 |
     |-------------------|
  6. | Buick Regal     3.28 |
  7. |  Datsun 810     2.75 |
     +-------------------+
```

例えば、新しい変数 predprice を作成します。この変数は翌年の車の価格を予測するものです。国 内製の車の価格は 5%、外国製の車は 10% の割合で値上がりすると推定した、とします。

変数を作成するに当たり、まずは先ほどの generate コマンドを使用して国内製の車の価格を予測します。この時点では外国製の車の予測価格は欠損値の状態になります。次に replace コマンドを使用して外国製の車の値を欠損値から正しい値に変更します。

変数 foreign はダミー変数なので、この変数 predprice は次に示すように 1 つのコマンドでも作成 できます。

```
. gen predprice2 = (1.05 + 0.05*foreign)*price
. list make foreign price predprice predprice2, nolabel
```

	make	foreign	price	predpr~e	predpr~2
1.	VW Rabbit	1	4697	5166.7	5166.7
2.	Olds 98	0	8814	9254.7	9254.7
3.	Chev. Monza	0	3667	3850.35	3850.35
4.		0	4099	4303.95	4303.95
5.	Datsun 510	1	5079	5586.9	5586.9
6.	Buick Regal	0	5189	5448.45	5448.45
7.	Datsun 810	1	8129	8941.9	8941.9

11.4 generate での文字列変数の作成

Stata は変数作成時の表現で文字列を認識すると、文字列をちょうど格納するのに必要な文字数分の保存タイプを用意します。次の列では変数 where は str1 です。

```
. list make foreign

     +-------------------+
     |        make   for |
     |-------------------|
  1. |   VW Rabbit   Fore |
  2. |     Olds 98   Dome |
  3. | Chev. Monza   Dome |
  4. |               Dome |
  5. |  Datsun 510   Fore |
     |-------------------|
  6. | Buick Regal   Dome |
  7. |  Datsun 810   Fore |
     +-------------------+

. gen where = "D" if foreign=="Domestic":origin
(3 missing values generated)
. replace where = "F" if foreign=="Foreign":origin
(3 real changes made)
. list make foreign where

     +-------------------------------+
     |        make   foreign   where |
     |-------------------------------|
  1. |   VW Rabbit   Foreign       F |
  2. |     Olds 98   Domestic      D |
  3. | Chev. Monza   Domestic      D |
  4. |               Domestic      D |
  5. |  Datsun 510   Foreign       F |
     |-------------------------------|
  6. | Buick Regal   Domestic      D |
  7. |  Datsun 810   Foreign       F |
     +-------------------------------+

. describe where
Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------
where           str1    %9s
```

Stata は文字列変数で作業するのに便利なツールを備えています。これから変数 make を make（メーカー）と model（モデル）に分けます。

```
. gen model = usubstr(make, ustrpos(make," ")+1,.)
(1 missing value generated)
. gen modelwhere = model + " " + where
. list make where model modelwhere
```

	make	where	model	modelw~e
1.	VW Rabbit	F	Rabbit	Rabbit F
2.	Olds 98	D	98	98 D
3.	Chev. Monza	D	Monza	Monza D
4.		D		D
5.	Datsun 510	F	510	510 F
6.	Buick Regal	D	Regal	Regal D
7.	Datsun 810	F	810	810 F

そして、分けた model と製造場所（変数 where）を合わせた変数を作成します。 上記コマンドの説明は次の通りです。

1. ustrpos(*s*1, *s*2)は文字列 *s*1 のなかで最初に文字列 *s*2 が見つかる位置を整数で返します。見つからなかったときは 0 を返します。上記例の ustrpos(make,″ ″)は変数 make の中でスペースが先頭から何番目に位置する、という情報を出力します。
2. usubstr(*s*, *start*, *len*) は、文字列 *s* の *start* 番目から *len* 文字分の文字列を取り出します。*len* =. の場合、*start* 番目から最後までの文字列になります。
3. 上記の 1 と 2 を同時に使うこともできます。usubstr(*s*, ustrpos(*s*,″ ″)+1, .)は、文字列 *s* の最初の 1 単語を除いた文字列を与えます。よって、上記例は「変数 make の全データから最初のスペースの次の文字から最後までを取り出す」という意味になり、たとえば 1 行目では“Rabbit”を抽出します。
4. 文字列に演算子“+”を使うと、その文字列（変数）をつなげて 1 つにします。つまり、表現″this″ + ″that″は″thisthat″という文字列になります。上記例では変数 modelwhere を作成した時、model + ″ ″ + where と入力したので変数 model と where の間にスペース (″ ″) が入りました。
5. 文字列変数では欠損値の扱いが他の変数とは異なり、空白の文字列（″ ″）という認識になります。変数 modelwhere でメーカーやモデルが分からない車（4 行目）では″ D″と表示します。(D の前にスペースがあります。)
6. Unicode 文字には専用の特別なコマンドが利用できます。詳細は **[U] 12.4.2 Handling Unicode strings** をご覧ください。

12 変数やデータを削除する

12.1clear, drop, keep コマンド

この章ではデータと変数を削除するツールを紹介します。データエディタでの操作方法は [GSM] **6 Using the Data Editor (**データエディタを使用する**)** で紹介しました。ここではコマンドウィンドウを使用する方法を見ていきましょう。

データや他のオブジェクト、例えば値ラベルをメモリから消すために Stata には clear、drop、keep の 3 つのコマンドがあります。このコマンドはメモリ内にだけ影響を与えます。つまり、ディスクに保存してあるものを変更することはありません。

12.2clear と drop all コマンド

仮に、何かの分析やシミュレーションを行う中でデータセットをメモリから消去し、別のデータセットをロードして使うとしましょう。その際に今まで使用していたデータセットの内容を保存する必要は無く、ただ何も入っていないデータセットを準備します。上記 3 つのコマンドの内どれを使用するべきか決めるには、何を行いたいのか把握する必要があります。これには普通のデータの他にも作成されるメタデータ（例えば値ラベル、保存した推定コマンド、保存した行列など）の扱いも含まれます。clear コマンドには多様なオプションがありますが、この章では簡単な使用方法だけを確認します。詳しくは help clear コマンドを使用するか、 [D] **clear** をご覧ください。

clear コマンドを実行すると、開いているデータセットの全ての変数と値ラベルを削除します。これが通常の用法です。ディスクに保存した結果は削除しないので、新しいデータセットをロードしてから保存した推定結果などを再度分析に利用できます。詳しくは help postest コマンドを実行するか、[U] **20 Estimation and postestimation commands** をご覧ください。

メモリ上のデータ等、全てを確実に削除したい場合は clear all コマンドを使用してください。このコマンドは Stata のメモリ上のデータおよびメタデータを削除するので、何もない状態から作業を開始できます。最初、グラフウィンドウなどが開いている状態でコマンドを使用すると、そのウィンドウは自動的に閉じるので、驚くかもしれません。この動作で使用中のメモリを空にします。データセットだけを削除して他の物（例えば値ラベルなど）をそのまま残しておきたい場合、drop all コマンドを使用してください。

12.3 drop コマンド

drop コマンドは変数やデータをデータセットのメモリから削除します。

- 変数を取り除く場合、drop *varlist* を使用します。

• データを取り除く場合、drop コマンドを *if* か *in* または両方の条件と共に使用します。
サンプルデータセット afwecarslab を使用して、drop の例を挙げます。

```
. use afewcarslab
(A few 1978 cars)
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
2.	Olds 98	8814	21	4060	2.41	Domestic
3.	Chev. Monza	3667	.	2750	2.73	Domestic
4.		4099	22	2930	3.58	Domestic
5.	Datsun 510	5079	24	2280	3.54	Foreign
6.	Buick Regal	5189	20	3280	2.93	Domestic
7.	Datsun 810	8129	.	2750	3.55	Foreign

```
. drop in 1/3
(3 observations deleted)
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.		4099	22	2930	3.58	Domestic
2.	Datsun 510	5079	24	2280	3.54	Foreign
3.	Buick Regal	5189	20	3280	2.93	Domestic
4.	Datsun 810	8129	.	2750	3.55	Foreign

```
. drop if mpg > 21
(3 observations deleted)
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.	Buick Regal	5189	20	3280	2.93	Domestic

```
. drop gear_ratio
. list
```

	make	price	mpg	weight	foreign
1.	Buick Regal	5189	20	3280	Domestic

```
. drop m*
. list
```

	price	weight	foreign
1.	5189	3280	Domestic

メモリ内のデータにのみこの操作は適用されます。データセットを保存すると変更内容を確定できます。

12.4 keep コマンド

keep コマンドは選択した変数または if や in 表現で指定したデータを除く、全ての変数を削除します。keep コマンドは drop コマンドのように *varlist* または if や in 表現と共に使用できます。しかし、1 つのコマンドで使用できるのは *varlist* または if や in 表現のどちらかだけです。この例の中ではまず clear コマンドを使用して、再びデータセット afewcarslab をロードしてから作業を始めています。

```
. clear
. use afewcarslab
(A few 1978 cars)
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
2.	Olds 98	8814	21	4060	2.41	Domestic
3.	Chev. Monza	3667	.	2750	2.73	Domestic
4.		4099	22	2930	3.58	Domestic
5.	Datsun 510	5079	24	2280	3.54	Foreign
6.	Buick Regal	5189	20	3280	2.93	Domestic
7.	Datsun 810	8129	.	2750	3.55	Foreign

```
. keep in 4/7
(3 observations deleted)
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.		4099	22	2930	3.58	Domestic
2.	Datsun 510	5079	24	2280	3.54	Foreign
3.	Buick Regal	5189	20	3280	2.93	Domestic
4.	Datsun 810	8129	.	2750	3.55	Foreign

```
. keep if mpg <= 21
(3 observations deleted)
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.	Buick Regal	5189	20	3280	2.93	Domestic

```
. keep m*
. list
```

	make	mpg
1.	Buick Regal	20

13 do ファイルエディタを使用する–Stata の自動化

13.1 do ファイルエディタ

Stata には do ファイルエディタと呼ばれる統合型テキストエディタがあり、様々なタスクで利用できます。do ファイルエディタという名前は do ファイルという用語から取っています。他のアプリケーションではバッチファイル（batch file）やスクリプト（script）とも呼ばれるものと同じで、do ファイルは Stata の実行コマンドで構成されます。詳しくは [U] **16 Do-files** をご覧ください。do ファイルエディタにはプログラム上の高度な機能が用意されていますが、それだけではなく複数のコマンドから長いプログラムを作成するための機能を用意しています。この機能は、複数の変数に対し同じ手順の操作を繰り返し行うループ設定や、複雑で繰り返しの伴うタスクをインタラクティブに行うのに適しています。

この章の内容を最大限活用していただくには、ご自分のコンピュータで操作しながら読み進めてください。まずは do ファイルエディタを開くところから始めましょう。do ファイルエディタは **do** ファイルエディタボタン  をクリックするか、コマンドウィンドウで「doedit」を実行します。

13.2 do ファイルエディタツールバー

do ファイルエディタには 7 個のボタンがあります。その多くは Stata のメインツールバーのボタンと似ており、ほぼ同じ機能を備えています。

ボタンの機能を忘れてしまった時はマウスカーソルをボタンの上に移動すると、ツールチップを表示します。

開く ディスク内の do ファイルを新しいタブで開きます。

保存 現在のファイルをディスクに保存します。印刷 do ファイルエディタの内容を印刷します。

検索 テキストを検索するための検索ダイアログを開きます。

表示 見えない表示の文字を表示します。

ズーム テキストの表示サイズを変更します。

実行 **(Do)** do ファイルのコマンドをすべて実行し、その途中経過を含む全ての出力とコマンドを結果ウィンドウに表示します。テキストの一部を選択する時はボタン

名が選択範囲を実行 **(Do)** に代わり、その選択部分のみを実行してコマンドの途中経過も全て表示します。以後このボタンを **Do** ボタンと呼びます。

13.3 do ファイルエディタを使用する

では、これから **[GSM] 1 Introducing Stata—sample session (Stata の紹介—サンプルセッション)** で行ったように、1978 年製の自動車の燃費について分析しましょう。分析中に大量のコマンドを使用しますが、後でコマンドを再入力せずに同じ分析を繰り返すことを考慮して do ファイルを作成することにします。

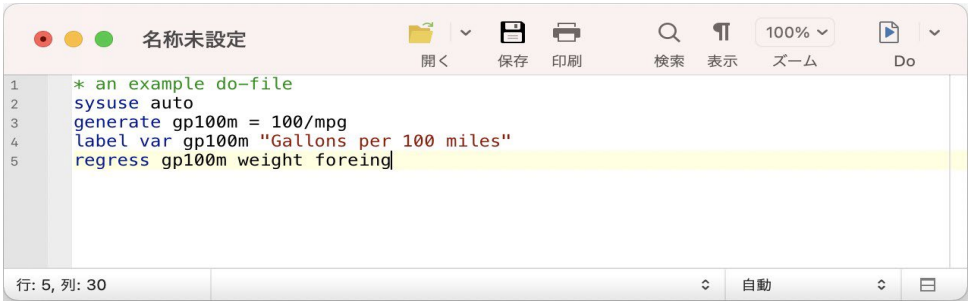
do ファイルはコマンドを順番に記入していくだけで作成できます。do ファイルでは、各コマンドを上から順に実行します。詳しくは **[U] 16 Do-files** をご覧ください。

1978 年製の自動車の燃費を分析するため、100 マイル（約 160km）あたりに使用するガソリンの量を表す、新しい変数を作成します。この変数と車重の関係について、国内製と外国製のグループ間の差を見ていきます。まずは新しい変数で回帰を行います。

操作を始めるには、まず **do** ファイルエディタボタンを押して編集画面を開きます。do ファイルエディタが開いたら、以下のコマンドを画面に打ち込んでください。５行目にある変数 foreign は意図的にスペルを間違えています。（ここではよくある間違いをわざと起こしてその解決策を示します。後程、実際に使い始める時に参考にしてください。）

