An Introduction to Stata Programming

CHRISTOPHER F. BAUM
An Introduction to Stata Programming

Christopher F. Baum
Boston College

A Stata Press Publication
StataCorp LP
College Station, Texas
Preface

This book is a concise introduction to the art of Stata programming. It covers three types of programming that can be used in working with Stata: do-file programming, ado-file programming, and Mata functions that work in conjunction with do- and ado-files. Its emphasis is on the automation of your work with Stata and how programming on one or more of these levels can help you use Stata more effectively.

In the development of these concepts, I do not assume that you have prior experience with Stata programming, although familiarity with the command-line interface is helpful. Examples are drawn from several disciplines, although my background as an applied econometrician is evident in the selection of some sample problems. The introductory chapter motivates the why: why should you invest time and effort into learning Stata programming? In chapter 2, I discuss elementary concepts of the command-line interface and describe some commonly used tools for working with programs and datasets.

The format of the book may be unfamiliar to readers who have some familiarity with other books that help you learn how to use Stata. Beginning with chapter 3, each odd-numbered chapter is followed by a “cookbook” chapter containing several “recipes”, 40 in total. Each recipe poses a problem: how can I perform a certain task with Stata programming? The recipe then provides a complete solution to the problem and describes how the features presented in the previous chapter can be put to good use. As in the kitchen, you may not want to follow a recipe exactly from the cookbook; just as in cuisine, a minor variation on the recipe may meet your needs, or the techniques presented in that recipe can help you see how Stata programming applies to your specific problem.

Most Stata users who delve into programming make use of do-files to automate and document their work. Consequently, the major focus of the book is do-file programming, covered in chapters 3, 5, 7, and 9. Some users will find that writing formal Stata programs, or ado-files, meets their needs. Chapter 11 is a concise summary of ado-file programming, with the following cookbook chapter presenting several recipes that contain developed ado-files. Stata’s matrix programming language, Mata, can also be helpful in automating certain tasks. Chapter 13 presents a summary of Mata concepts and the key features that allow interchange of variables, scalars, macros, and matrices. The last chapter presents several examples of Mata functions developed to work with ado-files. All the do-files, ado-files, Mata functions, and datasets used in the book’s examples and recipes are available from the Stata Press web site, as discussed in Notation and typography.
In summary, you may want to consider whether the convenience of an *egen* function is offset by its computational burden. Coding the logic in your do-file can be a more efficient approach.\(^6\)

### 4.2 Computing summary statistics over groups

**The problem.** Your dataset has a hierarchical nature, where observations represent individuals who are also identified by their household ID code or represent records of individual patient visits that can be aggregated over the patient ID or over the clinic ID. In the latter case, you can define groups of observations belonging to a particular patient or to a particular clinic.

With this kind of hierarchical data structure, you may want to compute summary statistics for the groups. This can be performed readily in Stata with **tabstat**, but that command will only display a table of summary measures. Alternatively, you could use **collapse** to generate a dataset of aggregated values for a variety of summary statistics, or you could use **contract** to generate a collapsed dataset of frequencies. However, you may find that these options do not fit the bill.

What if you want to juxtapose the summary statistics for each aggregate unit with the individual observations to compute one or more variables for each record? For instance, you might have repeated-measures data for a physician’s patients measuring their height, weight, and blood pressure at the time of each office visit. You might want to flag observations where their weight is above their median weight, or where their blood pressure is above the 75th percentile of their repeated measurements.

