



Fraunhofer Institute
Algorithms and
Scientific Computing

Getting Started



May 2005

Tanja Clees
Resi Höver-Klier
Klaus Stüben

Fraunhofer Institute SCAI
Schloss Birlinghoven
D-53754 St. Augustin, Germany

e-mail: samg@scai.fraunhofer.de

Document version 1.0

Purpose of this Document

This document provides some basic help on the installation of the Fortran90 library SAMG, separately for the individual types of licenses. We also comment on subroutines which might need some user-dependent adaptation and make remarks on calling SAMG routines from within C or C++.

This document does not cover the details for all possible system environments. Please check for additional readme files containing most recent information. However, in general, there should be no problems in installing SAMG. Should you need additional help, do not hesitate to contact us via our service e-mail,

samg@scai.fraunhofer.de¹.

Your SAMG Team

Contents

1	General Remarks and Conventions	3
1.1	Archives provided.....	3
1.2	Calling SAMG from C or C++.....	3
2	Binary License for Windows	4
2.1	The SAMG library.....	4
2.2	The FLEXIm server.....	5
3	Binary License for Linux/Unix	6
3.1	The SAMG library.....	6
3.2	The FLEXIm server.....	7
4	Source License	8
4.1	Installation notes.....	8
4.2	Integration of SAMG source code into your code/project.....	8
4.3	Using Intel Visual Fortran, Microsoft .NET and the like.....	9
4.4	Further important remarks.....	9

¹ **Important:** This e-mail addresses the whole SAMG Team. Therefore, it should not be used to transmit large amounts of data.

1 General Remarks and Conventions

1.1 Archives provided

Depending on the type of license purchased, one or more archives² are provided:

- The **SAMG package** for which you purchased a license is shipped in an archive named [samgXXX](#). Here [XXX](#) characterizes various aspects, for instance, SAMG's version number, the compiler used (in case of binary licenses), and the like.
- In case of counted and/or floating licenses, a **FLEXlm server** needs to be installed and started before the SAMG library can be used. Everything needed in this context is contained in a [flexlm_enduser](#) archive.

1.2 Calling SAMG from C or C++

SAMG is written in Fortran90. However, there are no significant problems to call SAMG from within C(++). The only mixed-language problems relevant here are caused by **inconsistent calling conventions** between C(++) and Fortran90. In particular, some precaution has to be taken in order to make the linker find those Fortran90 routines which need to be known on the level of C(++). Apart from the main SAMG interfaces themselves, these are routines such as [samg_reset_hidden](#), [samg_cleanup](#), [samg_refresh](#) and [samg_ctime](#), as well as all routines required to set or get values of hidden SAMG parameters.

We make the following **naming convention**: On the level of a C(++)-program, the names of all SAMG routines called are **capitalized**. According to the rules defined in the header file [samg.h](#), being part of every SAMG package, the C pre-compiler then changes all capitalized names to those names really produced by the Fortran90 compiler which was used to build the SAMG library (the "SAMG compiler"). In the [makefile\(s\)](#) provided for the demo drivers (binary license) or the include files (source license), the correct "-D" flag(s) for the respective SAMG compiler are already set for you. Typical flags are:

- for Windows: no "/D" flag at all,
- for Unix/Linux: always "-DSAMG_UNIX_LINUX" and, for a SAMG compiler creating ...
 -upper-case names (e.g. [SAMG_CTIME](#)): no additional "-D" flag;
 -lower-case names (e.g. [samg_ctime](#)): -DSAMG_LCASE;
 -lower-case names with trailing underscore (e.g. [samg_ctime_](#)): -DSAMG_LCASE_USCORE.

