

Gaussian

Contents

- [Description](#)
- [Getting started with Gaussian 03: Setting up your UNIX account](#)
- [Running Gaussian](#)
- [Gaussian documentation](#)
- [Gaussian utility programs](#)
- [Where to find Gaussian](#)
- [If you still need help](#)

Description

Gaussian 03 is generally referred to as an *ab initio* electronic structure program. It predicts the energies, molecular structures, and vibrational frequencies of molecular systems, along with molecular properties derived from these basic computation types.

This document provides basic instructions on running Gaussian and configuring your account on wk*.cos.gmu.edu (e.g. wk01.cos.gmu.edu, a.k.a Turtle). It is not a Gaussian tutorial. For more complete information, please consult the online http://www.gaussian.com/g_ur/g03mantop.htm

Getting started with Gaussian: Setting up your UNIX account

Gaussian is installed in the directory /usr/local/g2003 on Turtle. Throughout this document, the installation directory is referred to as g03root.

You must configure your UNIX account to use Gaussian on Turtle. You should define a set of environment variables and *source* a file provided by Gaussian to configure your account properly.

For tcsh or csh shell users, use

```
% setenv g03root /usr/local/g2003
% source $g03root/g03/bsd/g03.login
```

For bash or sh shell users, use

```
$ export g03root=/usr/local/g2003
$ source $g03root/g03/bsd/g03.profile
```

Running Gaussian

Before running Gaussian, you must properly configure your UNIX account on Turtle. The configuration instructions are in the previous section.

- **Starting a Gaussian job**
Many Gaussian jobs may take weeks to complete. Since Turtle is a shared system, we request that you start your Gaussian job using the `nice` command. For example,

```
nice g03 input_data_file &
```

starts a job having a Gaussian input data file named *input_data_file*.

Output files are created in the directory containing the input data file. Therefore, it is good practice to create a separate directory for each input data file and to `cd` to that directory before running Gaussian. If any output file (including a checkpoint file) already exists when Gaussian tries to create it, Gaussian will overwrite it.

A sample input data file for Gaussian might be named `test000.com` and contain the following lines.

```
# SP, RHF/STO-3G punch=archive trakio scf=conventional

Gaussian Test Job 00
Water with archiving

0 1
O
H 1 0.96
H 1 0.96 2 109.471221
```

When you run Gaussian, the system displays your job's main *process ID*. You can use the process ID to track your job and the work files that Gaussian generates. For example,

```
% nice g03 test000 &
[1] 2418
```

shows the main process ID associated with this Gaussian job is 2418.

- **Using multiple CPUs**

Currently, we do not run parallel jobs.

- **Gaussian temporary files**

Gaussian's temporary work files are stored in the directory specified by the value of the `$GAUSS_SCRDIR` environment variable, for example, `/Users/jdoe/scratch`. The extension on each filename correspond to the file's purpose. Each name starts with the same stem, `Gau-ProcessID`, based on the process ID of the controlling job. The initial Gaussian process may also spawn other processes with their own related process IDs.

The temporary files are stored in the scratch directory `/Users/jdoe/scratch`. Note the process IDs used in their names.

```
% ls /Users/jdoe/scratch
Gau-2418.inp  Gau-2422.d2e  Gau-2422.rwf
Gau-2422.chk  Gau-2422.int  Gau-2422.scr
```

The purpose of these temporary Gaussian files are noted below.

- Checkpoint file: `Gau-processID.chk`
- Read-Write file: `Gau-processID.rwf`

- Two-Electron Integral file: `Gau-processID.int`
 - Two-Electron Integral Derivation file: `Gau-processID.d2e`
 - Internal input file: `Gau-ProcessID.inp`
 - Internal scratch file: `Gau-ProcessID.scr`
- **Managing Checkpoint files**

Gaussian delete temporary files automatically when the job is successful. However, you may want to save the checkpoint file (`.chk`) to use later for another Gaussian job, to collaborate with another researcher, for use by a visualization program, or to restart a failed job. To accomplish this, specify a file or directory name for the checkpoint file by putting a `%Chk` command at the beginning of your Gaussian input data file. For example, if `$GAUSS_SCRDIR` is defined as `/Users/jdoe/scratch`, then setting

```
%Chk=/scratch/jdoe/test000
```

will create the checkpoint file `/scratch/jdoe/test000.chk`. It will not be removed even if the Gaussian job is successful.

- **Checking Your Gaussian Job**

You can check on the status of your job by viewing the contents of the `input_data_file.log` file.

If the Gaussian job is completed successfully, then the `.log` file will contain a random quotation and indicate normal termination. For example, the log file `test000.log`, resulting from a successful run, might look like the following.

```
The archive entry for this job was punched.
```

```
The best way to pay for a lovely moment is to enjoy it.
-- Richard Bach
Job cpu time:  0 days  0 hours  0 minutes  4.4 seconds.
File lengths (MBytes):  RWF=      13 Int=      2 D2E=      0 Chk=
9 Scr=      1
Normal termination of Gaussian 03 at Thu Jun  7 15:45:38 2007.
```

Gaussian documentation

Gaussian documentation can be viewed online at the Gaussian Inc. website.

- http://www.gaussian.com/g_ur/g03mantop.htm

After [configuring your UNIX account for Gaussian use](#), you can also get general information and a list of help topics by typing the following command on Turtle

```
% ghelp
```

Sample input data files may be found in `$g03root/g03/tests/com/` on Turtle.

Gaussian utility programs

The following table lists the function and names of Gaussian's utility programs.

<u>c8603</u>	Converts checkpoint files from previous program versions to <i>Gaussian 03</i> format.
<u>chkchk</u>	Displays the route and title sections from a checkpoint file.
<u>cubegen</u>	Standalone cube generation utility.
<u>cubman</u>	Manipulates <i>Gaussian</i> -produced cubes of electron density and electrostatic potential (allowing them to be added, subtracted, etc.).
<u>formchk</u>	Converts a binary checkpoint file into an ASCII form suitable for use with visualization programs and for moving checkpoint files between different types of computer systems.
<u>freqchk</u>	Prints frequency and thermochemistry data from a checkpoint file. Alternate isotopes, temperature, pressure and scale factor can be specified for the thermochemistry analysis.
<u>freqmem</u>	Determines memory requirements for frequency calculations.
<u>gauopt</u>	Performs optimizations of variables other than molecular coordinates.
<u>ghelp</u>	On-line help for <i>Gaussian</i> .
<u>mm</u>	Standalone molecular mechanics program.
<u>newzmat</u>	Converts between various molecular geometry specification formats.
<u>testrt</u>	Route section syntax checker and non-standard route generation.
<u>unfchk</u>	Converts a formatted checkpoint file back to its binary form (e.g., after moving it from a different type of computer system).