**The solution.** Computations such as these can be done with a judicious use of by-groups (see section 3.5). For instance,

```
by patientid: egen medwt = median(weight)
by patientid: egen bp75 = pctlile(bp), p(75)
```

We have stressed that you should avoid using variables to store constant values (which would occur if you omitted the **by patientid:** prefix). But here we are storing a separate constant for each **patientid.** You can now compute indicators for weight, blood pressure, and at-risk status by using the **byte** data type for these binary variables:

```
generate byte highwt = weight > medwt & !missing(weight, medwt)
generate byte highbp = bp > bp75 & !missing(bp, bp75)
generate byte atrisk = highwt & highbp
```

If you need to calculate a sum for each group (**patientid here), you can use the **total()** function for **egen.** Alternatively, to improve computational efficiency, you could use

---

\(^6\) As I discuss in chapter 13, Mata functions may also prove useful in reducing the computational burden involved with tasks like these.
by patientid: generate atriskvisits = sum(atrisk)
by patientid: generate n.atrisk = atriskvisits if .n == .N
gsort -n.atrisk
list patientid n.atrisk if inrange(n.atrisk, 1, .)

This sequence of commands uses the `sum()` function from `generate`, which is a running sum. Its value when `.n == .N` is the total for that `patientid`. We store that value as `n.atrisk` and sort it in descending order with `gsort`. The `list` command then prints one record per `patientid` for those patients with at least one instance of `atrisk` in their repeated measures.

4.3 Computing the extreme values of a sequence

The problem. Let’s assume that you have hierarchical data, such as observations of individual patient visits to a clinic. In the previous recipe, we described how summary statistics for each patient could be calculated. These include extrema, for instance, the highest weight ever recorded for each patient or the lowest serum cholesterol reading. What you may need, however, is the record to date for those variables: the maximum (minimum) value observed so far in the sequence. This is a “record” value in the context of setting a record, for instance, maximum points scored per game or minimum time recorded for the 100-yard dash. How might you compute these values for hierarchical data?8

The solution. First, let’s consider a single sequence (that is, data for a single patient in our example above). You might be tempted to think that this is a case where looping over observations will be essential—and you would be wrong! We exploit the fact that Stata’s `generate` and `replace` commands respect Stata’s sort order (see Newson [2004]). We need record only the first observation’s value, and then we can use `replace` to generate the “record high”:

```
sort visitdate
generate maxwt = weight in 1
replace maxwt = max(maxwt[_n - 1], weight) in 2/1
```

Usually, you need not worry about missing values, because the `max()` function is smart enough to ignore them unless it is asked to compare missing with missing. If we want to calculate a similar measure for each `patientid` in the dataset, we use the same mechanism:

```
sort patientid visitdate
by patientid: generate minchol = serumchol if .n == 1
by patientid: replace minchol = min(minchol[_n - 1], serumchol) if .n > 1
```

7. The `gsort` command is presented in section 3.5.1.
8. This recipe relies heavily on the response to the Stata frequently asked question “How do I calculate the maximum or minimum seen so far in a sequence?” (http://www.stata.com/support/faqs/data/sequence2.html), written by Nicholas J. Cox.
With repeated-measures data, we cannot refer to observations 1, 2, etc., because those are absolute references to the entire dataset. Under the control of a by-group, the \texttt{n} and \texttt{N} values are redefined to refer to the observations in that by-group, allowing us to refer to \texttt{n} in the \texttt{generate} command and the prior observation in that by-group with a \texttt{[n - 1]} subscript.

### 4.4 Computing the length of spells

**The problem.** Assume that you have ordered data (for instance, a time series of measurements) and you would like to examine spells in the data. These might be periods during which a qualitative condition is unchanged, as signaled by an indicator variable. As examples, consider the sequence of periods during which a patient's cholesterol remains above the recommended level or a worker remains unemployed or a released offender stays clear of the law. Alternatively, spells might signal repeated values of a measured variable, such as the number of years that a given team has been ranked first in its league. Our concern with spells can involve identifying their existence and measuring their duration. This discussion of these issues relies heavily on Cox (2007b). I am grateful to Nick Cox for his cogent exposition.