Windows only:

In addition, be aware of the fact that **Compaq Visual Fortran** and **Intel Visual Fortran** use different calling conventions (`__stdcall` and `__cdecl`, respectively). Since the SAMG DLL has been built with **Intel Visual Fortran**, you have to make sure that your compiler uses `__cdecl` for the SAMG routines, regardless whether you call SAMG from C(++) code, or Fortran90 code built by a Fortran90 compiler different from **Intel Visual Fortran** (with default settings). To be more specific:

Windows binary license: If you build the code calling SAMG with Microsoft Visual C++ or Intel Visual Fortran (with default settings for the calling convention), simply use/adapt [samg.h](#). No "/D" flag is needed. However, if you build the code calling SAMG with Compaq Visual Fortran, use the compiler flag `/iface:cref`. A corresponding makefile for SAMG's demos is provided.

Windows source license: If you build the SAMG library with Intel Visual Fortran using default settings for the calling convention, the same rules as described above apply. In case you build SAMG with Compaq Visual Fortran and build the code calling SAMG with Microsoft Visual C++, use the compiler flag `/DSAMG_CVF` for the C(++) code (see [samg.h](#) and the include files provided).

For further information, we refer to the header file [samg.h](#), the C++ demo driver [democ.cpp](#) (which is the analog of the Fortran90 demo driver, [demo.f](#)) and the [makefile\(s\)](#) for the demo drivers (binary license) or the include files (source license) provided.

² Usually [.zip](#) and [.tar.gz](#) files for Windows and Linux/Unix systems, respectively.

2 Binary License for Windows

2.1 The SAMG library

2.1.1 Contents of the SAMG archive

The `samgXXX` archive provided contains the following files:

- A *dynamic link* SAMG library (DLL) including a `.dll` and a corresponding `.lib` file³. They have been built with Intel's Fortran90 compiler.
- `demo.f90`: Fortran90 demo driver to be linked with the SAMG library.
- `democ.cpp`, `samg.h`: C++ demo driver and necessary header file to be linked with the SAMG library.
- A batch file `ifl_make.bat` and a makefile `make_dyn.ifl`: for compiling and linking the Fortran90 and C++ demo drivers using the "ifort" (**Intel Visual Fortran**) and "cl" (**Microsoft C++**) compiler, respectively.
- A batch file `wdf_ifl_make.bat` and a makefile `make_dyn.wdf_ifl`: for compiling and linking the Fortran90 and C++ demo drivers using the "df" (**Compaq Visual Fortran**) or "cl" (**Microsoft C++**) compiler, respectively.
- `demo.exe`, `democ.exe`: Pre-compiled executables of both the Fortran90 and C++ demo drivers needing the SAMG DLL.
- `demo.frm`, `demo.amg`, `demo.rhs` and `demo.sol`: A set of data files describing a small linear system of equations (matrix, right hand side and solution). These files are read by the demo drivers which then solve the corresponding linear system. All files have to be located in the directory from which the drivers are launched.
- `out_democ.txt`: Output of `democ.exe`. This should virtually be the same as the output of `demo.exe`.

2.1.2 Installation/Activation of the SAMG license key

Assuming that your SAMG installation requires a license key, be sure that you unzip the archive to the machine(s) for which you do have a valid license. Then proceed as follows:

- In case of **counted** and/or **floating licenses**, install and start a FLEXlm server as described in the corresponding installation notes, see Section 2.2.
- In case of a **node-locked license**, copy the license file provided to the corresponding machine. Let the environment variable `LM_LICENSE_FILE` point to (or include) this license file, e.g., by issuing

```
set LM_LICENSE_FILE=d:\samg\FHGSCAI.lic
```

from the active console. This assumes that `LM_LICENSE_FILE` is a new environment variable, and that the license file is named `FHGSCAI.lic`, located in the directory `d:\samg`. For permanent use, add this environment variable to Windows' list of environment variables.

