Umansysprop Documentation

Release 0.1

Dave Jones

Mar 05, 2018

Contents

UManSysProp is an online facility for calculating the properties of individual organic molecules and ensemble mixtures. Built using open source chemical informatics, the facilities provided are divided according to broad applications.

The code is licensed under the GPL v2 or above. Packages can be downloaded from PyPI. The source code can be obtained from GitHub, which also hosts the bug tracker. The documentation (which includes installation and quick start examples) can be read on ReadTheDocs.

CHAPTER 1

Table of Contents

1.1 Client Installation

The client component of UManSysProp can be installed on any machine with Python available. On Ubuntu, the waveform PPA can be used for simple installation:

```
$ sudo add-apt-repository ppa:waveform/ppa
$ sudo apt-get update
$ sudo apt-get install python-umansysprop
```

On other platforms, the package can be installed from PyPI. Specify the `client` option to pull in all dependencies required by the client component:

```
$ sudo pip install "umansysprop[client]"
```

1.2 Server Installation

The server component of UManSysProp is only tested on Linux platforms, although it should theoretically work on others. The application uses the WSGI architecture for communication with web-servers; integration with your web-server depends on understanding WSGI applications. The Flask deployment guide may be helpful in this case.

The server component can be installed from PyPI. Specify the `server` option to pull in all dependencies required by the server component:

```
$ sudo pip install "umansysprop[server]"
```

Please be aware that OpenBabel is a requirement of the server component. As this is SWIG based you will need a C/C++ build environment installed, along with the necessary Python and OpenBabel headers. The following command should suffice for this on Ubuntu:

```
$ sudo apt-get install build-essential python-dev libopenbabel-dev
```

1.3 Quick Start

It is strongly recommended that you use UManSysProp from within the IPython shell. The API is designed with documentation built-in which can be queried "live" from within the environment, and this is considerably easier from within the IPython shell. The rest of this guide will include tips for usage within IPython, but examples will be given in the syntax of the regular Python shell.

The first step in using the UManSysProp system is creating a *UManSysProp* instance. This requires the URL of the UManSysProp server which defaults to `http://umansysprop.seaes.manchester.ac.uk/`:

```
>>> import umansysprop.client
>>> client = umansysprop.client.UManSysProp()
```

Once you have a client instance, you can query it to find out what methods are available from the web API. Within the IPython shell this can be done simply by entering `client.` and pressing the Tab key twice. Alternatively, the following one-liner in the regular Python shell can be used to query non-private methods:

```
>>> [m for m in dir(client) if not m.startswith('_')]
['CCN_potential_inorg',
 'CCN_potential_inorg_org',
 'CCN_potential_org',
 'absorptive_partitioning',
 'absorptive_partitioning_no_ions',
 'activity_coefficient_inorg_org',
 'activity_coefficient_org',
 'critical_property',
 'hygroscopic_growth_factor_inorg',
 'hygroscopic_growth_factor_inorg_org',
 'hygroscopic_growth_factor_org',
 'sub_cooled_density',
 'vapour_pressure']
```

Once you've selected a method to call you can discover what parameters it takes and what it expects in those parameters by querying the method's documentation. Within the IPython shell this can be viewed simply by appending `?` to the method name. Alternatively, the `help()` function can be used in a regular Python shell:

```
>>> help(client.vapour_pressure)
Help on method vapour_pressure in module umansysprop.client:

vapour_pressure(self, compounds, temperatures, vp_method, bp_method)...
    Calculates vapour pressures for all specified *compounds* (given as a
    sequence of SMILES strings) at all given *temperatures* (a sequence of
    floating point values giving temperatures in degrees Kelvin). The
    *vp_method* parameter is one of the strings:

    * 'nannoolal'
    * 'myrdal_and_yalkowsky'
    * 'evaporation'
    ...
```

The various methods are not included within this documentation (which only covers the framework) simply because they are defined by the server API (not by this package). The documentation for each tool can viewed on the UManSysProp API documentation page.