```
* an example do-file
sysuse auto
generate gp100m = 100/mpg
label var gp100m ″Gallons per 100 miles″
regress gp100m weight foreing
```

データ入力後の do ファイルエディタの画面は次のようになります。



do ファイルエディタに文字を入力していくと、必要に応じて文字の色が変わります。do ファイルエディタには入力した情報によって表示色を変更する構文ハイライト機能があります。構文要素の色やテキストのプロパティを変更するには、do ファイルエディタ上で右クリックをしてユーザ設定... を操作し、表

示されるウィンドウで色タブをクリックします。キーワードハイライトの定義はユーザ独自に設定することもできます。

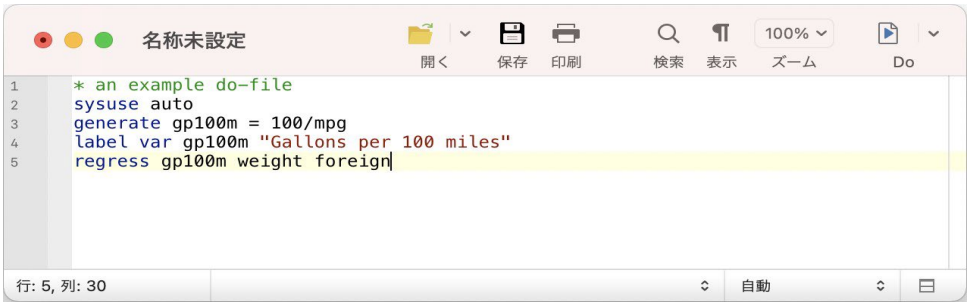
構文のハイライト機能は Stata のコマンドだけが対象ではありません。do ファイルエディタの右下のドロップダウンメニューをクリックして、言語を変更することで、ハイライト機能を切り替えることができます。Stata ではドロップダウンメニューから、Markdown を選択して動的なドキュメントを作成できますので、Markdown のハイライトも可能となっています。詳細は [PRT] **dynodoc** をご覧ください。また、Stata は Python、Java を呼び出すことができますので、python と Java のハイライトも可能です。詳細は [P] **Pystata integration** をご覧ください。Stata は編集しているファイルの拡張子を元にデフォルトの設定を行います、ファイルに拡張子がまた設定されていない場合は手動で言語の設定を行う必要があります。

また、do ファイルエディタには一度入力した言葉をオートコンプリートする機能があります。候補リストが表示されたら、さらに入力を進めて、候補を狭めることができます。候補リストでは上・下矢印キーで選択するか、入力を続けて 1 つの候補に絞り込みます。候補が 1 つになったら、リターンキーで do ファイルに入力します。

それでは、**Do** ボタン  をクリックしてコマンドを実行します。Stata はコマンドを上から順に実行し、結果を結果ウィンドウに表示します。

```
. do /tmp/SD00001.000000
. * an example do-file
. sysuse auto
(1978 automobile data)
. generate gp100m = 100/mpg
. label var gp100m "Gallons per 100 miles"
. regress gp100m weight foreing
variable foreing not found
r(111);

.
end of do-file
```

do ″/tmp/...″ コマンドは Stata が do ファイルエディタからどのようにコマンドを実行したのか を示しています。Stata はこれらのコマンドを一時的なファイルとして保存し、未保存の変更済み do コマンドを使用して実行します。上手くいったかのように見えましたが、Stata はエラーを返してきました。初めの 3 行分のコマンドは問題なく実行できましたが、4 行目のわざとスペルミスをしたコマンドでエラーが発生します。Stata は foreing という誤った名前の変数を見つけることができません。よって、do ファイルエディタに戻ってスペルミスをした最後の行の変数名を foreign と正しく入力しましょう。



再び **Do** ボタンをクリックします。すると、今度は do ファイルの 1 行目で中断しました。既存のデータセットを閉じなければ Stata で別のファイルを開くことは出来ないからです。

```
. do /tmp/SD00001.000000
. * an example do-file
. sysuse auto
no; dataset in memory has changed since last saved
r(4);
.
end of do-file
```

この状態では次のどちらかの操作を行ってください。

- do ファイルの一番初めに clear コマンドを追加します。これで Stata は自動的にメモリをクリアしてからデータセット auto.dta をロードします。これは便利ですが、Stata は予告なしにデータを消去するので注意が必要です。
- clear コマンドを手動で入力した後に do ファイルを改めて実行します。複雑な do ファイルを作成する時には、この手法は少し面倒かもしれません。

簡単なアドバイスをすると、do ファイルの作成中（デバック中）は 1 番目の方法（自動的にメモリを削除）を選択し、do ファイルが完成した段階では、次に述べるようにその内容に応じてどちらの手法を使うかを決めてください。かなりの部分を自動化した場合は、do ファイルが自動的にメモリを削除する方が便利でしょう。do ファイルの使用頻度が低い場合は、自動的にメモリを消さない方が安心です。このアドバイスを参考に clear の使用について検討してください。今回は clear オプションを sysuse コマンドと組み合わせて do ファイル実行前に自動的にメモリをクリアします。

Do ボタンをクリックして do ファイルを実行します。

```
. do /tmp/SD00001.000000
. * an example do-file
. sysuse auto, clear
(1978 automobile data)
. generate gp100m = 100/mpg
. label var gp100m "Gallons per 100 miles"
. regress gp100m weight foreign
      Source |       SS       df       MS              Number of obs =      74
-------------+------------------------------           F(  2,    71) =  113.97
       Model |  91.1761694     2  45.5880847            Prob > F      =  0.0000
    Residual |  28.4000913    71  .400001287           R-squared     =  0.7625
-------------+------------------------------           Adj R-squared =  0.7558
       Total |  119.576261    73  1.63803097           Root MSE      =  .63246

------------------------------------------------------------------------------
       p100m | Coefficient  Std. err.      t    P>|t|     [95% conf. interval]
-------------+----------------------------------------------------------------
      weight |   .0016254   .0001183    13.74   0.000     .0013896    .0018612
     foreign |   .6220535   .1997381     3.11   0.003     .2237871    1.02032
       _cons |  -.0734839   .4019932    -0.18   0.855    -.8750354    .7280677
------------------------------------------------------------------------------