2.1.3 First steps towards integrating SAMG

- In order to check if the SAMG library and the license file can be accessed, start with launching the pre-compiled `demo.exe` and/or `democ.exe`.
- Continue with modifying one of the makefiles according to your compilation environment and build your own `demo(c).exe` by running the corresponding batch file. The demo drivers `demo.f90` and `democ.cpp` are similar so that it is sufficient to build only one of these demos.
- You may take `demo.f90` and/or `democ.cpp` as examples of how to call SAMG routines from within your own code, and of how to set parameters controlling different aspects of SAMG⁴.
- **ATTENTION:** Please study the `makefiles` provided prior to linking SAMG to your own Fortran90 and/or C++ code. In case of linking to C(++), please study also `samg.h`.
- Make sure that the SAMG DLL resides in the directory from where you launch your own code. Alternatively, add the path of the SAMD DLL to Windows' `PATH` environment variable.

³ Unless otherwise requested, we provide a DLL. If, for important reasons, a *static* library is needed, just let us know.

⁴ For a detailed description of all parameters and their options, see the SAMG User's Manual.

2.2 The FLEXlm server

Below we describe steps to get the FLEXlm server started for SAMG. For more information on FLEXlm and its operation, we refer to the **FLEXlm User's Guide** which is provided as part of SAMG's FLEXlm archive.

2.2.1 Contents of SAMG's FLEXlm archive

The `flexlm_enduser` archive contains the following files:

- `lmgrd.exe`: the FLEXlm daemon, version 10.0;
- `FHGSCAI.exe`: our vendor daemon; requires FLEXlm v.9.2 or later;
- `lmtools.exe`: a Windows program for configuring FLEXlm services, starting and stopping the license server, testing license features etc.;
- `lmutil.exe`: a corresponding collection of console utilities;
- `flexlm_enduser_guide.pdf`: the original FLEXlm End Users Guide.

2.2.2 Installation of the necessary files to run FLEXlm for SAMG

Locate the machine which is intended to be the SAMG-FLEXlm license server (according to the data you specified to us). Depending on whether or not FLEXlm is already installed on that machine, do the following:

- If FLEXlm is *not* yet installed, unpack the `flexlm_enduser` archive to a directory on this machine.
- If FLEXlm is already installed, start its `lmtools.exe` and go to "Help" -> "About" to identify the version number. Then do the following:
 - If the version number is *older* than 9.2, replace `lmgrd.exe`, `lmtools.exe` and `lmutil.exe` with their corresponding newer versions provided with SAMG. In addition, put `FHGSCAI.exe` into the same directory in which `lmgrd.exe` resides.
 - Otherwise, just put `FHGSCAI.exe` into the same directory in which `lmgrd.exe` resides.

Now put the license file into a directory of (only) this machine. Open the license file with a text editor and, if necessary, replace "hostname" by the correct hostname of this machine.

2.2.3 Starting and testing the FLEXlm server for SAMG

There are various possibilities to use FLEXlm. The following services-based possibility will usually do:

- Start `lmtools.exe`.
- Go to "Config Services" and create a new service. Specify path+name of `lmgrd.exe` and the SAMG license file, and specify path+name of a logfile.
- Go to "Service/License File", choose "Configuration using Services", then choose the new service.
- Go to "Start/Stop/Reread" and start the new service.
- To test the SAMG-feature: Go to "Server Diags", specify "samg" as feature name and press the "Perform Diagnostics" button. FLEXlm should tell you that "this license can be checked out".
- Note that, if you have replaced an old FLEXlm by a new one (see above), it might be necessary to specify, for instance, services also for your other FLEXlm-based licenses.

2.2.4 Getting access to the FLEXlm server for SAMG from other machines

There is **no** (!) need to copy the license file to all machines in your LAN. Instead, on all machines in your LAN where you want to use SAMG, create the environment variable `LM_LICENSE_FILE` (if not already existing) and let it point to (or include) the hostname of the SAMG-FLEXlm license server, e.g. by issuing (for a console-based SAMG application prior to its call),

```
set LM_LICENSE_FILE=@mycomputer
```

where the hostname is assumed to be "`mycomputer`". For permanent use, do not forget to add this environment variable to Windows' list of environment variables.