**The solution.** One solution to this problem involves using a ready-made Stata command, \texttt{tsspell}, written by Nicholas J. Cox. This command can handle any aspect of our investigation. It does require that the underlying data be defined as a Stata time series with \texttt{tsset}. This makes it less than ideal if your data are ordered but not evenly spaced, such as patient visits to their physician, which can be irregularly timed.\footnote{As Cox points out (Cox 2007b, 250), the ordered data may not have a time dimension at all, but may refer to spatial orientation.} Another issue arises, though: that raised in section 4.1 with respect to \texttt{egen}. The \texttt{tsspell} program is fairly complicated interpreted code, which may impose a computational penalty when applied to a very large dataset. You may need only one simple feature of the program for your analysis. Thus you may want to consider analyzing the spells in do-file code, which is much simpler than the invocation of \texttt{tsspell}. As in section 4.1, you can generally avoid explicit looping over observations, and you will want to do so whenever possible.

Assume that you have a variable denoting the ordering of the data (which might be a Stata date or date-and-time variable, but need not be) and that the data have been sorted on that variable. The variable of interest is \texttt{employer}, which takes on the values A, B, C, ..., or missing for periods of unemployment. You want to identify the beginning of each spell with an indicator variable. How do we know that a spell has begun? The condition

\begin{verbatim}
generate byte beginspell = employer != employer'[n-1]
\end{verbatim}

will suffice to define the start of each new spell (using the \texttt{byte} data type to define this indicator variable). Of course, the data can be left-censored in the sense that we do not start observing the employee's job history on his or her date of hire. But the fact that \texttt{employer'[n-1]} is missing for period 1 does not matter, because it will be captured as
Author index

A
Azevedo, J. P. .............. 62, 173, 226

B
Blasnik, M. ............. 62, 173

C
Cañette, I. ...................... 264
Cook, R. D. ................. 185
Crow, K. ............ 161

D
Daly, F. ................. 59
Delwiche, L. D. .......... 25
Drukker, D. M. .......... 123, 327

E
Everitt, B. S. ............. 59

F
Franklin, C. H. .......... 112

G
Gini, R. ........ 184, 188
Greene, W. H. ....... 240

H
Hand, D. J. ............ 59

J
Jann, B. ........ 52, 98, 128, 181, 278, 286, 305, 328

K
Kantor, D. .......... 44
Kernighan, B. W. ...... 27
Kolev, G. L. .......... 59

L
Leuven, E. ............. 327
Longton, G. .......... 170
Lunn, A. D. ........ 59

M
McConway, K. J. .... 59
McDowell, A. .......... 331
Merton, R. C. ........ 166
Mitchell, M. N. ...... 185, 196

N
Newson, R. .......... 66

O
Ostrowski, E. .......... 59

P
Pasquini, J. .......... 184, 188
Pitblado, J. .......... 236, 263
Plauger, P. J. .......... 27

R
Rabe-Hesketh, S. .... 59
Rising, W. .......... 190, 230, 322
Author index

S
Schaffer, M. E. .......... 236, 261, 337
Sianesi, B. ................. 327
Slaughter, S. J. .......... 25
Sribney, W. .............. 236, 263
Stillman, S. ............. 236, 261

W
Wada, R. ................... 99
Weesie, J. ................. 105
Weisberg, S. ............. 185
Wiggins, V. ............... 132
Wooldridge, J. M. ....... 182

Y
Yaron, A. .................. 337
Subject index

A
added-variable plot ................1 8 5
ado-path. ............................234, 304
adoupdate command ............4, 30
Akaike information criterion .... 94
anova command ...................1 4 5
append command ........................80, 105, 107
args command ....................237, 240
ASCII text file ..................1 6, 178
assert command ....................19, 234
atanh() function ...............267
audit trail ............................79
autocorrelation ..................1 6 5
automated output ..............1 8 1
avplot command ...................1 8 5