.
end of do-file
```

結果が出力できたところで、 do ファイルエディタを最前面にしてファイル > 名前を付けて保存... と選び、do ファイルを保存します。分析を進めながら do ファイルにコマンドを追加するには、メニューからファイル > 開く... と選択します。分析を行いながら do ファイルにコマンドを書き加えていけば、毎回新しい Stata のセッションでコマンドを入力し直す手間を省けます。一番初めの clear オプションの削除については do ファイルの使い方に応じてよく考えてから決めてください。

do ファイルを保存後に、改めて do ファイルを実行するにはコマンドウィンドウに「do ファイル名」と入力します。この場合、ファイル名は保存した do ファイルです。

13.4 ファイルメニュー

do ファイルエディタが最前面にあるときにファイルメニューからの操作をすると、do ファイルに反映されます。メニュー内には新規ファイルを作成する「新規 > **do** ファイル」、既存のファイルを開く「開く**...**」、ファイルを上書き保存する「保存」、名前を付けてファイルを保存する「名前を付けて保存**...**」、ファイルを印刷する「印刷」があります。また、do ファイルエディタのツールバーにはそれぞれの機能に対応するボタンがあります。

さらに、新規 > プロジェクトによりプロジェクトを作成し、一連のファイルをまとめて管理することもできます。プロジェクトには do ファイル、データファイル、グラフファイルなど、必要なファイルを含めることが可能です。プロジェクトマネージャに関する詳細は [P] **Project Manager** をご覧ください。

13.5 編集メニュー

do ファイルエディタの編集メニューには、標準的な元に戻す、やり直す、切り取り、コピー、貼り付け、検索のコマンドがあります。他にも、編集メニューには次のような機能があります。

- ファイルの挿入**...** 他の do ファイルのコマンドをカーソル位置に挿入します。
- 行を選択 現在の行を選択します。
- 行の削除 現在の行を削除します。
- 検索 > 行に移動**...** 指定の行番号に移動します。行番号は do ファイルエディタウィンドウの左端または左下にあります。
- 上級設定 はプログラマー向けのサブメニューを表示します。
  - 右にシフト 1 タブ分のインデントを用意します。
  - 左にシフト 1 タブ分のインデントを解除します。
  - インデント解除 ネスト構造の親に相当する位置にインデントを調整します。
  - コメント/非コメントの切り替え 選択した行頭に//を追加/削除してコメント/非コメント化します。
  - ブロックコメントを追加 選択したブロックの前後に/* */を追加してコメント化します。
  - ブロックコメントを削除 選択したブロックの前後の/* */を削除して非コメント化します。
  - 選択部分を大文字に 選択した文字を全て大文字にします。
  - 選択部分を小文字に 選択した文字を全て小文字にします。

- 完成語 do ファイルエディタに入力されている語をもとに現在の語をオートコンプリートします。候補が複数ある場合は、それら全てを表示します。候補が表示されたら、1 つを選択するか、入力を続けて候補を狭めていきます。
- エンコード形式を **UTF-8** に変換**...** 現在のファイルを UTF-8 エンコード形式に変換します
- 改行コードを **Mac OS X/Unix** 形式に変換 **(\n)** 現在のファイルの改行コードを Mac OS X/Unix 形式に変換します。
- 改行コードを **Windows** 形式に変換 **(\r\n)** 現在のファイルの改行コードを Windows 形式に変換します。
- タブをスペースに変換する 既存の間隔を維持したまま、タブ文字をスペースに変換します。
- 先頭のスペースをタブに変換する 先頭のスペースをタブ文字に変換します。タブあたりのスペースの数は環境設定に従います。
- すべてのスペースをタブに変換する スペースをタブに変換します。タブあたりのスペースの数は環境設定に従います。
- Unicode スペースと曲線型の引用符を ASCII に変換する Unicode のスペースと曲線引用符を SCII でエンコードします。

小括弧（）、中括弧 { }、大括弧[ ]のマッチとバランス確認も編集メニューから選択できます。メニューから編集 > 検索 > 中括弧のマッチを選択すると、do ファイルエディタはカーソルの左右にある記号を確認します。どちらかの記号が { } の場合、その括弧とマッチする（組になる）括弧の直前にカーソルを移動します。マッチしない場合、カーソルは移動しません。

メニューから編集 > 検索 > 中括弧のバランスと選ぶと、do ファイルエディタはまずカーソルおよび選択箇所の左右を確認します。そして最寄りの { } を含む部分をハイライトします。もう 1 度中括弧のバランスを選ぶと、次の { } まで選択範囲を拡大します。マッチする記号が無い場合、カーソルは移動しません。{ } を確認する機能は、プログラミングを行う際に if 条件を使ったり、ループするような表現やコードのまとまりで作業する際に便利です。詳しくは [P] **foreach** , [P] **forvalues** , [P] **while** , [P] **if** をご覧ください。

中括弧のバランスについては次の例題で確認してください。do ファイルエディタに「 (now (is the) time)」と入力し、is と the の間にカーソルを置きます。メニューから編集 > 検索 > 中括弧のバランスと操作します。do ファイルエディタは(is the)を選択します。もう 1 度中括弧のバランスを選択すると、今度は(now (is the) time)を選択します。

Stata では文字列に Unicode 文字を含めることができます。Unicode 文字のエンコード形式には UTF-8 が用いられます。（詳しくは [U] **12.4.2 Handling Unicode strings** をご覧ください。）ただし、Stata 13 およびそれ以前のバージョンで作成した do ファイル、ado ファイルを含むテキストファイルには、アクセント付き文字、中国語、日本語、韓国語、キリル文字等を含む非 ASCII 文字が

UTF-8 でエンコードされていないことがあります。そうしたファイルでも、Stata 15 以降の do ファイルエディタで開くと、エンコード形式を選択する画面になり、選択した形式から UTF-8 へとエンコード形式を変換する処理が行えます。途中で変換をキャンセルしたり、誤ったエンコード形式を指定するなどしたときは、エンコード形式を **UTF-8** に変換**...** と選択すると再度変換ができます。変換は編集 > 元に戻すを選択してやり直すことができ、保存しなければ元のファイルが上書きされることはありません。Stata データセットを変換したり、複数の Stata ファイルを一括で変換する場合には、unicode translate コマンドをご利用下さい。

編集ヒント：右下にあるエディタの分割ボタン  をクリックすると、do エディタウィンドウを上下に分割できます。この機能は遠く離れた場所を同時に見る時に便利です。分割を戻すには、どちらかのペインの右下にあるボタンを再びクリックします。

13.6 表示> do ファイルエディタメニュー

Do ボタンについては既に紹介した通りです。メニューで表示 > **do** ファイルエディタ > 実行 **(Do)** と選択するのは実行 **(Do)** ボタンをクリックするのと同じです。

表示 > **do** ファイルエディタ > 最初から実行を選択すると、1 行目から現在選択しているの行までを実行します。これは、do ファイルの一部のみ実行する際に便利な方法です。

また、表示 > **do** エディタファイル > 最後まで実行 **(do)** と選択するとカーソルの行から do ファイルエディタ内のコマンドを最後まで実行します。これは do ファイルの一部を素早く実行するのに適しています。

表示 > **do** ファイルエディタ > サイレント実行 **(run)** は表示 > **do** ファイルエディタ > 実行 **(do)** と同様だが、コマンドは quietly に実行され、コマンドウィンドウには何も表示されません。

表示 > **do** ファイルエディタ > 行を実行 (run) は現在の行のコマンドをすべてコマンドウィンドウに送ります。その後カーソルは空白行とコメントを無視して、次の行に自動的移動します。この方法は 1 行ごとに do ファイルを実行するための簡単な方法です。

表示 > **do** エディタファイル > インクルード実行 **(include)** を選択すると、ローカルマクロ変数を展開して実行 **(do)** と同様の動作を行うことができます。

実行 **(Do)** は Stata の do コマンドと同じものです。詳しくは [U] **16 Do-files** をご覧ください。

また、do ファイルエディタメニューから表示 > **do** ファイルエディタ > ファイルをビューワで表示と選択すると、ビューワでファイルのプレビューができます。この機能は Stata の SMCL タグを使用するファイル、例えばヘルプファイルや log ファイルの作成や編集をする時に便利です。

13.7 インタラクティブコマンドを do ファイルとして保存する

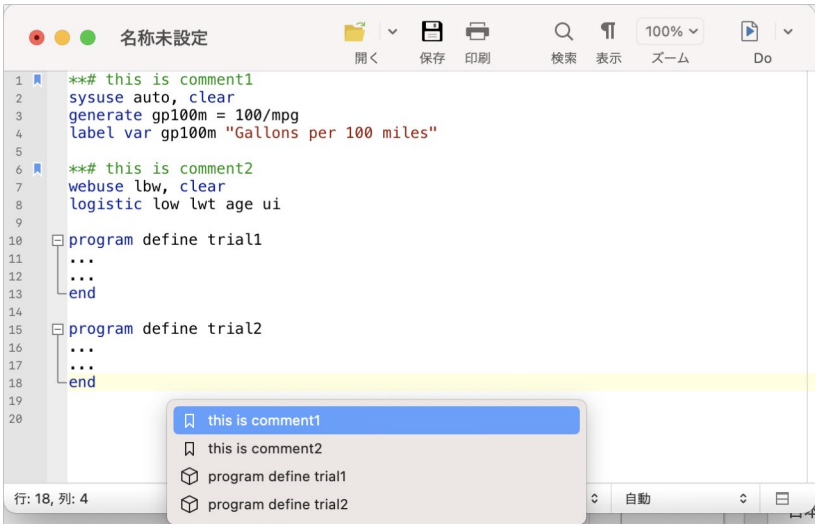
Stata でインタラクティブに作業を行っていると、少し前のコマンドを再実行したいと思うこともあるでしょう。履歴ウィンドウにある一部のコマンドまたは全てのコマンドを選択し、do ファイルエディタに送ることができます。別の方法としては、再実行したいコマンドを do ファイルとして保存し、do ファイルエディタで開くこともできます。また、結果ウィンドウからコマンドをコピーして do ファイルエディタに直接貼り付ける事もできます。詳しくは [GSM] **6 Using the Data Editor (**データエディタを使用する**)** をご覧ください。更に、 [R] **log** には cmdlog（コマンドログ）についての記載があります。cmdlog コマンドを使うと Stata に入力する全てのコマンドを do ファイルに記録できます。

13.8 do ファイル内を移動する

大きなファイルで作業を行う時、ブックマークがあると do ファイル内を簡単に移動できます。重要なセクションの前にブックマークを配置することで、後からそのセクションに簡単に戻ることができます。ブックマーク行は編集 > 検索 > ブックマークの検索/削除と操作するか、または、行頭をクリックすると、コメントアウト記号、**# が入力されます。この表に入力されるテキストは、ブックマークのタイトルとして扱われます。ブックマーク行は実行時に無視されますので、この行にコードを追加することは出来ません。//#を使用してコメント付きブックマークを設定することができます。**#は Mata や Java でも使用されるため、//#の使用がより望ましいと言えます。

ブックマーク間を移動するには、編集メニューまたはナビゲーションコントロールを使用します。ナビゲーションコントロールでは、ブックマークやプログラムや Java、Python 間を簡単に移動することができます。ナビゲーションコントロールでプログラムやブックマークを選択すると、選択下箇所にジャンプできます。

13.9 プロジェクト



ナビゲーションコントロールでブックマークに#を追加することで、ブックマークラベルのインデントを追加できます。例えば、ブックマークコメント**## Bookmark 2 は、ブックマークコメント**# Bookmark 1 に 1 段階インデントを追加することで作成できます。

編集 > 検索 > ブックマークの検索/削除を選択、または行を削除することでブックマークを削除できます。ブックマークを追加すると、余白にブックマークアイコンが追加され、スクロール中に見付けやすくなります。

13.9 プロジェクト

上級 Stata ユーザで多くのファイルをプロジェクトの一部として管理している場合、Stata には do ファイルエディタで使用できるプロジェクトマネージャがあります。プロジェクトマネージャに関する詳細は [P] **Project Manager** をご覧ください。

13.10 自動バックアップ

do ファイルエディタは文章を開くか、新規作成するとバックアップを作成します。既存のドキュメントを開く場合、Stata は元ファイルと同じディレクトリに、同じファイル名に~接頭辞を追加したファイル名で.stswp 形式のバックアップを作成します。新規作成または保存していないドキュメントを編集する場合は、temp フォルダに保存されます。ドキュメントを閉じると、バックアップは削除されます。しかし、コンピュータの電源が切れたり、クラッシュしたりきちんと Stata が閉じられなかった場合は、バックアップファイルは残ってしまいます。

デフォルト設定では、編集を行ったり 200 文字以上の追加または削除を行うと 4 秒ごとにバックアップを取ります。do ファイルエディタの上級設定で、この間隔は変更できます、またバックアップ機能そのものを停止することもできます。

do ファイルエディタでドキュメントを開くとき、Stata はまずバックアップがあるかを確認します。もしバックアップが見つかると、do ファイルエディタはユーザにバックアップの存在を知らせ、このバックア

ップを復元するか、オリジナルのドキュメントを開くか、キャンセルかのいずれを選ぶかを尋ねます。キャンセルを選択すると、ドキュメントを開かず、ディスクのバックアップをそのままにします。オリジナルのドキュメントを開くを選ぶと、doファイルエディタでオリジナルのドキュメントが開かれバックアップは削除されます。バックアップから復元を選ぶと、バックアップ ファイルは、デフォルトのファイル名が元のファイル名に設定され、ファイル名に文字列 Recovered が追加された状態で、Do ファイル エディタで新しい未保存のドキュメントとして開かれます。バックアップはディスクから削除されます。復元されたドキュメントは、新しいファイルとしてディスクに保存するか、元のドキュメントを上書きすることで保持できます。また、ドキュメントを保存せずに閉じて変更を無視することもできます。

13.11 構文ハイライトにユーザ定義のキーワードを追加する

Stata が構文の強調表示に使用するキーワードを含むテキストファイルを作成できます。テキストファイルには statauserkeywords.txt という名前を付け、スペースまたはタブの任意の組み合わせで区切ることができ、別の行に配置できるキーワードのリストを含める必要があります。キーワードは、有効なコマンド名に関する Stata のルールに従う必要があります。 コメントはサポートされていません。無効なコマンド名であるキーワードは無視されます。定義できるキーワードの数に制限はありません。ただし、ドキュメントを構文強調表示する場合、Stata はすべてのキーワードを検索する必要があるため、キーワードの非常に大きな辞書が do ファイルエディタのパフォーマンスに影響を与える可能性があることに注意する必要があります。Stata は一意のキーワードの辞書を保持しているため、キーワードの繰り返しのインスタンスは無視されます。

Stataは、グローバルキーワードファイルとローカルキーワードファイルの両方を検索します。 両方のファイルが存在する場合、それらのキーワードの辞書がマージされます。グローバルキーワードファイルはStata ディレクトリに保存する必要があります。これにより、各ユーザがコピーを持つ必要がなく、グローバルキーワードファイルを複数のユーザと共有できるようになります。独自のローカルキーワードファイルを作成することもできます。このファイルはホームディレクトリに保存する必要があります。Stata for Mac ユーザは、ドキュメントフォルダ内の Stataフォルダにローカルキーワードファイルを保存することもできます。

Stata は起動時にキーワードファイルを読み取ります。Stata の実行中にキーワードファイルに変更を加えた場合、それを有効にするにはStataを再起動する必要があります。

14 データを作図する

14.1 グラフを使う

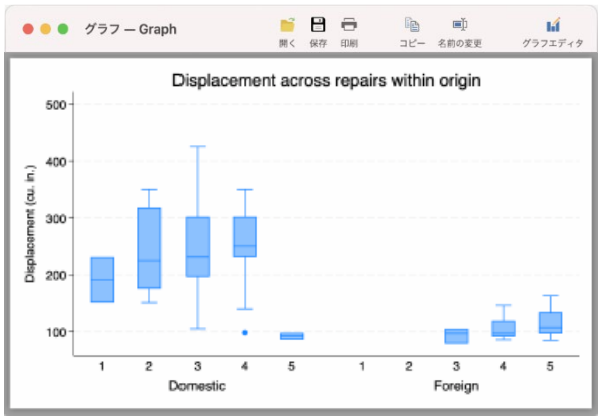
Stata はデータからグラフを作成するためのシステムを幅広く取り揃えています。グラフを作成するのに最も重要なコマンドは graph コマンドです。Stata ではこのシンプルなコマンドと様々なオプションを組み合わせてグラフを作図します。この章では例題として簡単なグラフを 1 つ作成し、Graph ウィンドウの基本を確認します。グラフ作成に関する詳細は [G] Stata Graphics Reference Manual をご覧ください。

14.2 簡単なグラフのサンプル

[GSM] **1 Introducing Stata—sample session (Stata** の紹介**—**サンプルセッション**)** のサンプルセッションでは、散布図を作成し、それにフィットした回帰線を追加してグリッドを作り、グループごとに比較しました。ここでは、データセット auto を使用して簡単なボックスプロット（箱ひげ図）を作成します。データセット auto の排気量と修理記録の関係をグループ間の差からみていきましょう。
では、コマンドウィンドウに「sysuse auto」と入力して Return を押し、データセットを開きます。

メニューからグラフィックス ＞箱ひげ図を選択し、メインタブを選んで、変数欄に「displacement」を入力します。次にカテゴリタブを選び、グループ **1** のチェックボックスにチェックを付けて 1 番目のグループ変数を「rep78」にします。続いてグループ **2** のチェックボックスにチェックを付けて、「foreign」を 2 番目のグループ変数に入力します。最後に、必要に応じてグラフを修正できるように適用ボタンをクリックします。完成したグラフにタイトルを付け忘れたので、ここで追加しましょう。Graph ウィンドウを 1 度閉じて **graph box** ダイアログでタイトルタブを選びます。そしてタイトルに「Displacement across repairs within origin」と入力し、再び適用ボタンをクリックします。

この一連の作業でタイトル付のグラフが出来上がります。



14.3 Graph ウィンドウ

Graph ウィンドウは作図したグラフとツールバーを表示します。最初の 4 つのボタンは他のウィンドウでも見慣れている開く、保存、印刷、コピーのボタンです。残り 2 つのボタンは今回初めてなのでここで説明しておきます。

名前の変更 グラフの名前を変更します。この機能は、複数のグラフを画面上で同時に開く時に利用します。名前の変更ボタンをクリックするとグラフに名前を付けることができるので、別のグラフを作成してもこのウィンドウは開いたままになります。

グラフエディタの開始 グラフの編集と加工をするためのグラフエディタを開きます。機能については次の章で紹介します。

では作成したグラフを保存しましょう。保存ボタンをクリックし、ダイアログでフォルダを選んで名前を入力します。Graph ウィンドウ内を右クリックしてコンテキストメニューから名前を付けて保存... としても同じ要領で保存できます。

14.4 グラフの保存と印刷

グラフ作成後ならウィンドウ内を右クリックし、名前を付けて保存... を選ぶと保存でき、グラフ表示後にウィンドウ内で右クリックをして印刷... を選ぶと印刷できます。ファイルメニューから保存または印刷を選択しても、保存または印刷できます。

14.5 Graph ウィンドウで右クリックする

Graph ウィンドウで右クリックをすると、コンテキストメニューに次の項目を表示します。

- 名前を付けて保存... グラフをディスクに保存します。
- コピー クリップボードにグラフをコピーします。
- グラフエディタの開始 グラフエディタを起動します。
- ユーザ設定... グラフの設定を編集します。
- 印刷... Graph ウィンドウの内容を印刷します。

14.6 Graph ボタン

Graph ボタン  はメインツールバーにあります。ボタンをクリックすると開いている Graph ウィンドウのうち、一番手前にあるウィンドウを最前面に移動します。ボタンを長押しすると開いているグラフウィンドウの一覧を表示します。この一覧からウィンドウを 1 つ選択すると、その選択したグラフが最前面に移動します。Graph ウィンドウを保存せずに閉じてから再び同じグラフを開くには、そのグラフを再作成しないといけません。

15 グラフを編集する

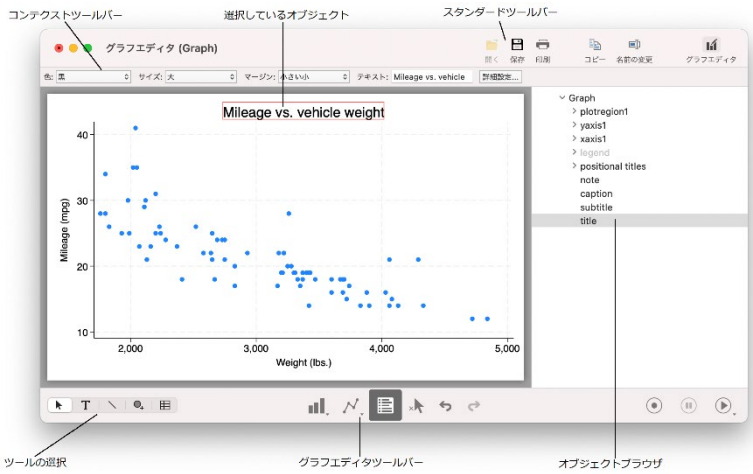
15.1 グラフエディタで作業をする

Stata のグラフエディタではグラフ上にテキスト、線、矢印、マーカーなどを自由に追加でき、グラフ上のオブジェクトのほぼ全てが編集可能です。

まずは実際に編集するために例題のグラフを準備しましょう。それからグラフエディタ内のツールを確認します。では、「sysuse auto」コマンドを実行して、データセット auto を開きます。グラフを作成するためのコマンドは次の通りです。

