Title

stata.com

memory — Memory management

Description

Memory usage and settings are described here.

`memory` displays a report on Stata's current memory usage.

`query memory` displays the current values of Stata's memory settings.

`set maxvar`, `set niceness`, `set min_memory`, `set max_memory`, and `set segmentsize` change the values of the memory settings.

If you are a Unix user, see *Serious bug in Linux OS* under *Remarks and examples* below.

Quick start

Display memory usage report

```
memory
```

Display memory settings

```
query memory
```

Increase the maximum number of variables to 8,000 in Stata/MP or Stata/SE

```
set maxvar 8000
```

Set maximum memory allocation to avoid potential memory allocation bug in Linux

```
set max_memory 16g, permanently
```

Syntax

Display memory usage report

```
memory
```

Display memory settings

```
query memory
```

Modify memory settings

```
set maxvar      #    [ , permanently ]
set niceness    #    [ , permanently ]
set min_memory  amt  [ , permanently ]
set max_memory  amt  [ , permanently ]
set segmentsize amt  [ , permanently ]
```

where *amt* is #[b | k | m | g], and the default unit is b.

Parameter	Default	Minimum	Maximum	
maxvar	5000	2048	120000	(MP)
	5000	2048	32767	(SE)
	2048	2048	2048	(IC)
niceness	5	0	10	
min_memory	0	0	max_memory	
max_memory	.	2×segmentsize	.	
segmentsize	32m	1m	32g	(64-bit)
	16m	1m	1g	(32-bit)

Notes:

1. The maximum number of variables in your dataset is limited to maxvar. The default value of maxvar is 5,000 for Stata/MP and Stata/SE, and 2,048 for Stata/IC. With Stata/MP and Stata/SE, this default value may be increased by using set maxvar. The default value is fixed for Stata/IC.
2. Most users do not need to read beyond this point. Stata's memory management is completely automatic. If, however, you are using the Linux operating system, see *Serious bug in Linux OS* under *Remarks and examples* below.
3. The maximum number of observations is fixed at 1,099,511,627,775 for Stata/MP and is fixed at 2,147,483,647 for Stata/SE and Stata/IC regardless of computer size or memory settings. Depending on the amount of memory on your computer, you may face a lower practical limit. See help obs_advice.

4. `max_memory` specifies the maximum amount of memory Stata can use to store your data. The default of missing (`.`) means all the memory the operating system is willing to supply. There are three reasons to change the value from missing to a finite number.
    1. You are a Linux user; see *Serious bug in Linux OS* under *Remarks and examples* below.
    2. You wish to reduce the chances of accidents, such as typing `expand 100000` with a large dataset in memory and actually having Stata do it. You would rather see an insufficient-memory error message. Set `max_memory` to the amount of physical memory on your computer or more than that if you are willing to use virtual memory.
    3. You are a system administrator; see *Notes for system administrators* under *Remarks and examples* below.
5. The remaining memory parameters—`niceness`, `min_memory`, and `segment_size`—affect efficiency only; they do not affect the size of datasets you can analyze.
6. Memory amounts for `min_memory`, `max_memory`, and `segmentsize` may be specified in bytes, kilobytes, megabytes, or gigabytes; suffix `b`, `k`, `m`, or `g` to the end of the number. The following are equivalent ways of specifying 1 gigabyte:

```
1073741824
   1048576k
      1024m
         1g
```

   Suffix `k` is defined as (multiply by) 1024, `m` is defined as 1024^2, and `g` is defined as 1024^3.
7. 64-bit computers can theoretically provide up to 18,446,744,073,709,551,616 bytes of memory, equivalent to 17,179,869,184 gigabytes, 16,777,216 terabytes, 16,384 petabytes, or 16 exabytes. Real computers have less.
8. 32-bit computers can theoretically provide up to 4,294,967,296 bytes of memory, equivalent to 4,194,304 kilobytes, 4,096 megabytes, or 4 gigabytes. Most 32-bit operating systems limit Stata to half that.
9. Stata allocates memory for data in units of `segmentsize`. Smaller values of `segmentsize` can result in more efficient use of available memory but require Stata to jump around more. The default provides a good balance. We recommend resetting `segmentsize` only if your computer has large amounts of memory.
10. If you have large amounts of memory and you use it to process large datasets, you may wish to increase `segmentsize`. Suggested values are

memory	`segmentsize`
32g	64m
64g	128m
128g	256m
256g	512m
512g	1g
1024g	2g

11. `niceness` affects how soon Stata gives back unused segments to the operating system. If Stata releases them too soon, it often needs to turn around and get them right back. If Stata waits too long, Stata is consuming memory that it is not using. One reason to give memory

back is to be nice to other users on multiuser systems or to be nice to yourself if you are running other processes.

The default value of 5 is defined to provide good performance. Waiting times are currently defined as

niceness	waiting time (m:s)
10	0:00.000
9	0:00.125
8	0:00.500
7	0:01
6	0:30
5	1:00
4	5:00
3	10:00
2	15:00
1	20:00
0	30:00

Niceness 10 corresponds to being totally nice. Niceness 0 corresponds to being an inconsiderate, self-centered, totally selfish jerk.