B
_b saved results ..................249
backslash .................................74
Bayesian information criterion ... 94
binary file handling .............179
biprobit command ............. 267
Boolean condition ..................43
bootstrap prefix command .... 151, 242, 244
built-in commands ..............86
built-in function ..................316
business-daily data ....... 123, 213
by prefix ..............................140
byable ..................................47, 226
bysort prefix .....................1 4 0
byte data type ...................92
bytecode ..............................271

calendar variable .............88
capture command ..............1 4
casewise deletion ............36, 92, 175
ceil() function ...............45
center command ...............52, 286
Center for Research in Security Prices ..................191
certification script ......... 234, 254, 345
char command ....................1 8 9
characteristics ....................1 8 8, 285
ckvar command ....................1 9 0
class programming ................1
codebook command .......... 20, 81
codebooks .............................20
collapse command ..........14, 53, 65, 100,
135, 175, 203, 288
colon operator ....................274
column-join operator ........1 7 4, 198, 272
column-range operator .... 273
comma-delimited file ...........18
comma-separated–values file ...1 7 8
compiled function ..................303
complex arithmetic ..................271
compound double quotes 39
compress command .............25
cond() function .............44, 268
conditional statements .........278
confirm command ..............287, 307
constraint command ............2 6 1
continuous price series .......209
cross() command .............14, 65, 175
correlate command 37, 89, 152, 192,
331
correlation() Mata function ....313
creturn command .............57, 76
cross command ....................1 1 2
cross() Mata function .............334
cscript command ...................2 5 4
CSV file...see comma-separated–values file

D
D. time-series operator ............13
daily data ........................123
data dictionary ....................22
data validation .....................80
decode command ...................39
decrement operator ..............277
delimiters ...........................17
delta method .......................241
dereferencing .......................54
describe command .................35, 87
desstring command .................38
dialog programming .................1
dictionary file .....................178
difference operator ..............see D.
diparm() option .................263
directory separator ...............74
display command ................31, 39
Do-file Editor ..................8
dofm() function .................132
dot product ........................273
duplicates command .............85, 111
dyex option .......................181
dynamic panel data ...............267

E
e(b) matrix .......................92
e(sample) function .........90, 92
e-class ..............................86
eform() option ...................228
egen anycount() command .......63
egen bom() command ............49
egen bomd() command ............49
egen command ........37, 47, 63, 231, 252, 288
egen corr() command ............49
egen count() command ............48
egen eom() command ............49
egen eomd() command ............49
egen ewma() command ............49
egen filter() command ............49
egen gmean() command ............49
egen group() command ............141
egen hmean() command ............49
egen iqr() command .............48
egen kurt() command .............48
egen max() command ............48
egen mdev() command ............48
egen mean() command .........48, 211
egen median() command ............48
egen min() command ............48
egen mode() command ............48
egen nvals() command ............169
egen pc() command ...............48
egen pctlile() command .........48, 232, 252
egen rowmax() command ............47
egen rowmean() command ............47
egen rowmin() command ............47
egen rowmiss() command .........47, 63
egen rownonmiss() command .........47, 63
egen rowsd() command ............47
egen rowtotal() command ............47
egen std() command ...............48
egen skew() command ............48
egen tag() command .............169
egen total() command ............48, 65
egen var() command .............49
egen more package ............49, 63, 169
elementwise operations ...........273
embedded spaces ...................17
.eps format .....................187
ereturn display command .......333, 338
ereturn list command ..........90, 93
ereturn post command ..........333
estadd command .................98, 129
estat ovtest command ..........93
estat vce command ..........93
estimates command ..........93
estimates for command ..........96
estimates notes command .......97
estimates replay command ......94
estimates save command ......94, 97
estimates stats command ......91
estimates store command .....93, 128, 260
estimates table command ...94, 128
estimates use command .......94
estimation commands ..........86
estout command ........98, 128, 181
eststo command ................181
esttab command ........129, 181
Euclidian distance ..........327
extended macro functions .56, 159, 172
external declaration ..........342
eydx option ..................182
eyex option ..................181, 196
F
F. time-series operator ........13
FAQs ................................29
fdasave command ...........178
fdause command ..............16
file command .................178
file handle ....................179
file open command ............179
file read command ............180
file write command ..........179
fillin command ...............112
findexternal() Mata function ..280
findit command ..............4, 29, 217
finite-precision arithmetic ...257
fixed format ..................17, 20
floor() function .............45
foreach command ............55, 154, 172
format() option ..............39
forvalues command ...........154
freduse command .............123
free format ...................17
function arguments ..........279
function library ..............304
G
generalized method of moments ..337
generate command .............40
global command .....56, 73, 259, 265
global macro ..................73, 155
GMM-CUE ........................337
.gph file suffix .............187, 204
graph combine command ......186
graph display command .......187
Graph Editor ..................188, 205
graph export command .......187
graph save command .........205
graph twoway connected command .......196
graph twoway rarea command ...251
graph twoway rline command ...196
graph twoway scatter command ........196
graphs ........................185
groupwise heteroskedasticity ....242
gsort command ..............66, 192
H
help files ......................228
HTML ................................98, 229
I
if qualifier ...................35, 36
implied decimal ..............24
in qualifier ...................35
include command .............281
increment operator ..........277
indicator variable ............42, 186
inequality constraints .......262
infile command .............17, 23, 115, 118
infix command ..............22, 202
inlist() function ............41, 84
inner product ................273
inrange() function ..........41, 64, 83
insheet command ............18, 38, 203
instrumental variables ......337
int() function ..............41, 46
integer division ..............41
interaction terms ............143
interval estimates ...........195
invsym() Mata function ......334
irecode() function ..........46
Subject index