```
. scatter mpg weight, name(mygraph) title(Mileage vs. vehicle weight)
```

グラフエディタを開くには、Graph ウィンドウで右クリックを行いグラフエディタの開始を選択します。グラフのタイトルを１回クリックしてください。グラフエディタとそれぞれの要素の名前を次に示します。



グラフエディタウィンドウの左下にあるツールを選択してグラフを編集します。デフォルトではポインタ (選択ツール)  が選択されています。

ポインタを使うとオブジェクトの設定を変更でき、そのオブジェクトをグラフ上でドラッグできます。グラフ内のオブジェクトをポインタで選択するとグラフのすぐ上にコンテキストツールバーを表示します。上記例題では、グラフタイトルを選択したので、コンテキストツールバーはタイトルを編集するための内容に変化します。コンテキストツールバーでは選択したオブジェクトの主要なプロパティを簡単に変更できます。オブジェクトを右クリックすると更に細部にわたってプロパティと操作の変更を行えます。Shift キーを押しながらオブジェクトをドラッグすると、水平または垂直方向に移動できるようになります。

テキスト、直線、マーカー（ラベル付けオプションあり）を追加するには、次の追加ツール  ,  ,  を使用します。直線はコンテキストツールバーで矢印に変更できます。テキスト、線、マーカーを追加する前にコンテキストメニューでプロパティを変更すると、デフォルトの設定も変更できます。設定を行うと、このセッション以降、追加される全てのオブジェクトのデフォルト設定となります。

ツールに慣れるためにも、いろいろと試してみてください。結果が気に入らない時は、同じツールを使用して元に戻すか、標準ツールバー（メインメニューの下）にある元に戻すボタン  をクリックします。メインメニューで編集 > 元に戻すと操作しても同じです。

オブジェクトのドラッグやプロパティを変更する時は、ポインタ（選択ツール）を再度選択する必要があります。

グリッド編集ツール  を選択すると、グラフ上のオブジェクト（軸ラベルや軸タイトル）を赤い罫線で囲まれたエリアに移動できます。この赤い罫線はグリッドと呼ばれ、オブジェクトの移動可能位置を表します。複数のグラフを同じウィンドウ内で表示する時（例えば、by グラフを作成する時）は選択するオブジェクトにより、移動可能なグリッドが異なることもあります。オブジェクトはグリッドの外および他のグリッドへの移動はできません。

グラフの右側にある Object Browser を使用するとオブジェクトを選択できます。このウィンドウはグラフ内のオブジェクトの階層を表示します。Object Browser でオブジェクトをクリックまたは右クリックすると、グラフ上でそのオブジェクトに同じことをしている状態になります。

グラフエディタには編集内容を記録して後から他のグラフに反映させる機能があります。記録を開始ボタン  をクリックすると、グラフに対して行った編集をすべて記録します。これには「元に戻す」と「やり直す」も含みます。記録しない編集を行う時は、記録を中断ボタン  をクリックします。記録を開始ボタンをもう 1 度クリックすると、記録を再開します。記録したい内容が終了したら記録を開始ボタンをもう 1 度押します。ポップアップを表示するので記録を保存します。記録後保存した内容は記録を再生ボタン  で見ることができ、それ以後に作成するグラフにも適用できます。記録したものを再生するには、Stata の graph コマンドに play オプションを付けます。詳しくは [G-1] **Graph editor** 内の「 Graph Recorder 」をご覧ください。

グラフエディタを終了するにはメインメニューからファイル > グラフエディタの終了と操作するか、グラフエディタの終了ボタンをクリックします。グラフエディタを終了する時にグラフに変更がある場合、ポップアップを表示してグラフを保存するように促します。グラフを保存しなくても、すぐに変更内容が消えることはありません。しかし、同じウィンドウで新しいグラフを作成すると内容が消えるリスクが高くなります。Stata で他のタスクを実行する場合はエディタを中止しなければなりません。

グラフエディタで行える操作一覧の一部を次に示します。

- 線、矢印、テキストを使用して注釈を追加できます。

- グリッドラインやリファレンスラインの追加および削除ができます。
- タイトル、キャプション、メモの追加および編集ができます。
- 散布図（scatterplot）を線（line）、線 + シンボル（connected）、面積図（area）、棒（bar）、スパイク（spike）、または垂直ドロップライン（drop line）に相互変更できます。
- グラフタイトルおよびテキストのサイズ、色、余白、を含むプロパティを変更できます。
- 凡例をグラフ内のどこにでも配置できます。
- グラフの縦横比を変更できます。
- 棒グラフを積み上げに変更することや、その表示をパーセントに変えることができます。
- 軸ラベルの角度を回転や変更できます。
- カスタムラベルや目盛りを軸に追加します。
- 軸のラベルと目盛りの数および間隔に関するルールを変更します。
- グラフ上のある一点の色、サイズ、シンボルをカスタム化して強調できます。この機能はマーカー、棒、スパイク等のグラフで行えます。
- マーカーラベルのテキストやプロパティを変更します。

グラフ上の全オブジェクトのプロパティを編集できるので、結果的にグラフのほぼ全ての内容を変更できます。詳しくは [G-1] **Graph editor** をご覧いただくか、「help graph editor」を実行してください。

16 ログを使い結果の保存や印刷を行う

16.1 Stata でログを使う

Stata で分析を行う時、科学者が実験ノートを取るのと同じように操作を記録しておくと、その分析を簡単に繰り返す事ができます。集中して作業を行う最中はあたかも全てのやり方や情報を理解したような気になります。ところが、後日この分析を再現しようとすると、重要部分の詳細をあいまいに記憶していることがあります。このような状況を回避するために、Stata には実験ノートの代わりとしてログファイルがあります。

ログファイルは、結果ウィンドウの内容を記録したものです。全てのコマンドとテキストによる出力を実行順に記録します。つまり、作業しながら実験ノートを自動で作成できます。ログファイルは結果ウィンドウに出力しながらディスク上に作成されるので、万一停電やコンピュータがクラッシュするような事故が起きても、それまでの作業内容は消えません。Stata で重要な作業を行う時はログファイルを作る事をお勧めします。

16.2 ロギング出力

結果ウィンドウに表示される出力はすべてログファイルに記録できます。ログファイルのファイル形式は 2 種類あります。デフォルトでは、結果ウィンドウの形式とリンクを全てそのまま保存できる Stata Markup and Control Language (SMCL) という形式を使用します。SMCL ファイルはビューワで開くと結果ウィンドウの表示と同じように表示します。通常のテキストファイルとしてログが必要な時は、テキスト形式のログファイル（.log）として保存します。SMCL ファイルはファイル > ログ > 変換... メニューで様々なアプリケーションに対応した形式に変換できるので、SMCL 形式で保存することをお勧めします。（詳しくは [R] **translate** をご覧ください。）

ログファイルを開始するにはログボタン  をクリックします。その後ドロップダウンメニューで開始... を選択すると、新たなログファイルを作成します。追加... を選択すると、既存のログファイルにログを追加します。どちらを選択した場合も、標準のファイルダイアログが開き、ログファイルのファイル名やフォルダの指定ができます。ファイルの拡張子を指定しない時は、拡張子.smcl がファイル名に追加されます。

短いセッションの例は次の通りです。

```
      name:  <unnamed>
       log:  /Users/Stata/Documents/Stata/base.smcl
  log type:  smcl
 opened on:  29 Mar 2023, 09:11:19
. sysuse auto
(1978 automobile data)
. by foreign, sort: summarize price mpg
-------------------------------------------------------------------------------
-> foreign = Domestic
    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
       price |         52    6072.423    3097.104       3291      15906
         mpg |         52    19.82692    4.743297         12         34
-------------------------------------------------------------------------------
-> foreign = Foreign
    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
       price |         22    6384.682    2621.915       3748      12990
         mpg |         22    24.77273    6.611187         14         41
. * be sure to include the above stats in report!
. * now for something completely different
. corr price mpg
(obs=74)
             |    price      mpg
-------------+------------------
       price |   1.0000
         mpg |  -0.4686   1.0000
. log close
      name:  <unnamed>
       log:  /Users/Stata/Documents/Stata/base.smcl
  log type:  smcl
 closed on:  29 Mar 2023, 09:11:19
-------------------------------------------------------------------------------
```

上記の例の中から何点か確認します。

- ファイルの保存場所、種類、開始時間のスタンプを含むヘッダもログファイルの一部です。ヘッダは複数のログファイルを使用する時に便利です。
- アスタリスク (*) から始まっている 2 行はコメントです。Stata はアスタリスクの後ろにあるテキストはコマンドとして認識せず、無視します。つまり、特殊記号などを使いながら好きなコメントを入力できます。コメントはその時の考えを残すのにも適しており、またログファイルをセクション分けすることにも利用できます。
- この例ではログファイルを log close コマンドで閉じましたが、Stata を閉じればログファイルも自動的に閉じるので、手動で閉じることは必須ではありません。

それぞれのファイルに名前がある時は、複数のログファイルを同時に開くことができます。詳細は help log コマンドで確認してください。

16.3 ログで作業する

ログファイルは Stata のビューワを使います。ビューワを開くには、以下の 2 通りの方法があります。

- ログボタンをクリックし、表示... を選ぶ。
- ファイル > ログ > 表示... を選ぶ。

ログが開いている時は（ログステータスバーでオン・オフの確認ができます）デフォルトでそのログを表示します。それ以外のファイルを開く時はログファイルの名前を入力するか、参照... ボタンからファイルのダイアログを開き、リスト内から選んでください。

ビューワウィンドウでは、普通のビューワのファイルと同じようにテキストをコピーして他のビューワやワープロソフトにコピー・貼り付けを行えます。また、コマンドウィンドウや do ファイルエディタにも貼り付け可能です。その際の注意点として、結果ではなくコマンドだけをコピーするようにしてください。各コマンド文の冒頭にあるプロンプト (".") はコマンドウィンドウが無視するのでコピーしても問題ありません。ワープロソフトで作業を行う時に貼り付けたテキストは、書式設定のないテキストになります。貼り付けの時は固定幅フォント、例えば Monaco を使用すると見やすくなります。

自分の考えや分析の流れを見失わないように、現在のログファイルを時折確認するのも大切になります。ビューワウィンドウではログファイルのスナップショットも取れるので、作業をしながらスクロールする必要はありません。最新の結果をビューワに反映させたい時は、再読み込みボタンをクリックします。

ログに関する詳細は [U] **15 Saving and printing output—log files** と [R] **log** をご覧ください。ビューワに関する詳細は [GSM] **3 Using the Viewer (**ビューワを使う**)** をご覧ください。

16.4 ログを印刷する

標準的な SMCL のログファイルを印刷する時は、最初にビューワウィンドウで目的のファイルを開きます。そして、ビューワウィンドウを右クリックする、ビューワ内で右クリックして印刷... を選択するか、メニューからファイル > 印刷 > 印刷... を選ぶ、のいずれかで印刷します。詳細を表示ボタンをクリックして、印刷ダイアログを表示します。

- ヘッダー、ユーザ、プロジェクトの欄は必要に応じて入力します。ヘッダーとフッターを印刷する、行番号を印刷する、ロゴを印刷するのオプションは必要な時にそれぞれのチェックボックスにチェックを付けてください。これらの設定は保存され、次回以降、ログの印刷を行う時に適用されます。
- 印刷設定ダイアログ内の **Stata** のヘッダーとフッターをクリックすると、プルダウンメニューが表れるので、そこで **Stata** のフォントと色を選択すると、フォントサイズ、配色スキームに関する設定を変更できます。白黒は白黒印刷設定、カラーはデフォルトのカラー印刷設定、カスタム **1** とカスタム **2** はカラー印刷のカスタム設定です。

PostScript ファイルや PDF 形式のログファイルは translate コマンドで作成します。詳細は [R] **translate** をご覧ください。

テキストファイルとしてログを作成した時（拡張子が.smcl ではなく.log）、ファイルはテキスト編集用アプリケーション、例えばテキストエディットや Stata の do ファイルエディタ、あるいはワープロソフトで開くことができます。開いたログファイルは編集（見出しやコメントの追加など）し、その上で印刷もできます。ワープロソフトでログファイルを開くと、デフォルトのフォントで表示および印刷します。ログファイルは非固定幅フォント（例えば Times や Helvetica）では配置が崩れ読みにくくなります。固定幅フォント（Monaco や Courier New など）ならば配置が崩れることもないので、可能な限り固定幅フォントを使用してください。

16.5 do ファイルとしてコマンドを再実行する

Stata は出力結果を除き、コマンドだけをログとして記録できます。これは do ファイルをインタラクティブに使用する際に適しており、このようなファイルは **cmdlog** ファイルと呼ばれています。
cmdlog ファイルを開始するにはコマンドウィンドウに次のように打ち込みます。

```
cmdlog using ファイル名
```

cmdlog ファイルを閉じるには次のように入力します。

```
cmdlog close
```

先程のセッションの内容を cmdlog で保存すると次のようになります。コマンドのみで構成されるので、このまま do ファイルとして使用できます。