12. `min_memory` specifies an amount of memory Stata will not fall below. For instance, you have a long do-file. You know that late in the do-file, you will need 8 gigabytes. You want to ensure that the memory will be available later. At the start of your do-file, you `set min_memory 8g`.

13. Concerning `min_memory` and `max_memory`, be aware that Stata allocates memory in `segmentsize` blocks. Both `min_memory` and `max_memory` are rounded down. Thus the actual minimum memory Stata will reserve will be

```
segmentsize*trunc(min_memory/segmentsize)
```

The effective maximum memory is calculated similarly. (Stata does not round up `min_memory` because some users set `min_memory` equal to `max_memory`.)

Options

`permanently` specifies that, in addition to making the change right now, the new limit be remembered and become the default setting when you invoke Stata.

`once` is not shown in the syntax diagram but is allowed with `set niceness`, `set min_memory`, `set max_memory`, and `set segmentsize`. It is for use by system administrators; see *Notes for system administrators* under *Remarks and examples* below.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Examples

Here is our memory-usage report after we load `auto.dta` that comes with Stata using Stata/MP:

```
. sysuse auto
(1978 Automobile Data)
. memory
```

Memory usage

	used	allocated
data	3,182	67,108,864
strLs	0	0
data & strLs	3,182	67,108,864
data & strLs	3,182	67,108,864
var. names, %fmts, ...	4,151	67,421
overhead	8	800
Stata matrices	0	0
ado-files	5,400	5,400
stored results	0	0
Mata matrices	0	0
Mata functions	0	0
set maxvar usage	5,271,736	5,271,736
other	3,691	3,691
grand total	5,284,248	72,457,912

We could then obtain the current memory-settings report by typing

```
. query memory
```

```
Memory settings
  set maxvar          5000       2048-32767; max. vars allowed
  set matsize         400        10-11000; max. # vars in models
  set niceness        5          0-10
  set min_memory      0          0-1600g
  set max_memory      .          32m-1600g or .
  set segmentsize     32m        1m-32g
```

Serious bug in Linux OS

If you use Linux OS, we strongly suggest that you set `max_memory`. Here's why:

> "By default, Linux follows an optimistic memory allocation strategy. This means that when malloc() returns non-NULL there is no guarantee that the memory really is available. This is a really bad bug. In case it turns out that the system is out of memory, one or more processes will be killed by the infamous OOM killer. In case Linux is employed under circumstances where it would be less desirable to suddenly lose some randomly picked processes, and moreover the kernel version is sufficiently recent, one can switch off this overcommitting behavior using [...]"
>
> – Output from Unix command `man malloc`.

What this means is that Stata requests memory from Linux, Linux says yes, and then later when Stata uses that memory, the memory might not be available and Linux crashes Stata, or worse. The Linux documentation writer exercised admirable restraint. This bug can cause Linux itself to crash. It is easy.

The proponents of this behavior call it "optimistic memory allocation". We will, like the documentation writer, refer to it as a bug.

The bug is fixable. Type `man malloc` at the Unix prompt for instructions. Note that `man malloc` is an instruction of Unix, not Stata. If the bug is not mentioned, perhaps it has been fixed. Before assuming that, we suggest using a search engine to search for "linux optimistic memory allocation".

Alternatively, Stata can live with the bug if you set `max_memory`. Find out how much physical memory is on your computer and `set max_memory` to that. If you want to use virtual memory, you might set it larger, just make sure your Linux system can provide the requested memory. Specify the option `permanently` so you only need to do this once. For example,

```
. set max_memory 16g, permanently
```

Doing this does not guarantee that the bug does not bite, but it makes it unlikely.

Notes for system administrators

System administrators can set `max_memory`, `min_memory`, and `niceness` so that Stata users cannot change them. You may want to do this on shared computers to prevent individual users from hogging resources.

There is no reason you would want to do this on users' personal computers.

You can also set `segmentsize`, but there is no reason to do this even on shared systems.

The instructions are to create (or edit) the text file `sysprofile.do` in the directory where the Stata executable resides. Add the lines

```
set min_memory 0, once
set max_memory 16g, once
set niceness 5, once
```

The file must be plain text, and there must be end-of-line characters at the end of each line, including the last line. Blank lines at the end are recommended.

The `16g` on `set max_memory` is merely for example. Choose an appropriate number.

The values of 0 for `min_memory` and 5 for `niceness` are recommended.

Stored results

`memory` stores all reported numbers in `r()`. StataCorp may change what `memory` reports, and you should not expect the same `r()` results to exist in future versions of Stata. To see the stored results from `memory`, type `return list, all`.

Reference

Sasieni, P. D. 1997. ip20: Checking for sufficient memory to add variables. *Stata Technical Bulletin* 40: 13. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, p. 86. College Station, TX: Stata Press.

Also see

[P] **creturn** — Return c-class values

[R] **matsize** — Set the maximum number of variables in a model

[R] **query** — Display system parameters

[U] **6 Managing memory**