ivreg2 command 236, 261, 337, 346
ivregress command 341

J
J() Mata function 172, 275
jackknife prefix 242
joinby command 115

K
Kronecker product 274

L
L. time-series operator 13
L2-norm 327, 328
lag operator see L
latent variable 131
BFGS output 98, 173, 174, 181, 197
lead operator see F
levelsof command 75, 172, 191
likelihood function 91
limits 316
lincom command 94, 182, 232, 249
linear constraints 261
linear filter see filter()
list subscripts 274
liststruct() Mata function 301
listwise deletion 175
local command 53
local macro 155
log price relative 209
log2html command 230
logit command 130
long form 100
longitudinal data 101
loop constructs 154, 276

M
Macintosh spreadsheet dates 20
macro 53–56
macro evaluation 54
macro list functions 57, 75, 159
mad() command 48
makematrix command 198
Mann–Whitney test 199
many-to-many merge 111
marginal effects 194
marksample command 223, 268
mat2txt command 62, 173
Mata 60, 271
.mata file suffix 303
mata mlib add command 305
mata mlib create command 304
mata mosave command 304
match-merge 109
matrix accum command 61
matrix colnames command 62, 92, 172, 180, 308
matrix command 60, 172
matrix language 60
matrix list command 60, 92, 173
matrix programming 271
matrix rownames command 62, 92, 172, 180, 308
matrix stripes 308, 334
matsize command 316
max() function 66
maxbyte() function 42
maximum likelihood 261, 262, 337
maxindex() Mata function 314
maxint() function 42
maxlong() function 42
mean() Mata function 314
merge command 80, 106, 109, 203, 207
_merge variable 109
method d0 264
mfx command 128, 181, 195, 232
min() function 66
minindex() Mata function 328
missing() function 36
missing string value 37
missing values 43
mkmat command 61
ml command 337
.mlib file suffix 304
mlopts parsing tool 238
mm_cond() Mata function 278
mm_meancolvar() Mata function 328
.mo file suffix 303
mod() function 42, 132
Subject index

month() function ........................................... 132
moremata package ............................................ 278, 305, 328
moving correlation ........................................... 165
moving-window statistics .................................... 162
mreldif() function ............................................ 256
mvcorr command ............................................. 165
mvdecode command .......................................... 37
mvencode command .......................................... 37
mvsumm command ............................................. 162