```
sysuse auto
by foreign, sort: summarize price mpg
* be sure to include the above stats in report!
* now for something completely different
corr price mpg
```

cmdlog ファイルを作らずに作業開始した後、cmdlog ファイルが必要になった時は履歴ウィンドウのコマンド一覧を do ファイルとして保存すれば問題ありません。履歴ウィンドウは実行した直近 5000 個のコマンドを記録しています。履歴ウィンドウで右クリックを行い、メニューからすべて保存**...** を選んでください。この方法は、まずエラーが出力されたコマンドをフィルタにかけて取り除いてから実行してください。詳細については [GSM] **2 The Stata user interface (Stata** ユーザインターフェイス**)** 内の「履歴ウィンドウ」をご覧ください。do ファイルエディタに直接コマンドを送りたい時は右クリックで表示するコンテキストメニューから選択範囲を **do** ファイルエディタへ送るを選びます。この方法はセッション中に入力したコマンドのみでテキストファイルを作成する時に便利です。

詳しくは [GSM] **13 Using the Do-file Editor—automating Stata (do** ファイルエディタを使用する**—Stata** の自動化**)**、 [U] **16 Do-files**、 [U] **15 Saving and printing output—log files** をご覧ください。

17 ウィンドウやフォントの設定をする

17.1 ウィンドウの位置、大きさ、フォントを変更して保存する

モニターの画質や好み等により、Stata のウィンドウの表示形式やフォントを変更したい場合もあります。また、グラフなどを論文で利用する際にはフォントの大きさ等について規則があるかもしれません。Stata では上記 2 つのような状況に対応するため、ウィンドウの表示設定を変更できます。

まず、各ウィンドウで設定可能なプロパティを確認した後、初期設定の管理方法について紹介します。

17.2 Graph ウィンドウ

Graph ウィンドウ上で右クリックを行い、コンテキストメニューからユーザ設定**...** を選ぶと、Graph ウィンドウの設定を変更できます。表示されるダイアログでグラフの表示形式を変更できます。ウィンドウタブでは印刷時の設定を、クリップボードタブではクリップボードの設定を変更できます。

グラフの初期設定はグラフの見せ方を複数のスキーム（scheme）に分けてコントロールします。スキームを使い分けることによって、目的のグラフを簡単に作成できます。つまり、The Economist や Stata Journal 向けのスキームもあり、これらの出版物の規定で簡単にグラフを作成できます。スキームを変更しても現在のグラフには反映しません。設定したスキームは次に作成するグラフから適用されます

17.3 そのほかのウィンドウ

ディスプレイのフォントとフォントサイズの変更は Stata のほぼ全てのウィンドウで行えます。

ウィンドウのフォントとフォントサイズを変更できる時は、コンテキストメニューにユーザ設定**...** が表示されます。ユーザ設定**...** を選択するとウィンドウダイアログが表示され、フォントの種類とサイズを選ぶことができます。結果、ビューワ、do ファイルエディタの各ウィンドウでは固定幅フォントのみ選択できます。固定幅フォントは出力結果の数字と文字を整列させ、読みやすくするために使用します。上記以外のウィンドウでは特に制限はなく、どのフォントでも使用できます。

17.4 配色を変更する

フォントだけでなく、背景や前景のテキストの色を変更することもできます。構文のハイライトカラーを選択すると、Stata のコマンドをその他のテキストと異なる色で表示できます。この設定は、

do ファイルエディタとコマンドウィンドウの両方に適用されます。macOS 10.14.2 以降ではダークモードをサポートしています。

結果とビューワウィンドウでは入力、テキスト、結果、エラー、リンク、選択テキストの表示色を選ぶことができます。それぞれ同じように設定されているので、各ウィンドウで右クリックを行い、それぞれ配色を選ぶか自分で配色を作成します。結果とビューワウィンドウのデフォルト設定はビルトインの“標準配色”になり、これは白い背景に黒いテキストを使用します。他にビルトイン配色とカスタム配色を設定できます。ビューワの設定変更を行うと一度に全てのビューワウィンドウを変更します。

17.5 複数の設定セットを管理する

Stata の設定は Stata を閉じた時に自動的に保存され、次回 Stata を開く時にリロードされます。しかし、時には Stata のウィンドウを並べ替えた後、以前に設定した別の並びに戻したいこともあるでしょう。以前の並びに戻すには変更した設定に名前を付けて保存します。そして必要に応じて、保存した設定をロードします。設定のロード後に変更を加えても、上書き保存をしない限りこの設定が書き換えられる心配はありません。

メニューから **Stata** > ユーザ設定 > ユーザ設定の管理**...** を開き、設定を管理します。各項目の詳細は次の通りです。

- ユーザ設定の管理**...** 保存された設定ウィンドウを表示します。ここでは、保存された設定の読み込み、変更、削除が行えます。
- ユーザ設定の保存**...** 現在の設定を設定セットに保存します。保存した設定は **Stata** > ユーザ設定 > ユーザ設定の管理**...** に表示されます。
- 工場出荷時設定 現在の設定セットを工場出荷時の状態に戻します。
- 工場出荷時のウィンドウ設定 現在のウィンドウに関する設定セットを工場出荷時の状態に戻します。

17.6 ウィンドウを閉じたり開いたりする

結果とコマンドウィンドウ以外のウィンドウは閉じることができます。現在閉じているウィンドウを開く時はウィンドウメニューから開きたいウィンドウを選んでください。

18 Stata について詳しく学ぶ

18.1 Stata を学ぶ上での次のステップ

今までの学習で Stata を使用するのに十分な情報をカバーしました。しかし、本マニュアルで説明した内容は基礎的な操作方法に関することなので、統計の分析とデータ管理分野について学習したい方は次のように操作してください。

- 興味深いデータセットを入手して Stata を使いながらいろいろ試してみてください。
  1. (a) まず、メニューやダイアログを使用して操作を行います。次に、結果ウィンドウにどのようなコマンドが出力されるのか確認します。Stata はシンプルで分かりやすいコマンド構文を使っているので、コマンドの理解は比較的容易にできます。また、コマンドは簡単に覚えることができるので、コマンド入力を積極的に利用しましょう。結果として、速く作業できるようになります。
     (b) グラフについてはグラフエディタでいろいろ試してみてください。
- コマンドウィンドウを使い始めると、短時間でより多くの操作を行えるようになります。同時に、新たな問題として、理由の分からないエラーが表示され、戸惑うこともあるでしょう。このような時は、次のように対応してください。
  1. (a)「help コマンド名」を実行するか、メニューからヘルプ **> Stata** のコマンド**...** と選択し、コマンド名を入力してください。
     (b) ヘルプファイル内にあるコマンド構文例を確認し、コマンドウィンドウに入力した物と比較してください。Stata は小さな誤字・脱字でもコマンドを実行できなくなるので、細部まで比べる必要があります。
- メニューでヘルプ > 検索**...** と選択して Stata 内を検索してください。ヘルプファイル内の多くの統計的ルーチンが載っているので、役に立つでしょう。
- Stata Index 内の Combined subject table of contents に一通り目を通し、わからない箇所を詳しく読みます。
- Stata を操作しながら User's Guide を読んでください。User's Guide は最初から最後まで通して読むように作成してあり、Stata のエキスパートになるために必要な情報をほぼ全て含んでいます。一読の価値は十分にあるでしょう。それほど上級者向けの技術を身に着ける必要が無い人は、この章の後半にあるアドバイスを参考にしてください。
- 各リファレンスマニュアルに目を通してください。その際、目的の統計手法のことだけを読んでください。また、リンクをたどって他のトピックも必要に応じて参照できます。リファレンスマニュアルは最初から最後まで順に読むようには作られていません。むしろ、百科事典のように必

要な箇所だけを読むようになっています。マニュアルの中で使用するデータセットはメニューからファイル > 例題データセット... を選択して表示される、Stata18 manual datasets 項目にあります。データセットを活用して、例題にスムーズに取り組むことができます。

- Stata に関する情報は更にあり、よくある質問（FAQ）もその 1 つです。次の web アドレス（https://www.stata.com/support/faqs/）から確認できます。
- Stata の資料ページ（https://www.stata.com/links/）にはたくさんの役立つリンクがあります。このページには数多くの Stata に関するすばらしい資料のリンクがあります。
- Stata と統計に関する情報交換を行うリストサーバ、Statalist に参加するのもいいでしょう。
- The Stata Blog: Not Elsewhere Classified (https://blog.stata.com/) では Stata 社の社員が書いた記事を読み、Stata についての理解を深めることもできます。
- Stata の Facebook ページ（https://facebook.com/statacorp）を見たり、
- Instagram（https://www.instagram.com/statacorp/）で Stata に参加したり、
- LinkedIn（https://www.linkedin.com/company/statacorp）を確認したり、
- Twitter（https://twitter.com/stata）でフォローしたりすることで、最新情報を確認できます。
- 様々な分野で統計を使用する研究者に有益な情報を、論文、コラム、本のレビュー等の形で掲載する Stata Journal を定期購読するのもおすすめです。Stata Journal の定期購読申込みは https://www.stata-journal.com からどうぞ。
- Stata に関する副読本も提供しています。興味のある人は Stata Bookstore（https://www.stata.com/bookstore/ を尋ねてみてください。
- Stata NetCourse®, NetCourse 101 は Stata について学ぶのに最適なコースです。コースの情報と日程に関しては https://www.stata.com/netcourse/をご覧ください。
- Stata 社も Stata トレーニングを各地やウェブ上で開催しています。コースの情報と日程に関しては https://www.stata.com/training/classroom-and-web/をご覧ください。
- Stata の動画を見るには、https://www.youtube.com/user/statacorp にアクセスしてください。

18.2User’s Guide およびリファレンスマニュアルのおすすめ項目

User’s Guide は最初から最後まで通して読むことを前提として作られています。逆にリファレンスマニュアルは必要な時に、必要な項目だけを参照します。

本マニュアルを読み終えてから User’s Guide も最初から最後まで読んでいただくのが理想です。しかし Stata をすぐにある程度使えるようになりたいという場合は、次に示す User’s Guide とリファレンスマニュアルの項目を先に参照してください。この一覧では基本機能を紹介する項目と、見落としがちですが便利な機能について紹介します。

Stata の基礎要素

[U] **11 Language syntax**

[U] **12 Data**

[U] **13 Functions and expressions**

データ管理

[U] **6 Managing memory**

[U] **22 Entering and importing data**

[D] **import** — Stata へのデータのインポートに関する概要

[D] **append** — データセットのアペンド

[D] **merge** — データセットのマージ

[D] **compress** — メモリ内のデータの圧縮

[D] **frames Intro** — フレームに関する概要

グラフィックス

[D] Stata Graphics Reference Manual

再現可能な調査

[U] **16 Dofiles**

[U] **17 Ado-files**

[U] **13.5 Accessing coefficients and standard errors**

[U] **13.6 Accessing results from Stata commands**

[RPT] **Dynamic documentations Intro** — ダイナミックなドキュメントに関する概要

[RPT] **putdocx Intro** — Office Open XML (.docx) ファイルの作成機能に

[RPT] **putexcel** — 分析結果の Excel ファイルへのエクスポート

[RPT] **putpdf Intro** — PDF ファイルの作成機能に関する概要

[R] **log** — セッション履歴のファイルへのコピーする

見落とす可能性が高い便利な機能

[U] **29 Using the Internet to keep up to date**

[U] **19 Immediate commands**

[U] **24 Working with strings**

[U] **25 Working with dates and times**

[U] **26 Working with categorical data and factor variables**

[U] **27 Overview of Stata estimation commands**

[U] **20 Estimation and postestimation commands**

[R] **estimates** — 推定結果の保存・管理

基本的な統計

[R] **anova** — 分散分析と共分散分析

[R] **ci** — 平均/比率/カウントの信頼区間

[R] **correlate** — 変数の相関

[D] **egen** — generate の拡張機能

[R] **regress** — 線形回帰

[R] **predict** — 推定後の予測値、残差等の取得

[R] **regress postestimation** — regress 推定後のツール

[R] **test** — 推定後の線形仮説検定

[R] **summarize** — 頻度、要約、コマンド結果の表

[R] **table** — 高度な統計表

[R] **tabulate oneway** — 一元配置表

[R] **tabulate twoway** — 二元配置表

[R] **ttest** — *t* 検定（平均比較検定）

Matrices

[U] **14 Matrix expressions**

[U] **18.5 Scalars and matrices**

[M] Mata Reference Manual

プログラミング

[U] **16 Dofiles**

[U] **17 Adofiles**

[U] **18 Programming Stata**

[R] **ml** — 最尤法

[P] Stata Programming Reference Manual

[M] Mata Reference Manual

システム値

[R] **set** — システム値の概要

[P] **creturn** — c-class 値の取得

インターネットの情報

Stata の Web サイト（https://www.stata.com）では、Stata の追加情報を得ることができます。 Web サイトには FAQ の確認、他のユーザとの交流の場の提供、Stata の公式アップデート情報など、役立つ情報を掲載しています。また、Stata と統計に関する情報交換のためのリストサーバである、

Statalist にも申し込めます。

ここではインターネット上で提供されているインタラクティブなコースである、Stata NetCourses®に関する情報も入手できます。このコースの長さも数週間から８週間までの幅がありますので、ご都合に合わせてお選びください。さらに Stata はウェブベースのトレーニングやウェビナーも行っています。詳しくは https://www.stata.com/learn/をご覧ください。

ウェブサイト内には Stata ユーザが興味を持ちそうな本を取り揃えているページ、Stata Bookstore もあります。それぞれの本には技術スタッフからユーザが興味を持ちそうな理由が簡単なコメントとして付いています。

一度 Stata のウェブサイトを確認してみてください。Stata をオンライン上で登録し、Stata News の無料購読のお申込みもできます。

本やマニュアル等、Stata Press の出版物の情報は https://www.stata-press.com をご覧ください。マニュアル内で使用する例題のデータセットは Stata Press のウェブサイトにあります。

Stata Journal は年４回発行し、統計、データ分析、教授方法、Stata 言語の有効な活用法などについての記事を掲載します。ウェブサイト https://www.stata-journal.com をご確認ください。

Stata の公式ブログ（https://blog.stata.com）には Stata のニュースや使用法に関するアドバイスの記事を掲載しています。ブログに投稿する記事はそれぞれ記名式になっており、Stata の開発・サポート・販売を実際に行う Stata 社の社員が書いています。また、Stata Blog: Not Elsewhere Classified には世界中の Stata ユーザが作成した Stata に関するブログへのリンクもあります。

Stata の Facebook（https://facebook.com/statacorp）、Twitter（https://twitter.com/stata）、Instagram（https://www.instagram.com/statacorp/）、LinkedIn（https://www.linkedin.com/company/statacorp）といった情報源を活用してみてください。これらは最新の Stata に関する情報が確認できます。さらに、Twitter には、フランス語版：https://twitter.com/stata fr とスペイン語版： https://twitter.com/stata es もあります。Stata の使用法に関する例題を集めた短い動画は YouTube（https://www.youtube.com/user/statacorp）で見ることができます。

Stata の公式アップデートにアクセスする方法や Stata の無料追加の詳細などは **[GSM] 19 Updating and extending Stata—Internet functionality (Stata** のアップデートと拡張**—**インターネットでの機能**)** をご覧ください。

19 Stata のアップデートと拡張―インターネットでの機能

19.1 Stata のインターネットでの機能性

Stata はインターネットとうまく動作できるようになっています。自分のコンピュータに保存してあるファイルを開くように、インターネット上のデータセットやヘルプファイルを表示できます。また、ユーザが許可すれば、インターネットを通して自動更新も行えます。さらに、Stata の機能を拡張するユーザ作成コマンドのインストールもできます。これらのコマンドは Stata Journal (SJ)で発表したものか、ユーザが作成して Stata のコミュニティーに共有したものです。

この章では Stata の可能性を広げる方法を紹介していきます。

19.2 インターネットからのファイルを使用する

Stata は URL をローカルファイルのパスと同じように認識します。ウェブ上のデータセット、グラフ、do ファイルを使用する時は、Stata で簡単に開く事ができます。1 つ例題を紹介しましょう。

https://www.stata-press.com/data/には多くのデータセットがあります。例えば、 **[U] 11 Language syntax** 内で使用するデータセット census12 は、https://www.stata-press.com/data/r18/census12.dta にあります。ではデータセットを開いてみましょう。データセットを開くには use で始まる、次のようなコマンドを使用します。

