teaberry

A transient bond model simulator
version 0.1.3
13 January 2020

Takashi Uneyama

Table of Contents

1 Introduction

In some soft matter systems, dynamics of particles is strongly constrained and characteristic relaxation behavior is observed. A widely known example is the entangled polymer systems. In entangled polymer systems, due to the uncrossability between polymer chains, the motion of chains is strongly constrained.

One way to model such dynamical constraints is to introduce some transient bonds to the system. The responsive particle dynamics (RaPiD) model [Kindt-Briels-2007, Briels-2009, Briels-2015] employs the numbers of entanglements between chains as extra thermodynamic degrees of freedom, in addition to the particle positions. The slip-spring model for entangled polymers The slip-spring models [Likhtman-2005, Uneyama-2011, Uneyama-Masubuchi-2012] also utilize transient bonds (slip-springs) to mimic the entanglement effect. By unifying the RaPiD model and the slip-spring model, we can construct a highly coarse-grained transient bond model [Uneyama-2019]. We can perform dynamical simulations with transient bonds, with well-defined effective (pseudo) free energy.

`teaberry` is a package for the transient bond model simulations. It contains the simulator and utilities for the transient bond model. The `teaberry` simulator can simulate one, two and three dimensional systems with transient bonds. The diffusion and linear viscoelastic behavior can be analyzed by utility tools from the simulation data.

2 Compilation of teaberry

2.1 To Download the Latest Version of teaberry

The latest version of teaberry is available at the following URL. Access the web page and download the latest version via HTTP (FTP is not supported).

http://polymer-physics.jp/uneyama/teaberry.html

2.2 Build from the Source

You need to build teaberry from the source package to use it. The source package of teaberry is using GNU Automake and GNU Autoconf, therefore you can build and install teaberry just like usual free software. Note that teaberry requires zlib (http://www.zlib.net/) and Lua (http://www.lua.org/). You have to install them before building teaberry.

The HDF5 file is supproted by teaberry. To enable the HDF5 support, you should install HDF5 https://www.hdfgroup.org/HDF5/ before building teaberry.

The source package is distributed as a gzipped tar archive file, thus first you need to extract it. To extract the archive, do

```
$ zcat teaberry-0.1.3.tar.gz | tar xvf -
```

or if you are using GNU tar, do

```
$ tar zxvf teaberry-0.1.3.tar.gz
```

Then the source directory will extracted. Move to the directory teaberry-0.1.3.

```
$ cd teaberry-0.1.3
```

To build teaberry, do configure-make just like other free software packages.

```
$ ./configure
$ make
```

Now library files for simulator cores and simple simulator programs using teaberry will be built and stored under src/ and examples/ subdirectories. If you have an error messages and the compilation is aborted, some commands or libraries may be missing. Install the required packages and retry.

If you want to customize or tune teaberry,

```
$ ./configure --help
```

will help you.

3 Invoking teaberry

The format for running the teberry program is:

```
$ teberry option ... input
```

input is the input file for teberry. If no input file is specified, teberry will read the default input file teberryin.lua.

teberry supports the following options:

--input=*input*
-i *input* Read the parameters for simulation from the input file *input*. If no input file is specified, teberry will read the input file named teberryin.lua.

--state=*state*
-s *state* Read the initial state from the HDF5 file *state*. The state file *position* must be the HDF5 file generated by teaberry or the HDF file which has exactly the same format. By default, teberry samples the initial state from the equilibrium distribution.

--help
-h Show summary of options.

--version
-v Show version of program.

To use teaberry, you should prepare the input file. See the examples for details.

4 Formats of Input and Output Files

4.1 Input File

The input file for `teaberry` is a Lua script. The input file sets simulation parameters including the output files. The state file can be also used as the additional input file. If the state file is specified, the simulator loads the state from the file. The state file is the HDF5 file and details are explained in the output file section.