N
naturally coded indicator ................................... 142
nearest neighbor ............................................. 327
nested loop .................................................... 155, 157
nested macros ................................................. 54
nestreg command ............................................. 153
net install command ......................................... 7
njcstuff package ............................................. 188
nlcom command ............................................... 241, 260, 266
normalden() function ......................................... 263
null model ...................................................... 91
null pointer ................................................... 297
nullmat() function ........................................... 174, 199
numlist .......................................................... 35

O
object file ..................................................... 303
object-oriented programming ................................ 1
ODBC .......................................................... 26
odbc command ................................................. 26
one-to-many merge ............................................ 109, 118
one-to-one merge ............................................. 110
optimize_init_evaluator() Mata function .................. 343
optimize_init_evaluator_type() Mata function .......... 343
optimize_init() Mata function ................................ 342
optimize_init_params() Mata function ..................... 343
optimize_init_which() Mata function ....................... 343
optimize() Mata function .................................... 337, 343
optimize_result_value() Mata function .................. 343
optional arguments ........................................... 280

P
p-value .......................................................... 199
panel data ....................................................... 70, 101, 267
panel variable ................................................... 88
panelsetup() Mata function .................................. 313
panelsubmatrix() Mata function ............................ 313
panelsubview() Mata function ................................ 316
passing functions to functions ................................ 316
pctrange command ............................................ 254
.pdf format ..................................................... 187
percentiles ..................................................... 195
permute command ............................................ 151
.png format ..................................................... 187
point estimates ................................................ 195
pointer function .............................................. 321
pointer variables .............................................. 296
pointers ......................................................... 316, 334
pointers to pointers .......................................... 298
post command .................................................. 177
postfile command ............................................. 176
predict command .............................................. 10, 232, 332
prefix commands .............................................. 139
preserve command ............................................ 14, 175, 268
price deflator .................................................. 162
probit command ................................................ 130
program command ............................................ 218
program properties .......................................... 228
propensity score .............................................. 327
psmatch2 command ............................................ 327
pwcorr command ............................................... 37, 331

Q
qofd() function ............................................... 134
qreg command .................................................. 147
Subject index

R
r-class ................................ 86, 89
range subscripts ................. 274
ranksum command .................. 199
.raw file suffix ................. 178
rc0 option ..................... 142
rcod command ................. 256
real() function ............... 38
recod command ............... 45, 156
record() function ........... 45
recursive estimation .......... 146
reg3 command ............... 332
regress command ............ 332
relational operators ............. 274
relative difference .......... 154
reldif() function ........... 154, 257
rename command ............ 105, 107
renvars command ........... 105
replace command ........... 40
replay feature .............. 238
reserved words .............. 280
reshape command .......... 100, 290
restore command ........... 14, 175, 269
return list command ....... 86
return types .................. 279
reverse() function ........ 203
reverse recursive estimation .. 146
rich text format .......... 98, 99
rnormal() (Mata) .............. 276
robvar command ............. 242
rolling prefix ............ 146, 162, 249
rolling-window estimation .... 146
root finders ................. 305
round() function .......... 46
row-join operator ........ 174, 198, 273
row-range operator ........ 272
rowwise functions ........... 47