```
. use https://www.stata-press.com/data/r18/census12.dta
(1980 Census data by state)
. describe
Contains data from https://www.stata-press.com/data/r18/census12.dta
 Observations:            50                  1980 Census data by state
    Variables:             7                  6 Apr 2022 15:43
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
state           str14   %14s                  State
state2          str2    %-2s                  Two-letter state abbreviation
region          str7    %9s                   Census region
pop             long    %10.0g                Population
median_age      float   %9.2f                 Median age
marriage_rate   float   %9.0g
divorce_rate    float   %9.0g
-------------------------------------------------------------------------------
Sorted by:
```

URL をローカルファイルと同じように認識する機能は Stata の至る所で見られます。構文内で「ファイル名」を使用するものは、そこに URL を入力することができます。

この機能の例題として HTTP というプロトコルを使用してファイルを取り出す機能があります。Stata は HTTP, HTTPS, FTP の各プロトコルを理解できます。

19.3 Stata のアップデート

Stata の追加情報は大きく分けて 2 つに分類できます。1 つは Stata 社が提供するアップデートをはじめとした情報、もう 1 つは世界各国の Stata ユーザが作成したユーザ作成プログラムです。ユーザ作成プログラムは主に SJ で出版され、Statalist 等を通して提供されるものを指します。

Stata が提供しているアップデート（以後、公式アップデートと呼びます）とユーザ作成プログラムはインターネットを通して入手します。まずは、公式アップデートから紹介しましょう。Stata 社は公式 Stata のアップデートファイルを配信します。アップデートファイルは新規機能の追加やバグ修正を行うものがあります。

Stata はデフォルトで自動アップデートチェックがオンになっており、7 日ごとにアップデートのチェックを行います。設定の変更や確認にはメニューから編集 > ユーザ設定 > 一般的なユーザ設定... を選択し、インターネットタブをクリックします。

自動かつ簡単にバックグラウンドで Stata を最新版にできるので、自動アップデートチェックはおすすめです。デフォルトの状態でしばらくアップデートの有無をチェックしていない時は、Stata 起動時にポップアップを表示して警告します。

手動でアップデートの有無を確認するには、メニューからヘルプ > アップデートのチェックを選択するか、コマンドウィンドウで「update query」を実行します。どちらの方法でも、同じように確認できます。チェックが終了すると、アップデートのステータスを結果ウィンドウに表示します。最新版に更新済みの場合は、結果ウィンドウにその内容を表示します。アップデートが必要な時は、結果ウィンドウにその内容と Install available updates というリンクを表示します。このリンクをクリックするか「update all」コマンドを実行するか、どちらの場合でも、最新版にするために必要なものをダウンロードできます。アップデート終了後に一度 Stata は再起動するので、現在のセッションで続けたい作業（例えば、コマンド履歴の保存など）がある時は、アップデートを先送りにすることもできます。

トラブルシューティングメモ：/Applications/Stata への書き込み許可がない場合、この方法で公式アップデートをインストールすることはできません。公式アップデートのダウンロードは行えますが、 update コマンドを使用する必要があります。詳しくは **[U] 29 Using the Internet to keep up to date** をご覧ください。

19.4 自動アップデートチェック

Stata は自動で定期的にアップデートチェックを行います。デフォルトでは、7 日に一度の頻度で、Stata 社のウェブサイトでアップデートの有無を確認します。間隔は自動手動を問わず、最後に「update query」を実行した時から 7 日です。チェック間隔の変更も行えます。

Stata のアップデートチェックは、今すぐにチェックを行う、次回 Stata を起動する時に行う、次の更新確認まで先送りにする、という 3 つの選択肢を、インターネットに接続する前に尋ねます。このポップアップを無効にし、確認を求めずにアップデートチェックを行う設定もあります。
アップデートが可能な時、Stata はメッセージを表示します。その後は Stata の指示に従ってアップデートしましょう。

自動アップデートの設定変更は、メニューから **Stata** > ユーザ設定 > 一般的なユーザ設定**...** と選択し、インターネットタブで行います。

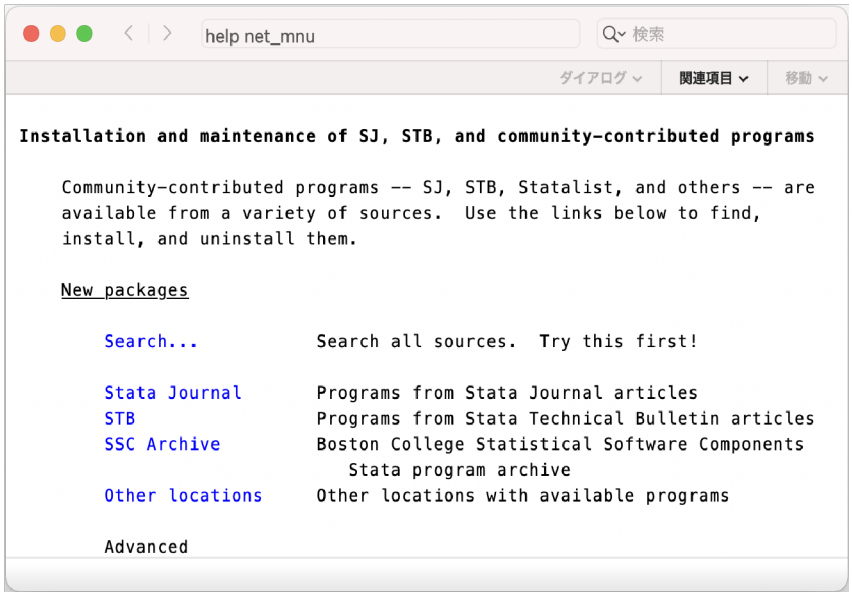
19.5 ユーザ作成のプログラムをキーワードで探す

Stata にはインターネット上のユーザ作成プログラムを検索するビルトインユーティリティがあります。アクセスするにはメニューからヘルプ > 検索**...** を選択し、インターネットリソースの検索を選んで、キーワードを入力します。また、ヘルプ > **Stata** ジャーナル/ユーザ作成コマンドを選ぶと、更に細分化した検索用の選択肢を表示します。このユーティリティでは SJ プログラムを含む、インターネット上のすべてのユーザ作成プログラムを検索します。ビューワに表示された項目をクリックすると、その項目のヘルプファイルに移動できます。

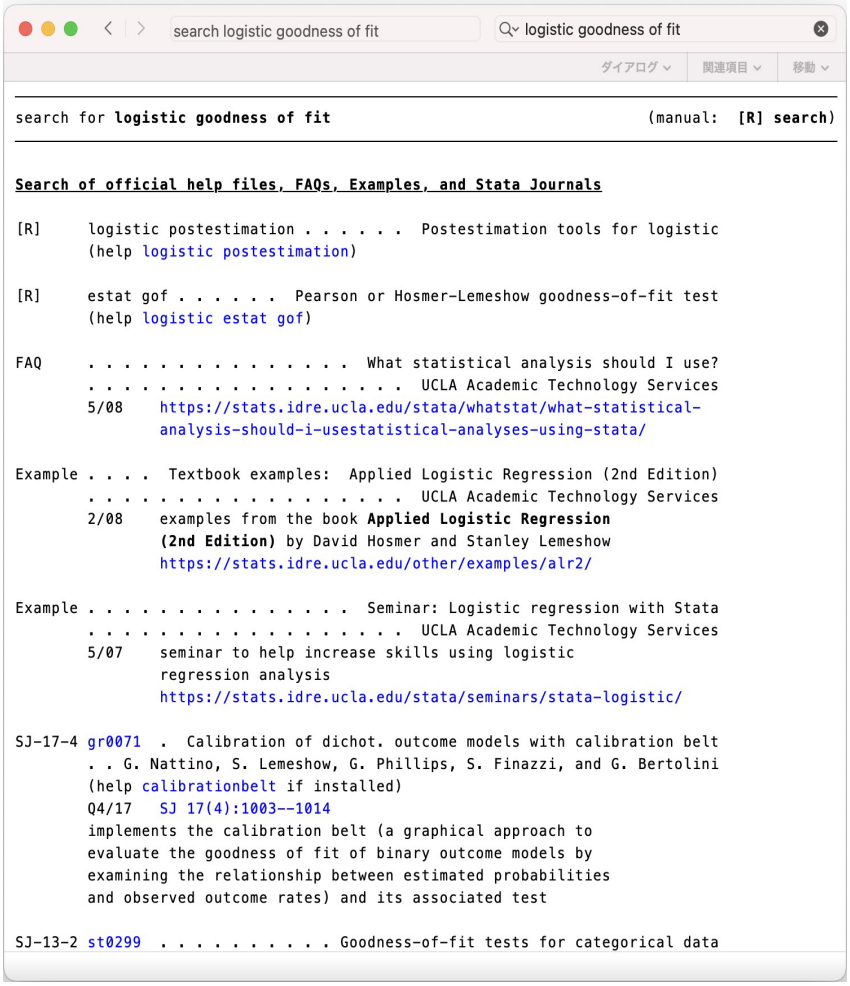
他の検索用コマンド「search キーワード, net」の構文については [R] **search** をご覧ください。

19.6 ユーザ作成のプログラムをダウンロードする

ユーザ作成プログラムは簡単にダウンロードできます。メニューからヘルプ > **Stata** ジャーナル/ ユーザ作成コマンドと選択するところから始めます。

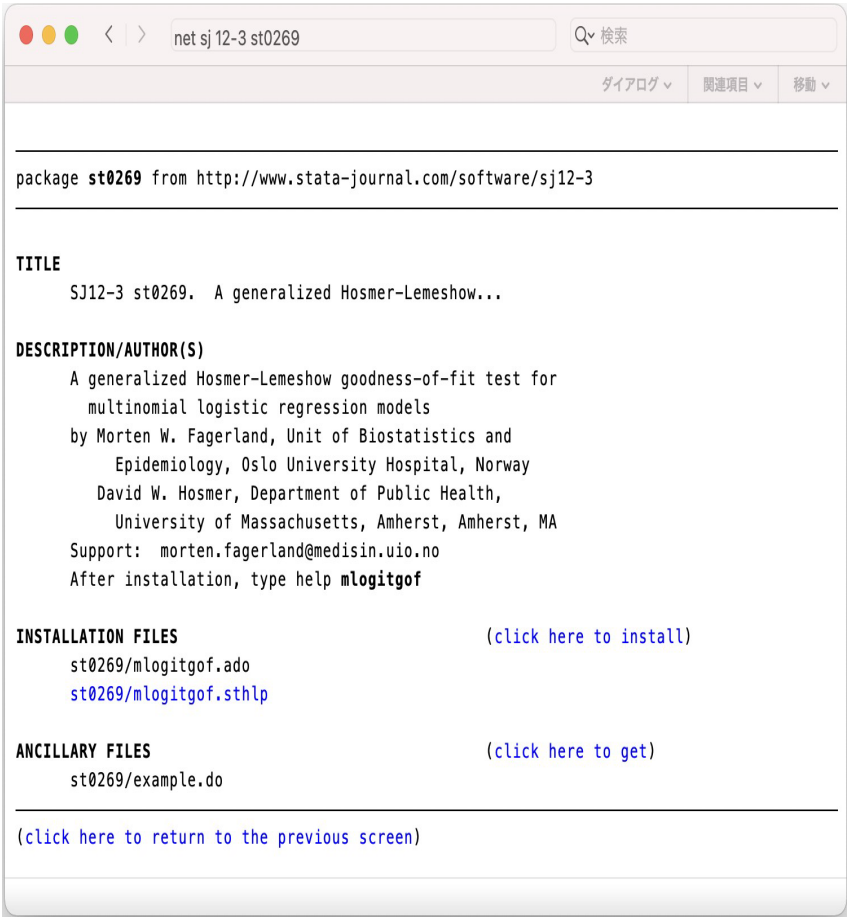


上記ビューワが示すように、まずは検索**...** を行います。ロジスティック回帰の適合度に関する情報かユーザ作成プログラムを探していると仮定します。メニューでヘルプ > 検索**...** と選んだ後に全てを検索を選択します。検索ボックスに「logistic goodness of fit」と入力して Return を押します。



1 番目の項目はロジスティック回帰実行後に使用できる推定後コマンドです。2 番目ロジスティック回帰後に適合度を計算する、Stata のビルトイン機能、estat gof コマンドです。リンクをクリックして内容を確認してみましょう。次のリンクは UCLA（カリフォルニア州立大学ロサンゼルス校）のウェブサイトにある FAQ に繋がります。あとの 3 つのリンクは SJ に繋がります。多項ロジスティック回帰に興味があれば、残りのリンク確認すると良いでしょう。SJ の 12 巻 3 号に記事があります。st0269 をクリックすると、コマンドとそれに関連する記事が表示されます。

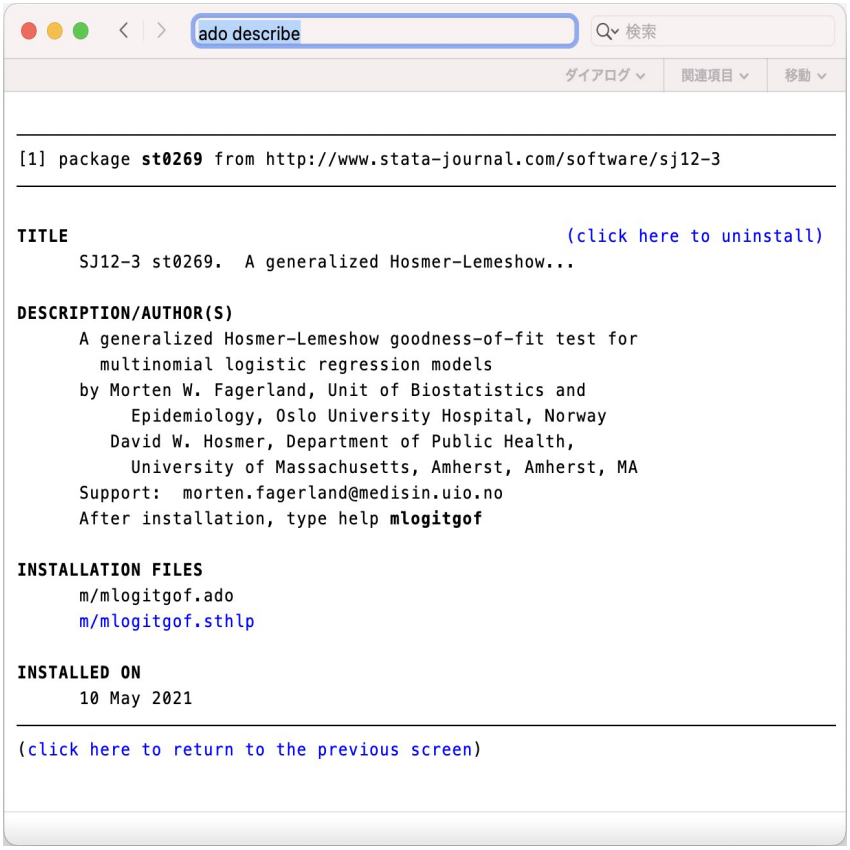
19.6 ユーザ作成のプログラムをダウンロードする



このパッケージには 3 つの新しいコマンドに対して、3 つのヘルプファイルがあることがわかります。gr0021 2/polarsm.sthlp リンクをクリックし、polarsm コマンドがどのようなものか確認してください。このコマンドを実際にインストールして試すには、一度戻るボタンをクリックしてから click here to install リンクをクリックします。付属の補助ファイル（コマンドの仕組みについて説明を行うものが一般的）も使用するには、click here to get のリンクをクリックするとダウンロードできます。このようにファイルをダウンロードしても Stata の動作に影響はありません。これらのユーザ作成コマンドを安全にアンインストールする方法はこの後すぐに紹介します。

ユーザ作成コマンドを最新の状態に更新するには adoupdate コマンドを使用します。「ado update」はアップデートの有無の確認を、「ado update, update」はアップデートの有無の確認とインストールを行います。このパッケージを削除しましょう。アンインストールはとても簡単で、メニューからヘルプ > **Stata** ジャーナル/ユーザ作成コマンドと選択し、List リンクをクリックします。すると、次のような画面を表示します。

この画面でプログラムのオンライン解説を選択すると、インストールしてあるプログラムの解説を表示します。この画像は一番下までスクロールすると確認できるダイアログです。もちろん、実際の画面とこの画像はインストール日付が異なります。



削除（アンインストール）するにはパッケージの説明画面で click here to uninstall をクリックし、表示されるダイアログで **OK** をクリックします。

net コマンド使用してユーザ作成プログラムのダウンロード行う詳細については [R] **net** をご覧ください。

A Stata のトラブルシューティング

A.1 Stata が起動しなかったら

Stata を起動しようとして失敗する時、Stata または OS がエラーメッセージを表示します。エラーメッセージが表示された時の対処法はメッセージの種類により異なります。次の対処法を参考にしてください。

Cannot find license file

このメッセージは「ライセンスファイルが見つからない」ことを示します。このエラーは深刻なものではなく、Stata は目的のライセンスファイルを発見できなかっただけです。このエラーが表示される一番の理由は、インストール手順が完了していないためです。

DVD と同梱してある、ライセンス用紙のコードをインストール時に入力してロックを解除しましたか。未解除の時は一度 Stata に戻り、初期化手順を完了してください。

Error opening or reading the file

技術的な理由で何か明確なエラーが発生したことを示します。Stata は目的のファイルを発見しましたが、そこで I/O エラーが発生しています。

このエラーが発生するほぼ唯一の原因はハードディスクのエラーです。その場合、Stata のテクニカルサポートまでご連絡ください。詳しくは [U] **3.8 Technical support** をご覧ください。

License not applicable

有効なライセンスはありますが、インストールした Stata の種類には適用できないと判断されました。

例えば、Stata/BE 用のライセンスを所有するものの、誤って Stata/SE や Stata/MP をインストールしようとしたり、Stata/SE 用のライセンスを所持しつつ、Stata/MP をインストールしたりした場合です。このように、お持ちのライセンスとインストールの種類が一致していない時は、正しい種類の
Stata をインストールし直してください。

その他のメッセージ

上記以外のメッセージは Stata がライセンスでは許可していない行為を行っていると判断しています。例えば、ネットワークライセンスをお持ちでは無いのに Stata をネットワーク上で起動しようとすることです。しかし、考えられる理由はこれだけではなく、他にも多くの可能

性があります。このような場合、次の 2 つのどちらかに分類されます。1 つは本当にお手元のライセンスでは認められていない行為を行っていることで、もう 1 つはインストールした Stata に問題がある場合です。どちらの場合でも、一度お問い合わせください。ライセンスはアップグレードも可能ですし、Stata 自身に問題がある場合は代わりの DVD をご提供できます。詳しくは [U] **3.8 Technical support** をご覧ください。

A.2 トラブルシューティングのヒント

Stata Mac 版は、動作環境に macOS Sierra 10.13 (High Sierra) またはそれ以降のバージョンを要求します。

Stata の起動途中にクラッシュが発生した場合は、ホームディレクトリの Library ディレクトリにある Preferences ディレクトリを開いてください。そこで com.stata.stata18.plist というファイルをごみ箱に移動した後、再び Stata を起動してください。この設定ファイルが、電源異常またはコンピュータの正常なシャットダウンの未実施などによるクラッシュで破損することが稀にあります。

また、Stata の Web サイトにある Mac 版・よくある質問（FAQ）（https://www.stata.com/support/faqs/mac）も合わせてご覧ください。ウェブページに問題の解決策が掲載されていることがあります。

サポートチームへお問い合わせいただくことも可能です。その際は可能な限り多くの情報を提供していただいております。一度コンピュータを再起動して Stata を改めて開始し、問題が再び起こるまでの作業手順を書き残しながら現象を再現してみてください。その手順の情報が必要になります。

更にコンピュータに関する、できるだけ多くの情報を提供してください。Mac のモデル、OS のバージョンが必要です。**Apple** メニュー > この **Mac** についてを選択し、システムレポート**...** ボタンをクリックして、システム情報アプリケーションを起動します。システム情報をファイルに保存し、サポートチーム宛の E メールに添付できる形にまとめられます。最後に、Stata のシリアル番号とリリース日（revision date）をお調べください。E メールで連絡する時はこれらの情報を分かる範囲で教えてください。Stata の情報は Stata のコマンドウィンドウで「about」と実行すると表示されます。about コマンドではバージョンやリリース日を含む、使用中の Stata に関する全ての情報を表示します。

B 上級者向け Stata の使用法

B.1 Stata が起動する時に毎回コマンドを実行する

Stata は起動する時に自動的に profile.do ファイルを検索します。目的のファイルが存在する場合、その中にあるコマンドを実行します。この時、まずはインストールしてあるフォルダ内を探します。それから、現在の作業フォルダ、パスの示すフォルダ (ターミナル・シェルスクリプトから呼び出した場合)~/Documents/Stata、ado-path ([P] **sysdir** 参照) の順に検索します。profile.do ファイルを、~/Documents/*Stata*/フォルダ内に入れることをおすすめします。

例えば、Stata を起動する度に日付の入ったログを取り始めるようにします。~/Documents/Stata フォルダ内に次のあまり見慣れない形式のコマンドを入力して profile.do を作成します。