3 Binary License for Linux/Unix

3.1 The SAMG library

3.1.1 Contents of the SAMG archive

The `samgXXX` archive provided contains the following files:

- `libamgYYY.so`: A *shared* SAMG library⁵ compiled with a Fortran90 compiler. Here `YYY` characterizes various aspects, for instance, SAMG's version number, the compiler used, etc.
- `demo.f90`: Fortran90 demo driver to be linked with the SAMG library.
- `democ.cpp`, `samg.h`: C++ demo driver and necessary header file to be linked with the SAMG library.
- A `makefile` for compiling and linking the Fortran90 and C++ demo using a Fortran90 compiler and "g++", respectively. We call the executables "`demo`" and "`democ`", respectively.
- `demo.frm`, `demo.amg`, `demo.rhs` and `demo.sol`: A set of data files describing a small linear system of equations (matrix, right hand side and solution). These files are read by the demo drivers which then solve the corresponding linear system. All files have to be located in the directory from which the drivers are launched.
- `out_democ`: Output of `democ`. This should virtually be the same as the output of `demo`.

3.1.2 Installation/Activation of the SAMG license key

Assuming that your SAMG installation requires a license key, be sure that you unzip the archive to the machine(s) for which you do have a valid license. Then proceed as follows:

- In case of **counted** and/or **floating licenses**, install and start a FLEXlm server as described in the corresponding installation notes, see Section 3.2.
- In case of a **node-locked license**, copy the license file provided to the corresponding machine. Let the environment variable `LM_LICENSE_FILE` point to (or include) this license file, e.g., by issuing `setenv LM_LICENSE_FILE /home/myself/samg/license.dat` (assuming a **(t)cs**h shell) prior to calling a SAMG application. This assumes that `LM_LICENSE_FILE` is a new environment variable, and that the license file is named `license.dat`, located in the directory `/home/myself/samg`. For permanent use, do not forget to add the above (or an analogous) statement to `.cshrc` or a corresponding file.

3.1.3 First steps towards integrating SAMG

- Start with modifying the `makefile` according to your compilation environment and build the demo executables `demo` and/or `democ` by issuing `make`. The demo drivers `demo.f90` and `democ.cpp` are similar so that it is sufficient to build only one of these demos. Make sure that the "." directory is in your `LD_LIBRARY_PATH`, e.g. by issuing `setenv LD_LIBRARY_PATH .:$LD_LIBRARY_PATH` (assuming a **(t)cs**h shell and an already existing `LD_LIBRARY_PATH`) prior to calling a SAMG application. For permanent use, do not forget to add the above (or an analogous) statement to `.cshrc` or a corresponding file.
- You may take `demo.f90` and/or `democ.cpp` as examples of how to call SAMG routines from within your own code, and of how to set parameters controlling different aspects of SAMG⁶.
- **ATTENTION**: Please study the `makefile` provided prior to linking SAMG to your own Fortran90 and/or C++ code. In case of linking to C(++), please study also `samg.h`.

⁵ Unless otherwise requested, we provide a *shared* lib. If, for important reasons, a *static* lib is needed, just let us know.

⁶ For a detailed description of all parameters and their options, see the SAMG User's Manual.

3.2 The FLEXIm server

Below we describe steps to get the FLEXIm server started for SAMG. For more information on FLEXIm and its operation, we refer to the **FLEXIm User's Guide** which is provided as part of SAMG's FLEXIm archive.

3.2.1 Contents of SAMG's FLEXIm archive

The `flexlm_enduser` archive contains the following files:

- `lmgrd`: the FLEXIm daemon, version 9.2.1;
- `FHGSCAI`: our vendor daemon; requires FLEXIm v.9.2.1 or newer;
- `lmutil`: a tool for starting, stopping, monitoring the license server etc.;
- `flexlm_enduser_guide.pdf`: the original FLEXIm End Users Guide.