Calling any of the methods will (in the event of success) return a *Result* instance. This is simply a `list()` which contains a sequence of *Table* instances. Each table has a name and this can be used to access the table in the owning *Result* list. For example:

```
>>> result = client.vapour_pressure(
... ['CCCC', 'C(CC(=O)O)C(=O)O', 'C(=O)(C(=O)O)O',
... 'CCCCC/C=C/C/C=C/CC/C=C/CCCC(=O)O'],
... [298.15, 299.15, 300.15, 310.15],
```

```
... vp_method='nannoolal', bp_method='nannoolal')
>>> result
[<Table name="pressures">]
>>> result.pressures
<Table name="pressures">
```

Table instances have a friendly string representation which can be used at the command line for quick evaluation of the contents:

```
>>> print(result.pressures)
       |           CCCC | C(CC(=O)O)C(=O)O | C(=O)(C(=O)O)O | CCCCC/C=C/C/C=C/CC/
↪C=C/CCCC(=O)O
-------+----------------+------------------+----------------+---------------------
↪-----------
298.15 | 0.220914923012 |   -6.33293991048 | -5.19636054531 |                   -9.
↪66033139516
299.15 | 0.235479319348 |   -6.28117761855 | -5.15170377256 |                   -9.
↪58901500825
300.15 | 0.249933657549 |   -6.22986499517 | -5.10742877511 |                   -9.
↪51835276669
310.15 | 0.388688301563 |   -5.74023509659 | -4.68464352888 |                   -8.
↪84581513627
```

The *Table* class also provides several attributes which can be used to access the data in a variety of common extension formats, specifically numpy ndarrays and pandas DataFrames:

```
>>> result.pressures.as_dataframe
Compound          CCCC  C(CC(=O)O)C(=O)O  C(=O)(C(=O)O)O  \
Temperature
298.15        0.220915         -6.332940       -5.196361
299.15        0.235479         -6.281178       -5.151704
300.15        0.249934         -6.229865       -5.107429
310.15        0.388688         -5.740235       -4.684644

Compound     CCCCC/C=C/C/C=C/CC/C=C/CCCC(=O)O
Temperature
298.15                              -9.660331
299.15                              -9.589015
300.15                              -9.518353
310.15                              -8.845815
```

1.4 API Reference

The API reference below is primarily geared towards those users that wish to use the UManSysProp client component.

1.4.1 `umansysprop.client` Module

This module contains the client library for interacting with the UManSysProp server. Only one user-accessible class is defined in the module:

UManSysProp

class umansysprop.client.**UManSysProp**(*base_url='http://umansysprop.seaes.manchester.ac.uk/'*)

Provides a simple Python interface to the methods provided via the JSON API on the UManSysProp website. Constructing an instance of this class will cause the new instance to query the server for all available

methods. Each method will be exposed as a method of the instance, with docstrings obtained from the server. For example:

```
>>> import umansysprop.client
>>> client = umansysprop.client.UManSysProp()
>>> client.vapour_pressure
<bound method ?.vapour_pressure of <umansysprop.client.UManSysProp object at␣
↪0x7f80fbae7b50>>
>>> client.vapour_pressure.__doc__
u"\nCalculates vapour pressures for all specified *compounds* (given..."
```

The class can be constructed with an alternative *base_url* if you wish to point it a different server. The *base_url* parameter defaults to the UManSysProp website.