S
S. time-series operator .... 13
SAS XPORT Transport file .... 178
savedresults command ....... 259, 346
scalar ..................... 257
scalar command .......... 58
scheme programming ........ 1
Schwarz criterion .......... 94
_se saved results .......... 249
search command .......... 29
seasonally difference operator see S.
seemingly unrelated regression ... 331
semielasticity ............. 181
separate command ....... 114, 186
serse command ........... 205
set rmsg command ........... 347
set seed command .......... 149, 243
set trace command .......... 31
similarity .................. 327
simulate command .......... 148, 242
SMCL ................................
sort command ............ 98, 228
sort order ................. 87
sortpreserve option ....... 238
space-delimited files .... 17
spells ..................... 67, 267
sphdist command .......... 322
spreadsheet data ........... 20
SQL databases ............. 26
SSC archive ................ 29, 202, 259
ssc command ............. 7, 47, 202
ssc hot command ........... 204
st_addvar() Mata function . 284
st_data() Mata function .... 283
st_global() Mata function .... 285
st_local() Mata function .... 285
st_matrix() Mata function .... 285
st_matrixcolstripe() Mata function .... 308
st_matrixrowstripe() Mata function .... 308
st_nobs() Mata function .... 283
st_numscalar() Mata function .... 282, 285
st_nvar() Mata function .... 283
st_sdata() Mata function .... 283, 285
st_sstore() Mata function .... 284, 285
st_store() Mata function .... 284
st_strscalar() Mata function .... 285
st_subview() Mata function .... 285, 319
st_sview() Mata function .... 283, 285
Subject index

st_tsrevar() Mata function .......... 339
st_varindex() Mata function .......... 286
st_varname() Mata function .......... 286
st_view() Mata function .......... 282, 322
stack command ...................... 112
standardized values see egen std() command
stars, for significance ................ 94
Stat/Transfer ............... 25, 37, 178
stata() Mata function .......... 322
Stata Journal ...................... 7, 29
Stata Technical Bulletin ......... 7, 29
Statalist ... 30, 119, 126, 163, 167, 169, 191, 249
Statistical Software Components
archive see SSC archive
statsby command ............... 145
statsmat command .......... 62, 175
stepwise command ............... 153
stock returns ..................... 191
storage optimization ............... 25
string() function ............... 39
string missing values .......... 37
strpos() function .......... 203
strtoreal() Mata function ...... 286
structure .......................... 299
stset command ............... 189
substr() function .......... 203
subview matrix ...................... 319
sum() function ............... 42, 169
summarize command .......... 87, 192
sureg command ..................... 331
svmat command ............... 192
svy command ..................... 153
svyset command .......... 153, 189
syntax command ........ 218, 231, 307
sysdir command ............... 6

tab-delimited file .......... 18, 178
tabstat command ........ 44, 62, 65, 175, 198
tabulate command .......... 84, 142
tabulating results .......... 94
tempname command ........ 176, 200
temporary objects .............. 295
temporary variable .......... 223
tempvar command .......... 195, 223, 268
test command ............... 94, 232
text editors .............. 16
text files .............. 16
texteditors command .......... 16	.tif format ..................... 187
time-series calendar ......... 132
time-series operators .......... 12, 257
tokenize command .......... 157
tokens() Mata function ...... 288
tostring command .......... 39
transmorphic matrix .......... 279
transpose ..................... 115
transpose operator .......... 273
tscollap command .......... 136, 288
tsfill command .......... 112, 162, 268
tslide command .......... 124, 163, 251
tsfmtim command .......... 132, 291
tsset command ........... 13, 49, 88, 123, 132, 189, 268
tsspell command .......... 67, 267
ttest command .......... 89, 152, 199, 242
type d0 evaluator .......... 343
U
unbalanced panel .......... 112, 267, 331
update command .......... 6, 30
update option .......... 208
V
variable indices .......... 286
varlist ......................... 35
vce(bootstrap) option .......... 152
vce(jackknife) option .......... 153
version command .............. 218
veryshortlabel option .......... 114
view matrices .............. 271, 285, 319
viewsourcen command .......... 86, 305
void matrices .......... 279
W
which command .......... 31, 86, 217
while command .......... 154
wide form ....................... 99, 331
wildcards ..................... 35, 47, 103
word processors ................. 16
wrapper program ................ 249

X
xi command ..................... 142
xpose command .................. 115
xt command ..................... 100, 331
xtabond command ............... 267
xtabond2 command ............. 267
xtile command .................. 46
xtset command .................. 13, 88, 102, 181, 188