A simple example input Lua file for a three dimensional simulation is as follows:

```
save_position_sequential = true
save_dx_sequential = true
save_bond_sequential = true
save_stress_sequential = true
seed = 831071

interval = 50
iteration_max = interval * 100

n = 8 * 8 * 8

dt = 0.02
xi = 1.0
tau = 100.0

dimension = 3
l = {8, 8, 8}
```

The simulation is performed for a three dimensional box of which size is 8 times 8 times 8, and the particle density is 1 (the number of particles in the systems is 8 times 8 times 8). The time step size is 0.02 and the data will be sequentially output by the interval 1 (50 times 0.02). The simulation starts from the equilibrium initial state and runs up to time 100. The strength and life time of the transient bonds will be controlled by the parameters `xi` (= 1) and `tau` (= 100).

The details of the input parameters are listed below. Some parameters have their default values, and if they are not specified, the default values will be substituted.

`save_position_sequential`
: (*boolean*, default: false)

  Whether the particle postion data should be saved sequentially or not. If this is set to be true, the sequential file names will be generated by `position_template`. If this is not set to be true, the position data will be output to `position_output`, and the old data will be overwritten.

`save_dx_sequential`
: (*boolean*, default: false)

  Whether the OpenDX visualization data should be saved sequentially or not. If this is set to be true, the sequential file names will be generated by `dx_`

`template`. If this is not set to be true, the OpenDX data will be output to `dx_output`, and the old data will be overwritten.

`save_bond_sequential`
(*boolean*, default: false)

Whether the transient bond data should be saved sequentially or not. If this is set to be true, the sequential file names will be generated by `bond_template`. If this is not set to be true, the bond data will be output to `bond_output`, and the old data will be overwritten.

`save_stress_sequential`
(*boolean*, default: false)

Whether the stress tensor data should be saved sequentially or not.

`save_energy_sequential`
(*boolean*, default: false)

Whether the energy data should be saved sequentially or not.

`save_geometry_sequential`
(*boolean*, default: false)

Whether the box geometry data should be saved sequentially or not.

`seed` (*number*)

The random seed for the Mersenne Twister random number generator.

`iteration_step`
(*number*, default: 0)

The initial step number.

`apply_shear`
(*boolean*, default: false)

Whether the shear flow is applied to the system or not. If this is set to be true, the shear rate with `kappa` and angular frequency with `omega` will be applied to the system. This cannot be utilized for the one dimensional system.

`omega` (*number*, default: 0)

The angular frequency of the shear flow. 0 means the steady shear.

`kappa` (*number*, default: 0)

The shear rate. For steady shear, this value is nothing but the shear rate. For oscillatory shear, this value is the maximum shear rate. The shear amplitude is given by `kappa / omega`.

`interval` (*number*)

The interval for the data output. The data files will be output every `interval` steps.

`iteration_max`
(*number*)

The maximum number of iterations in the simulation.

`n` (*number*)
The number of particles in the simulation box.

`dt` (*number*)
The time step size.

`xi` (*number*)
The bond fugacity. This parameter controls the equilibrium number of bonds in the system.

`tau` (*number*)
The bond life time. This parameter controls the reconstruction of bonds.

`dimension`
(*number*)
The dimension of the simulation box. This should be set to be 1, 2 or 3.

`l` (*array of numbers*)
The simulation box size. This array should contain `dimension` values.

`position_output`
(*string*, default: "position.dat.gz")
The output file name for particle positions.

`bond_output`
(*string*, default: "bond.dat.gz")
The output file name for transient bonds.

`stress_output`
(*string*, default: "stress.dat")
The output file name for the stress tensor.

`energy_output`
(*string*, default: "energy.dat")
The output file name for the energy.

`geometry_output`
(*string*, default: "geometry.dat")
The output file name for the box geometry.

`dx_output`
(*string*, default: "teaberryout.dx")
The output file name for the OpenDX visualization data. The OpenDX format is supported only for two and three dimensional systems.

`h5_output`
(*string*, default: "teaberrystate.h5")
The output file name for the HDF5 state data.

`position_template`
(*string*, default: "position.%d.dat.gz")
The output file name template for particle positions. "%d" will be replaced by the number.