Methods can be called like a normal Python method, but will result in a request being sent to the web-server, processed, and the JSON-formatted results being re-constructed as a *Result* list on the client. For example:

```
>>> result = client.vapour_pressure([
... 'CCCC', 'C(CC(=O)O)C(=O)O', 'C(=O)(C(=O)O)O',
... 'CCCCC/C=C/C/C=C/CC/C=C/CCCC(=O)O'],
... [298.15, 299.15, 300.15, 310.15],
... 'nannoolal', 'nannoolal')
>>> result
[<Table name="pressures">]
>>> result.pressures
<Table name="pressures">
>>> result.pressures.data
{(298.15, u'C(=O)(C(=O)O)O'): -5.196360545314141,
 (298.15, u'C(CC(=O)O)C(=O)O'): -6.33293991047814,
 (298.15, u'CCCC'): 0.22091492301164387,
 (298.15, u'CCCCC/C=C/C/C=C/CC/C=C/CCCC(=O)O'): -9.660331395164858,
 (299.15, u'C(=O)(C(=O)O)O'): -5.151703772558254,
 (299.15, u'C(CC(=O)O)C(=O)O'): -6.281177618554678,
 (299.15, u'CCCC'): 0.2354793193478599,
 (299.15, u'CCCCC/C=C/C/C=C/CC/C=C/CCCC(=O)O'): -9.58901500825095,
 (300.15, u'C(=O)(C(=O)O)O'): -5.107428775107059,
 (300.15, u'C(CC(=O)O)C(=O)O'): -6.229864995169396,
 (300.15, u'CCCC'): 0.24993365754879804,
 (300.15, u'CCCCC/C=C/C/C=C/CC/C=C/CCCC(=O)O'): -9.518352766693454,
 (310.15, u'C(=O)(C(=O)O)O'): -4.684643528880829,
 (310.15, u'C(CC(=O)O)C(=O)O'): -5.74023509658808,
 (310.15, u'CCCC'): 0.38868830156274703,
 (310.15, u'CCCCC/C=C/C/C=C/CC/C=C/CCCC(=O)O'): -8.845815136269506}
```

Please refer to the reference for *Result* and *Table* for more information on accessing the result data.

1.4.2 umansysprop.results Module

This module defines the classes used to encapsulate results returned by the UManSysProp server. Each tool method on the client will return an instance of the *Result* class which in turn contains one or more *Table* instances.

Result

class umansysprop.results.**Result**(**tables*)

Represents a list of named *Table* objects.

The result of a method is represented as a sequence of tables. This class contains a list of *Table* objects each of which may be retrieved by name, or by index in the list (tables with identical names are not ignored, but only the first table may be retrieved by name).

Note: This class has an extended string representation intended for easy command line debugging. Simply print an instance of the class to view a dump of all the tables contained within it.

Table

class umansysprop.results.**Table**(*name*, *rows*, *cols*, *func=None*, *data=None*, *title=''*, *rows_title=None*, *cols_title=None*, *rows_unit=None*, *cols_unit=None*)

Represents a single table in a *Result*.

A tool is expected to return a sequence of *Table* objects in a *Result* object. Each table has a *name* (which can be used to access it in the *Result* object), an ordered list of keys for *rows* and *cols*, and a function which is used to derive the data for each cell. The function accepts two arguments, the row and column key in that order, and is expected to return a scalar value. The reason for constructing a table in this manner (lazy evaluation) is that it enables renderers to query the table structure and layout without necessarily calculating anything. Calculated data is cached on the assumption that such calculations are expensive.

The row and column keys can be any immutable value (immutability is required as they will form keys in a dict at evaluation time). Keys which are tuples will be treated specially as renderers. For example, if each row key is a 2-tuple, then each row in the resulting table will have two row headers in two separate columns at the left of the table. This can aid in representing data with more than 2 dimensions in a table.

Consider a result set keyed by values A, B, and C. The table can be constructed with a series of 2-tuple row keys (A, B), while the column can be scalar C values. The resulting table will be rendered as follows:

		C1	C2	C3
A1	B1	data	data	data
	B2	data	data	data
A2	B1	data	data	data
	B2	data	data	data

Optional attributes also exist for *title*, *rows_title*, *cols_title*, *rows_unit*, and *cols_unit* (these all default to an empty string if omitted). In the case that tuples are used for row or column keys, the corresponding title and unit values must be tuples as well.