```
log using `: display %tCCCYY-NN-DD-HH-MM-SS ///
    Clock("`c(current date)' `c(current time)'","DMYhms")', ///
    name(default log file)
```

Stata を起動すると通常の開始画面が開きます。同時に、次の追加コマンドを実行します。

```
running ~/Documents/Stata/profile.do ...
```

このコマンドはどのような意味なのでしょうか。コマンドを一つずつ確認していきましょう。

- c(current date)と c(current time)はローカルシステムのマクロで、現在の日付と時間を含むものです。詳細は [P] **creturn** をご覧ください。
- ローカルマクロの両端にある左 (`) と右 (') の引用符は、これらを拡大します。詳細は [P] **macro** をご覧ください。
- Clock()関数は結果として表示する文字列や日付マスクである"DMYhms"を使用して、Stata が認識できる日付時間番号を作成します。詳しくは [D] **Datetime** をご覧ください。
- 「%tCCCYY-NN-DD-HH-MM-SS」形式は Stata が算出した数字を「年-月-日-時-分-秒」の形式で表示するように指定します。これはファイルをきれいにソートする際に便利です。詳しくは [D] **Datetime display formats** をご覧ください。

- あまり見ることのない「: display …」という綴りは、形式を指定した日付を直接ファイル名として使用するように指定します。これはインライン拡張のマクロ関数を使用する、上級者向けコンセプトです。詳細は [P] **macro** を参照してください。
- 「log using」コマンドはログファイルを開始します。例題は [GSM] **18 Learning more about Stata (Stata についてもっと詳しく学ぶ)** にありますので、参照してください。
- name オプションはログファイルに内部用の「default log file」を定義し、他のログファイルと名前が被らないようにします。詳細は [R] **log** をご覧ください。
- 最後に、///記号は連続性コメントです。これらの 3 つの行は 1 つのコマンドであることを示しています。コメントに関しての詳細は [P] **comments** をご覧ください。

この 1 つのコマンドに Stata のプログラミングを行う上での上級コンセプトがたくさん含まれています。

profile.do は実行された後には他の do ファイルと同じように扱えます。結果としては Stata を起動後に「run profile.do」と実行しても同じです。ただし profile.do は、Stata 起動時に自動で実行されるという特徴があります。

システム管理者には sysprofile.do という便利な do ファイルもあります。このファイルは基本的に profile.do と同じですが、Stata は先に sysprofile.do の有無を確認します。存在する場合、Stata はファイル内のコマンドを実行します。その後に profile.do を探します。存在する場合、ファイルのコマンドを実行します。

sysprofile.do が便利さを発揮する例は、システム管理者が Stata システムディレクトリのパスを変更したい時です。このような場合、sysprofile.do に次のようなコマンドを組み込んで作成します。

```
sysdir set SITE "~/Documents/Stata"
```

do ファイルの説明については [U] **16 Dofiles** をご覧ください。do ファイルはシンプルなテキストファイルで、Stata が実行できるコマンドのシーケンスです。

B.2 Stata を開始する他の方法

.dta のデータセット、.do の do ファイル、.gph のグラフファイルのいずれかをダブルクリックすると、新しい Stata のインスタンスが開いて目的のファイルを表示します。この時、do ファイル以外に

ついては、ダブルクリックしたファイルの存在するフォルダがそのインスタンスの作業フォルダになります。

ダブルクリック後の動作は、一般的なアプリケーションと同様です。データセットファイルをダブルクリックすると、Stata が起動し、データセットを開きます。グラフファイルをダブルクリックすると、

Graph ウィンドウでグラフが開き、do ファイルをダブルクリックすると、do ファイルエディタで do ファイルが開くか、または、do ファイルにあるコマンドを実行します。

do ファイルのダブルクリックでコマンドが実行されるようにするには、**Stata** > ユーザ設定 > 一般的なユーザ設定**...** を選択し、ウィンドウツールバーボタンをクリックした後、左側の枠内で **do** ファイルエディタを選択します。その後、上級設定をクリックし、**do** ファイルエディタの **Finder** から開いた **do** ファイルを編集するのチェックを外します。

B.3 Stata のバッチモード

Stata をバッチモードで実行する場合、ターミナルから起動してください。ターミナルから起動する際の Stata のコマンドは、たとえば Stata/SE の場合、次の通りです。

StataSE [*-option* [*-option* [...]]][*stata command*]

-option には次の一覧にあるオプションを指定できます。

パラメータ	結果
-b	バックグラウンド（バッチ）モードを設定し、ログファイルをテキストで出力します
-e	バックグラウンド（バッチ）モードに切り替えた上で Stata コマンドが完了した時にその応答をポップアップ表示することなく、テキストであるログファイルに出力します
-q	ロゴと初期化メッセージを非表示にします
-rngstream#	乱数生成器を mt64s に設定（詳細は [R] **set rng** ）し、乱数ストリームを#に設定（詳細は [R] **set rngstream** ）します
-s	バックグラウンド（バッチ）モードを設定してログファイルを SMCL テキストで出力します

Stata をターミナルから起動するには、シェルが Stata が認識できる必要があります。そのためには、Stata アプリケーションのファイル群にある Stata の app ファイルへのパスをシェルのパスに追加しておく必要があります。その後、シェルからの Stata の呼び出しが可能になります。

たとえば、Stata が/Applications/Stata にインストールされている場合、Stata/SE へのパスは /Applications/Stata/StataSE.app/Contents/MacOS になります。Stata/SE を開始するには、StataSE と入力して実行します。

Stata/MP の場合、パスは/Applications/Stata/StataMP.app/Contents/MacOS となります。Stata/MP を開始するには、StataMP と入力して実行します。

Stata/BE へのパスは/Applications/Stata/Stata.app/Contents/MacOS となります。Stata/BE を開始するには、Stata と入力して実行します。

たとえば bigjob.do というファイル名の do ファイルをバッチモードで実行するには、次を入力して実行します。