3.2.2 Installation of the necessary files to run FLEXIm for SAMG

Locate the machine which is intended to be the SAMG-FLEXIm license server (according to the data you specified to us). Depending on whether or not FLEXIm is already installed on that machine, do the following:

- If FLEXIm is *not* yet installed, unpack the `flexlm_enduser` archive to a directory of this machine.
- If FLEXIm is already installed, use `lmutil` to find out its version number by issuing

```
lmutil lmver lmgrd
```

Then do the following:

- If the version is *older* than 9.2.1, replace `lmgrd.exe` and `lmutil.exe` by the corresponding newer versions provided with SAMG and put `FHGSCAI` into the same directory in which `lmgrd` resides.
- Otherwise, just put `FHGSCAI` into the same directory in which `lmgrd` resides.

Now put the license file provided - called ***yourlicensefile*** in the following - into a directory of this machine. Open the license file with a text editor and, if necessary, replace "hostname" with the correct hostname of this machine. To be specific in the following, we assume this hostname to be ***machine1.mydomain.com***.

3.2.3 Starting and testing the FLEXIm server for SAMG

There are various possibilities to use FLEXIm. The following possibility will usually do:

- Go to the directory where `lmgrd` and `FHGSCAI` are located.
- To start the license server, issue `lmgrd -c yourlicensefile >& flex.log`. This assumes that you use a **(t)cs**h shell, that ***yourlicensefile*** is located in the same directory as `lmgrd` and `FHGSCAI`, and that the FLEXIm logfile shall be called ***flex.log***.
- The last line in the logfile ***flex.log*** should then resemble

```
9:52:57 (FHGSCAI) Server started on machine1.mydomain.com for: samg
```

3.2.4 Getting access to the FLEXIm server for SAMG from other machines

There is **no** need to copy the license file to all machines in your LAN. Instead, on all machines in your LAN where you want to use SAMG, create the environment variable `LM_LICENSE_FILE` (if not already existing) and let it point to (include) the hostname of the SAMG-FLEXIm license server, e.g. by issuing,

```
setenv LM_LICENSE_FILE @machine1.mydomain.com
```

(assuming a **(t)cs**h shell) prior to calling a SAMG application. In order to check whether or not the license server can be accessed, start one of the demo executables, `demo` or `democ`.

4 Source License

4.1 Installation notes

After having unpacked the SAMG archive provided, build the SAMG libraries via one of the following tools:

- the **(t)cs**h shell script [amgconfig_user](#)⁷ in case of **Linux/Unix** or
- the batch procedure [amgconfig_user.bat](#)⁸ in case of **Windows**.

However, before using these tools, you will probably need to **modify/adjust the relevant include files** in the directory **includes** according to your system environment. Note that the suffix of a concrete include file indicates the machine/compiler it is to be used for. You can easily add your own include files.

If everything is properly adjusted, just follow the instructions appearing when running the tools. If the tools do not work in connection with your particular environment, just ask us for further assistance.

Remark: If you want to use an environment such as **Intel Visual Fortran**, refer to Section 4.3 below.

4.2 Integration of SAMG source code into your code/project

Before integrating the SAMG source files into your code/project, note the following:

4.2.1 User-adjustable routines: files [amguser.f](#) and [amguser_nocoo.f](#)

The file [amguser.f](#) contains a set of routines some of which a user might wish to adjust to his personal needs. These routines address the following issues:

- message output handling,
- time measurement,
- current memory status,
- user-defined checks,
- interfacing a user-provided coarsest-level solver,
- providing coordinates (**variable-to-coordinates mapping**) to SAMG.

For more information on the individual routines, see the self-explanatory comments in [amguser.f](#). In most cases, there will be no need to modify [amguser.f](#).