`bond_template`
: (*string*, default: `"bond.%d.dat.gz"`)

  The output file name template for transient bonds. "%d" will be replaced by the number.

`dx_template`
: (*string*, default: `"teaberryout.%d.dx"`)

  The output file name template for OpenDX visualization data. "%d" will be replaced by the number. The OpenDX format is supported only for two and three dimensional systems.

4.2 Output Files

The simulator `teaberry` outputs several data files with different formats. The data formats are briefly explained here.

The particle positions in two or three dimensional systems are stored in the position output file. The example is as follows:

```
# 3 512
5.321799 3.163433 3.117007 0 1 0
3.819845 6.562836 6.264025 0 0 0
4.672288 1.792827 6.872268 0 0 0
4.997906 6.038286 5.756395 0 -1 0
6.041774 6.178037 3.480612 0 0 0
    :        :        :
```

The first line shows the dimension and the number of particles. Then the particle positions follows. The particle position consists of the position in the box and the box indices. The data file is gzipped, thus use `zcat` or `gunzip` to use the file for other applications.

The transient bonds are stored in the bond output file. The example is as follows:

```
# 768
241 292
176 193
433 287
188 306
274 439
 :   :
```

The first line shows the total number of bonds, and the bond data follows. One bond is specified by two indices, to which the bond is attached. The data file is gzipped, thus use `zcat` or `gunzip` to use the file for other applications.

The stress tensor is stored in the stress output file. For example, we have the data as:

```
-0.00826368 0.0426762 0.0628783 -0.0418648 -0.0419426 0.0574673
-0.0289722 0.246913 -0.0906975 -0.0346256 -0.0173753 -0.0148011
-0.00820888 0.0429851 -0.140312 0.0467222 0.00715025 0.107648
    :        :         :          :         :          :
```

In one dimensional systems, a single line contains only a single value, which corresponds to the `xx` component of the stress tensor. In two dimensional systems, a single line contains

the `xx`, `yy`, and `xy` components of the stress tensor. In three dimensional systems, a single line contains `xx`, `yy`, `zz`, `xy`, `yz`, and `zx` components. If the stress is not sequentially output, the file contains only one line, which shows the latest stress data.

The energy of the system is stored in the energy output file. For example, we have the data as:

```
2.245837 1.484333
   :        :
```

The first value is the bond energy and the second value is the site energy. The energy is divided by the system volume so that it it becomes the energy density. The bond energy is calculated for all the bonds in the system, and the site energy is calculated for all the particle pairs which are both bonded and non-bonded. If the stress is not sequentially output, the file contains only one line, which shows the latest energy data.

The box geometry of the system is stored in the geometry output file. For example, we have the data as:

```
3
8.000000 8.000000 8.000000
0 0
:
```

The first line shows the dimension of the system, and the second line shows the box lengths. The third line shows the displacement of the box by the shear. If the geometry is not sequentially output, the file contains only one line, which shows the latest geometry data.

The OpenDX visulaization data file contains the data which are required for most of visualizations. The objects in the OpenDX data file is listed below:

`box` (*field*)
: The simulation box.

`delta_l` (*array of float*)
: The displacement of the box by the shear flow.

`particles` (*field*)
: The particle data. This field has three component; `id`, `data`, and `positions`. `id` is the particle id, which starts from 0. `data` is the number of bonds attached to the target particle. `positions` is the particle positions.

`bonds` (*field*)
: The transient bond data. This field has two component; `positions` and `connections`. `positions` is the end point positions of bonds. `connections` is the connectivity information for the end point positions.

The HDF5 state file contains the state data of the simulator. Use utilities for HDF5 files to use and manipulate the data. The groups and datasets are listed below:

`/` (*group*)
: The root group.

`/bonded_particles`
(*group*)
This group contains the state of particles and bonds.

`/bonded_particles/c`
(*dataset*, *I32LE*)
The cell indices for particles.

`/bonded_particles/dimension`
(*dataset*, *I32LE*)
The dimension of the current simulation.

`/bonded_particles/m`
(*dataset*, *I32LE*)
The number of transient bonds.