```
% StataSE -b do bigjob
```

これにより、bigjob.do を実行し、出力はすべて画面表示されず、同じフォルダ内に作成される bigjob.log へ記録されます。全てのコマンドが終了すると、Stata がダイアログを表示します。

また、

```
% StataSE -e do bigjob
```

上記を実行すると、同様に bigjob.do を実行し、出力はすべて画面表示されず、同じフォルダ内に作成される bigjob.log へ記録されますが、全てのコマンドが終了した際に、Stata がダイアログを表示しません。

さらに、

```
% StataSE -s do bigjob
```

上記を実行すると、bigjob.do を実行し、出力はすべて画面表示されないところまでは同様ですが、同じフォルダ内に作成されるログファイルは bigjob.smcl になります。

上記の 3 つのコマンドは、次のように入力するとバックグラウンドで実行できます。

```
% StataSE -b do
bigjob & %
StataSE -e do
```

```
bigjob & %
StataSE -s do
bigjob &
```

メモ：通常のインタラクティブな Stata の立ち上げ時と同様に、bigjob.do を実行する直前には profile.do が実行されます。

全般的なメモ

do ファイルの実行中は、Stata アイコンが Dock に表示されます。

アイコンには、do ファイルの進行の割合を示すパーセンテージが付属します。（このパーセントは do ファイル内の文字数を元に計算します。つまり、表示されるパーセントは残り時間を正確に表しているとは限りません。）

バッチ作業をキャンセルすることもできます。キャンセルするには、タスクバーにあるアイコンを右クリックしバッチ処理の中止を選択します。

バッチ作業が完了すると、タスクバーのアイコンが跳ねて作業の完了を知らせます。アイコンをクリックすると Stata が閉じます。バッチ作業の完了後に Stata を自動的に閉じるにはパラメータ-b で

はなく-e を使用してください。

do ファイルは、必ずしもバッチモードで実行する必要はありません。do ファイルを Stata のメインウィンドウからインタラクティブに実行することもできます。Stata を起動し、「log using ファイル名」を実行した後、「do ファイル名」を実行します。その後、do ファイルが実行される様子の観察や、あるいは Stata の最小化のどちらも実施可能です。

B.4 Stata のロケールを変更する

Stata のロケールを英語に変更するには、次のようにします。

```
set locale ui en
```

お使いの OS と同じロケールに戻すには、次のようにします。

```
set locale ui default
```

ロケールと Stata の関係についての詳細は [U] **12.4.2.4 Locales in Unicode** をご覧ください。

B.5 More

結果ウィンドウの出力をポーズを置きながら表示するようにするには、「set more on」コマンドを実行します。

ウィンドウに表示しきれない結果があるとき、結果ウィンドウ左下に—more—というプロンプトが表示されます。たとえば、多くのデータをリストする時に表示されます。

```
. list make mpg
     ┌──────────────────────┐
     │ make                 mpg │
     ├──────────────────────┤
  1. │ Linc. Mark V          12 │
  2. │ Linc. Continental     12 │
  3. │ Cad. Eldorado         14 │
  4. │ Linc. Versailles      14 │
  5. │ Merc. Cougar          14 │
     ├──────────────────────┤
  6. │ Merc. XR-7            14 │
  7. │ Peugeot 604           14 │
  8. │ Cad. Deville          14 │
  9. │ Buick Electra         15 │
 10. │ Merc. Marquis         15 │
     ├──────────────────────┤
 11. │ Buick Riviera         16 │
 12. │ Olds Toronado         16 │
 13. │ Dodge Magnum          16 │
 14. │ Chev. Impala          16 │
 15. │ Volvo 260             17 │
     ├──────────────────────┤
 16. │ Dodge St. Regis       17 │
 17. │ AMC Pacer             17 │
 18. │ Audi 5000             17 │
 19. │ Toyota Corona         18 │
 20. │ Buick LeSabre         18 │
     └──────────────────────┘
—more
```

続きの画面を表示するには、3 つの方法があります。まず、キーボードのキー（例えばスペースキー）を押す方法です。次にメインウィンドウのツールバーにある **More** ボタン  をクリックする方法です。最後に、結果ウィンドウの左下にある—more—リンクをクリックする方法です。結果ウィンドウの出力結果で次の行を 1 行表示するには Return キーを 1 回押してください。

B.6 メモリサイズに関する考慮

Stata のメモリ管理は自動で行われます。少数の Stata ユーザが必要とする、メモリを効率化する小ワザに関しては [D] **memory** をご覧ください。

C Mac 版 Stata について―追加要素

C.1 Stata のデータセットやグラフを他のプラットフォームで使用する

Stata のデータセット（.dta）およびグラフファイル（.gph）は作成した OS が Windows または Unix でも、プラットフォームに依存することなく Mac 版で開くことができます。また、Windows 版 Stata と Unix 版 Stata のユーザでも Mac 版で作成したファイルを使用できます。。.dta と.gph ファイルはバイナリファイルです。FTP を利用して Stata のファイルを送信する時は、テキスト形式ではなくバイナリ形式を使用してください。

他のプラットフォームで作成したファイルは、コマンドウィンドウから直接開くことができます。すなわち、既に開いた Stata のコマンドウィンドウから use（ファイル名）と入力・実行してデータセットを読み込むか、または Finder 上のデータセットファイルをダブルクリックして新しい Stata で開くことができます。

C.2 Stata のグラフを他のドキュメントにエクスポートする

Stata で作成したグラフをワープロソフトやプレゼンテーションソフトで使用する場合もあります。グラフをエクスポートする方法は 2 通りあります。1 つはクリップボードを使用してグラフのコピーと貼り付けを行う方法です。もう 1 つは画像ファイル形式で保存し、そのグラフをアプリケーションにインポートする方法です。

C.2.1 クリップボードを使用してグラフをエクスポートする

Stata のグラフを最も簡単にエクスポートする方法はドラッグ& ドロップです。Graph ウィンドウでグラフを選択して移動したい先のアプリケーションにドラッグします。

または、コピー& ペーストでエクスポートすることも可能です。

始めに、グラフを作成するか、作成済みのグラフを再表示します。クリップボードにコピーするには Graph ウィンドウ内を右クリックし、コピーをクリックします。Stata は PDF フォーマットでグラフのコピーを作成します。このフォーマットは macOS のネイティブな画像形式です。ベクター形式を編集できるプログラムではグラフ上の要素を編集できます。

クリップボードにグラフをコピーした後、グラフをインポートするアプリケーションに切り替えてから貼り付けましょう。一般的にこの操作を行うにはメニューから編集 > 貼り付けと操作します。中には、あらかじめコピー先に境界ボックスの作成を必要とするアプリケーションもあります。単純な貼り付けができなかった場合、該当のアプリケーションのマニュアル等で画像の貼り付けに関する記述を参照してください。

C.2.2 グラフを他のファイルでエクスポートする

Stata はグラフを複数の画像ファイル形式でエクスポートできます。グラフを右クリックした後に名前を付けて保存**...** を選び、「ファイルの種類」をクリックすると次の形式を選択できます:

TIFF プレビュー付きまたはなしの Enhanced PostScript (EPS, . eps)、Graphics Interchange Format (GIF, . gif)、Joint Photographic Experts Group (JPEG, . jpg)、Portable Document Format (PDF, . pdf)、Portable Network Graphics (PNG, . png)、PostScript (PS, . ps)、Scalable Vector Graphics (SVG, . svg)、Tag Image File Format (TIFF, . tif)

PDF、PS、EPS、SVG はベクター形式で JPEG、GIF、PNG、TIFF はビットマップ形式です。グラフのサムネイルを使う時は、**EPS with TIFF preview** を選択してください。このプレビューオプションを選択してもグラフ印刷に影響はありません。PNG と SVG はウェブサイト等にグラフを掲載するのに適しています。詳しくは [G-2] **graph export** をご覧ください。

C.3Stata と Notification Manager

Stata が時間のかかるコマンドや do ファイルを実行している間でも、他のアプリケーションで作業できます。Stata は実行していたコマンドが終了すると、音と Dock にあるアプリケーションのアイコンが跳ねることで知らせます。デフォルトでは、アイコンを一度だけ跳ねさせるようになっています。このお知らせ機能は Stata がバックグラウンドに置かれている状態でコマンドか do ファイルが実行されている時のみ起動します。Stata はアプリケーションが前面にある場合やユーザからの指示が必要な場合（例えば、more）はお知らせをしません。お知らせがどのように動作するかは Stata の設定画面で

コントロールできます。

C.4 Mac OS X での Stata(コンソール) について

C.4.1 必須事項

Stata(console) は Mac 版の Stata/SE と Stata/MP に含まれています。ターミナルウィンドウ内で起動し、グラフィカルユーザインターフェイス（GUI）無し、つまりデータエディタ、ビューワ、Graph ウィンドウが無い状態で表示されます。グラフやデータセットは通常通り保存できますが、インタラクティブに表示することはできません。Stata(console) は Stata をリモート操作で実行したりバッチモードをバックグラウンドで行う際に使用します。

通常のインストールでもバックグラウンドで作業することも可能です。コマンドのオプションに関しては atjrefGSMB.3 Stata batch modeStata のバッチモードに記載されています。

Stata(console) をインストールする前に Stata/SE または Stata/MP を Mac にインストールしてライセンスを有効にしておく必要があります。シングルユーザライセンスを 1 つ所有していてログイン ID を複数必要としている場合、営業部へご連絡いただき、複数ユーザライセンスへのアップグレードをご相談下さい。Unix のシェルから操作できる状態で Stata(console) の設定を行ってください。また、同時にコンピュータに管理者権限（adminisitrator）でのアクセスが可能であることがセットアップには必要になってきます。管理者権限のアクセスでは、常にコンピュータにダメージを与える可能性があることを理解してください。

次のセクションのインストールでは StataMP を使用する場合は stata-mp を、StataSE の場合は stata-se を使用してください。

C.4.2 ターミナルユーティリティを使用して Stata(console) をインストールする

Stata をデフォルトのパス（/Applications/Stata）にインストールした場合、Stata(console) は **Stata >** ターミナルユーティリティをインストール**...** と選択するとインストールできます。コンソール版を起動して stata-se または stata-mp をターミナルウィンドウに入力して開始します。

Stata のフォルダ名を変更したり、ユーティリティをインストールした後にアプリケーションフォルダから移動している場合、コンソールへのシンボリックリンクが壊れていることになります。上記の場合、**Stata >** ターミナルユーティリティをインストール**...** と選択してユーティリティを再インストールします。

C.4.3Stata(console) を手動でインストールする

Stata(console) をビルトインユーティリティではなく手動でインストールする場合、このセクションを確認してください。Unix の操作に自信がない場合、あるいはコンピュータに害をなす可能性を危惧している場合、このセクションは飛ばしてください。もし、下記のコマンド

---

setenv PATH /Applications/Stata:$PATH

と、このコマンド

PATH=/Applications/Stata:$PATH

の意味が分からない場合、インストール自体をおすすめしません。

1. Mac に管理者権限（adminstrator privileges）があるアカウントでログインしてください。
2. まだインストールしていない場合、Stata/SE または Stata/MP を/Applications/Stata にインストールし、Stata のライセンスを有効にします。

インストールが完了しました。通常のユーザのように全てがインストールされているか、まずは確認します。Stata を一般のユーザとして使用するには、

/Applications/Stata/StataSE.app/Contents/MacOS をパスに含める必要があります。では、簡単に確認してみましょう。

```
mymac:> export PATH=$PATH:/Applications/Stata/StataSE.app/Contents/MacOS mymac:>
stata-se
```

Stata が起動するはずです。全てが正しく起動している事を確認してから、実際のユーザがシェルのスタートアップスクリプトのパスに/Applications/Stata/StataSE.app/Contents/MacOS を含むようにしてください。.bash profile ファイルをまだ所有していない場合、1 つ作成してから次の行を追加するようにしましょう。

```
export PATH=$PATH:/Applications/Stata/StataSE.app/Contents/MacOS
```

C.4.4Stata(console) をアップデートする

Stata(console) は通常の Stata（GUI）がアップデートされると、自動的にアップデートされます。全てのアップデートは通常の Stata（GUI）から行ってください。

C.5 Python から Stata を呼び出す

Python のパッケージがあれば、Python から Stata を呼び出すことができます。この機能には、 StataとＭａｔａ間のインタラクティブなＩ Python コマンドを API が含まれています。Python のパッケージ pystata の詳細は、https://www.stata.com/python/pystata/のオンラインドキュメントまたは、[P] Pystata module をご覧ください。

C.6 ライセンスを変更する

C.6 ライセンスを変更する

Stata を既にインストールしていて、ライセンスを変更する際は、**Stata > Stata** について**...**、と操作し、左下のライセンスの更新**...** ボタンをクリックします。更新オプションの確認か新しいライセンスの入力を選択できます。