Note: Like *Result*, this class has an extended string representation intended for easy command line debugging. Printing an instance of this class will produce a human readable string representation of the table's row and column keys along with the calculated data.

as_ndarray

Returns the content of the table as a numpy `ndarray` with the shape `(rows, cols)`. Rows and columns will be in the order given by the *rows* and *cols* attributes. Please note that row and column keys are *not* included in the resulting array (as ndarrays purposely do not support heterogeneous data types).

Warning: Accessing this property will implicitly import the numpy module. This is not done during module import to avoid creating an explicit dependency on numpy.

as_dataframe

Returns the content of the table as a pandas `DataFrame`. The *rows* and *cols* attributes will be included as the index and columns of the resulting DataFrame.

> **Warning:** Accessing this property will implicitly import the pandas module. This is not done during module import to avoid creating an explicit dependency on pandas.

col_dims
The number of dimensions within the column keys. If this is greater than one, then *cols* is a sequence of tuples.

col_titles
Returns *cols_title* as a tuple, regardless. This property is intended to make renderers simpler.

cols
An ordered sequence of keys for the columns of the table. If *col_dims* is greater than one, this this is a sequence of tuples. These values, combined with *rows* can be used to index *data* in display order like so:

```
for row in table.rows:
    for col in table.cols:
        print(table.data[(row, col)])
```

cols_iter
Returns an iterator over *cols* where each key is returned as a tuple, regardless.

cols_title
A string or tuple of strings giving the title of each column dimension. Note that if *col_dims* is 1, this may be either a string or a 1-tuple containing a string. The associated *col_titles* attribute may be easier to work with.

cols_unit
A string or tuple of strings giving the units of each column dimension. Note that if *col_dims* is 1, this may be either a string or a 1-tuple containing a string. The associated `col_units` attribute may be easier to work with.

data
The data contained within the table. This is presented as a dict keyed by *(row_key, col_key)* tuples. To retrieve data in the same order as it should be presented, iterate over the *rows* and *cols* attributes.

data_iter
Returns an iterator over *data* where each key, value combination is returned as a tuple of (row_key, col_key, value), and each row and column key is returned as a tuple, regardless of the number of row and column dimensions. Furthermore, items are returned in declared row then column order. This property is intended to make renderers simpler.

name
The name of the table. This is intended for scripting usage and as such will only ever contain a string beginning with an alphabetic character followed by zero or more alphanumeric characters or underscores.

row_dims
The number of dimensions within the row keys. If this is greater than one, then *rows* is a sequence of tuples.

row_titles
Returns *rows_title* as a tuple, regardless. This property is intended to make renderers simpler.

rows
An ordered sequence of keys for the rows of the table. If *row_dims* is greater than one, then this is a sequence of tuples. These values, combined with *cols* can be used to index *data* in display order like so:

```
for row in table.rows:
    for col in table.cols:
        print(table.data[(row, col)])
```

rows_iter
Returns an iterator over *rows* where each key is returned as a tuple, regardless. This property is intended to make renderers simpler.

rows_title
A string or tuple of strings giving the title of each row dimension. Note that if *row_dims* is 1, this may be either a string or a 1-tuple containing a string. The associated *row_titles* attribute may be easier to work with.

rows_unit
A string or tuple of strings giving the units of each row dimension. Note that if *row_dims* is 1, this may be either a string or a 1-tuple containing a string. The associated `row_units` attribute may be easier to work with.

title
The human readable title of the table, typically rendered in the web interface as the table's caption.

1.5 Change log

1.5.1 Release 0.1 (2015-08-07)

Initial release

1.6 License

1.6.1 GNU General Public License

Version 2, June 1991 Copyright © 1989, 1991 Free Software Foundation, Inc. 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

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.

NO WARRANTY

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

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) <year> <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.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

u

Index