`/bonded_particles/n`
(*dataset*, *I32LE*)
The number of particles.

`/bonded_particles/n`
(*dataset*, *F64LE*)
The particle positions.

`/bonded_particles/s`
(*dataset*, *I32LE*)
The bond connectivity information.

`/box_geometry`
(*group*)
This group contains the state of the simulation box and its geometry.

`/box_geometry/c`
(*dataset*, *I32LE*)
The cell index of the box.

`/box_geometry/delta_l`
(*dataset*, *F64LE*)
The displacement of the box by the shear flow.

`/box_geometry/dimension`
(*dataset*, *I32LE*)
The dimension of the system.

`/box_geometry/l`
(*dataset*, *F64LE*)
The lengths of the simulation box.

`/box_geometry/theta`
(*dataset*, *F64LE*)
The current phase of the oscillatory shear.

`/box_geometry`
(*group*)
This group contains the state of the pseudo random number generator.

`/rng_state/mt_state/`
(*dataset*, *I32LE*)
This group contains the state of the Mersenne-Twister pseudo random number generator.

`/rng_state/mt_state/initf`
(*dataset*, *I32LE*)
"initf" variable for the Mersenne-Twister.

`/rng_state/mt_state/left`
(*dataset*, *I32LE*)
"left" variable for the Mersenne-Twister.

`/rng_state/mt_state/n`
(*dataset*, *I32LE*)
the size of the state space for the Mersenne-Twister.

`/rng_state/mt_state/next`
(*dataset*, *I32LE*)
"next" variable for the Mersenne-Twister.

`/rng_state/mt_state/state`
(*dataset*, *U64LE*)
The current state of the Mersenne-Twister.

`/rng_state/nrand_has_saved_number`
(*dataset*, *I32LE*)
Whether the normal random number generator has the stored random number or not.

`/rng_state/nrand_w`
(*dataset*, *F64LE*)
The normal random number.

5 Utility Programs

To support the analysis, some utility programs are provided with `teaberry`. You can use these utility programs to analyze the simulation data. (Of course, you can analyze the output data of `teaberry` by your own programs.) Currently, three utility programs are provided. The details are described below.

5.1 Relaxation Modulus

The shear relaxation modulus of a system can be calculated from the stress autocorrelation function via the Green-Kubo formula. `teaberry-stress2modulus` calculates the shear relaxation modulus. The shear relaxation modulus can be calculated only for two and three dimensional systems. For one dimensional systems, the bulk relaxation modulus is calculated instead.

To calculate the shear (or bulk) relaxation modulus, execute `teaberry-stress2modulus` as

```
$ teberry-stress2modulus input > output
```

where *input* is the stress data file (by default, it is "stress.dat"). `teaberry-stress2modulus` calculate the autocorrelation function and print the calculated data to stdout. In the example shown above, the output data are redirected to *output*.

After you obtain *output*, rescale the time and modulus scales to obtain the correct shear (or bulk) modulus data. The time unit is the step size, and the modulus unit is the system volume.

5.2 Trajectory, Mean-Square Displaecement, and Mean-Quartic Displacement

To convert the particle position data files to trajectroy data files, use `teaberry-position2trajectory` as:

```
$ teberry-position2trajectory first last geometry position trajectory
```

where *first* and *last* are the first and last steps for the conversion, *geometry* is the geometry file, *position* is the template for the input position data files, and *trajectory* is the template for the output trajectory data files.

For example, if you have the position data from "position.0.dat.gz" to "position.100.dat.gz", and the geometry data "geometry.dat", execute as

```
$ teaberry-stress2modulus 0 100 geometry.dat \
  position.%d.dat.gz trajectory.%d.dat.gz
```

The trajectory of each particle will be separately saved in the files "trajectory.0.dat.gz", "trajectory.1.dat.gz", ...