A typical case where a user might wish to modify [amguser.f](#) is when **coordinates-based SAMG options** are to be exploited. In such cases, the dummy version of the routine [samg_user_coo](#) provided in [amguser.f](#) needs to be replaced by a specific routine describing the variable-to-coordinates-mapping of the current application. Clearly, it is not convenient to modify [amguser.f](#) accordingly whenever the geometry of an application changes. It is more convenient to use [amguser_nocoo.f](#) instead of [amguser.f](#) (these two files are identical except that [samg_user_coo](#) is missing in [amguser_nocoo.f](#)) and include a specific [samg_user_coo](#) routine as part of the user's program environment.

4.2.2 Dummy routines: [dummy_omp.f](#)

A file [dummy_omp.f](#) (or an analogous file/library) is needed to build an SAMG library **without** OpenMP.

⁷ Before you can use the **(t)cs**h shell script [amgconfig_user](#), you may need to issue "`chmod u+x amgconfig_user`".

⁸ For running [amgconfig_user.bat](#), the auxiliary program [ask.exe](#) is needed. For completeness, this program is provided as part of the SAMG archive for Windows.

4.3 Using Intel Visual Fortran, Microsoft .NET and the like

If you want to use a visual development environment such as **Microsoft .NET**, **Intel Visual Fortran** or **Compaq Visual Fortran** for assembling SAMG (rather than the tools mentioned above), note that the following files need to be included in your Fortran90 library project:

amg.f	amgparms.f	dummy_omp.f*	mod_t.f
amg_mod.f	amgparms_user.f	modbfitv.f	opt_a.f
amgaux.f	amgset.f	modl2prj.f	opt_f.f
amgcl.f	amgsmo.f	modlsq.f	opt_k.f
amgint1.f	amgsol.f	modpwuse.f	opt_t.f
amgint2.f	amgsplit.f	modtvec.f	pwaccess.f
amgint3.f	amgstrng.f	mod_a.f	
amgmem.f	amguser.f or	mod_f.f	
amgopt.f	amguser_nocoo.f	mod_k.f	

*) Dummy routines needed to build the SAMG library **without** OpenMP

Note also that all Fortran90 sources are in free format. That is, before you can compile the above ".f" files, you need to turn on the **free format option**. Alternatively, you might replace the files' suffix ".f" by ".f90". Note, however, that ".f90" is not allowed on (old) IBM AIX (which is actually the reason why we stick to ".f").

4.4 Further important remarks

4.4.1 C(++) - to – Fortran90 calling conventions

Please refer to the provided header file [samg.h](#) for **C(++) - to – Fortran90 calling conventions**. In particular, be aware of the fact that on Windows **Compaq Visual Fortran** and **Intel Visual Fortran** use different calling conventions (`__stdcall` and `__cdecl`, respectively). See also Section 1.2.

4.4.2 Known problems with OpenMP

Note that only very recent versions of Intel 8.x (for Linux and Windows!) can (nearly?!) correctly handle **OpenMP!** If you want to compile with OpenMP, use the respective option

- `/Qauto_scalar` in case of **Windows** and
- `-auto_scalar` in case of **Linux**.

This overwrites the actual default OpenMP settings for these compilers.

Depending on the concrete configuration of your multi-processor machine, it might be necessary to experiment with the `stacksize`. Issuing `limit stacksize unlimited` (or a similar statement) might help.

4.4.3 Known problems with compilers

All compilers tested so far are able to compile SAMG correctly if proper compiler/linker options are specified (see, for instance, the include files provided). However, sometimes it is not easy to find proper options, and some compilers have bugs, in particular, in connection with OpenMP. Please inform us if, for example, a compiler reports an "internal compiler error" (i.e. most probably a bug of the respective compiler) or if any other unexpected behavior occurs. This is very important information for us. If we can reproduce the problem, we will try to find a workaround and/or contact the respective compiler vendor.