After the position data files are converted to the trajectory data files, another utility program `teaberry-trajectory2msd` can be used to obtain the mean-square displamcenet data. This can be executed as:

```
$ teaberry-trajectory2msd first last delta trajectory msd
```

where *first* and *last* are the first and last indices for the trajectory data, *delta* is the time lag size, *trajectory* is the template for the input trajectory data files, and *msd* is

the output data file. If *delta* is greater than zero, `teaberry-trajectory2msd` calculates the time-averaged mean-square displacement and its relative standard deviaton [Uneyama-Miyaguchi-Akimoto-2015]. If *delta* is zero, `teaberry-trajectory2msd` simply calculates the ensemble averaged mean-square displacement. Therefore, to obtain the simple ensemble averaged mean-square displacement for the system with 1024 particles (the index is from 0 to 1023, and thus the trajectory data files are from "trajectory.0.dat.gz" to "trajectory.1023.dat.gz"), execute the program as

```
$ teaberry-trajectory2msd 0 1023 0 trajectory.%d.dat.gz msd.dat
```

The mean-quartic displacement data can be calculated in a similar way, by utilizing `teaberry-trajectory2mqd`. This can be executed as

```
$ teaberry-trajectory2msd first last trajectory mqd
```

The parameters are almost the same for the case of `teaberry-trajectory2msd`, except that we do not need to specify *delta* and the output file is *mqd*. Only the ensemble averaged mean-quartic displacement is calculated by this utility. The non-Gaussianity parameter can be calculated by combining the mean-square and mean-quartic displacements. To obtain the simple ensemble averaged mean-quartic displacement for the system with 1024 particles (the index is from 0 to 1023, and thus the trajectory data files are from "trajectory.0.dat.gz" to "trajectory.1023.dat.gz"), execute the program as

```
$ teaberry-trajectory2mqd 0 1023 trajectory.%d.dat.gz mqd.dat
```

6 Reporting Bugs

Currently, the error handling routines in `teaberry` is not complete and therefore `teaberry` may suddenly stops if some input error or calculation error is caused. Especially, parameters are not strictly checked and care should be paid.

If you find a bug in `teaberry`, please send electronic mail to uneyama@mp.pse.nagoya-u.ac.jp with the version number and simulator core name. Please describe the problem in detail, for example, how and/or when the program works wrong.

If you have other questions, comments or suggestions about `teaberry`, contact the author via electronic mail to uneyama@mp.pse.nagoya-u.ac.jp. The author will try to help you out, although he may not have time to fix your problems.

7 References

[Kindt-Briels-2007] P. Kindt and W. J. Briels, *J. Chem. Phys.* **127**, 134901 (2007).

[Briels-2009] W. J. Briels, *Soft Matter* **5**, 4401-4411 (2009).

[Briels-2015] W. J. Briels, "Responsive Particle Dynamics for Modeling Solvents on the Mesoscopic Scale" in *Computational Trends in Solvation and Transport in Liquids*, G. Sutmann, J. Grotendorst, G. Gompper and D. Marx eds. (Forschungszentrum Julich, 2015).

[Likhtman-2005] A. E. Likhtman, *Macromolecules* **38**, 6128 (2005).

[Uneyama-2011] T. Uneyama, *Nihon Reoroji Gakkaishi (J. Soc. Rheol. Jpn.)* **39**, 135 (2011).

[Uneyama-Masubuchi-2012] T. Uneyama and Y. Masubuchi, *J. Chem. Phys.* **135**, 184904 (2012).

[Uneyama-2019] T. Uneyama, *J. Chem. Phys.* **150**, 024901 (2019).

[Uneyama-Miyaguchi-Akimoto-2015] T. Uneyama, T. Miyaguchi, and T. Akimoto, *Phys. Rev. E* **92**, 032140 (2015).

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
   a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.
8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.
10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software

which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and a brief idea of what it does.
Copyright (C) yyyy  name of author

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type ‘show w’.
This is free software, and you are welcome to redistribute it
under certain conditions; type ‘show c’ for details.
```

The hypothetical commands ‘`show w`’ and ‘`show c`’ should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ‘`show w`’ and ‘`show c`’; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
‘Gnomovision’ (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

Concept Index