lyapunov-fm

Claude Heiland-Allen

2012–2019

Contents

1 COPYING

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. 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 them 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 prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same

freedoms that you received. 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.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

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

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without

permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey 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; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical

medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If 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 convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey

the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU 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 that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

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.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS 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.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

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 state 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 3 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, see <http://www.gnu.org/licenses/>.

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

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

> <program> Copyright (C) <year> <name of author>
> This program 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, your program's commands might be different; for a GUI interface, you would use an "about box".

```
  You should also get your employer (if you work as a programmer) or school,
665 if any, to sign a "copyright disclaimer" for the program, if necessary.
For more information on this, and how to apply and follow the GNU GPL, see
<http://www.gnu.org/licenses/>.

  The GNU General Public License does not permit incorporating your program
670 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. But first, please read
<http://www.gnu.org/philosophy/why-not-lgpl.html>.
```

2 doc/acl.bst

```
% BibTeX 'acl' style file for BibTeX version 0.99c, LaTeX version 2.09
% This version was made by modifying 'aaai-named' format based on the master
% file by Oren Patashnik (PATASHNIK@SCORE.STANFORD.EDU)

 5 % Copyright (C) 1985, all rights reserved.
% Modifications Copyright 1988, Peter F. Patel-Schneider
% Further modifictions by Stuart Shieber, 1991, and Fernando Pereira, 1992.
% Copying of this file is authorized only if either
% (1) you make absolutely no changes to your copy, including name, or
10 % (2) if you do make changes, you name it something other than
% btxbst.doc, plain.bst, unsrt.bst, alpha.bst, and abbrv.bst.
% This restriction helps ensure that all standard styles are identical.

% There are undoubtably bugs in this style.  If you make bug fixes,
15 % improvements, etc.  please let me know.  My e-mail address is:
%       pfps@spar.slb.com

%   Citation format: [author-last-name, year]
%                    [author-last-name and author-last-name, year]
20 %                    [author-last-name {\em et al.}, year]
%
%   Reference list ordering: alphabetical by author or whatever passes
%       for author in the absence of one.
%
25 % This BibTeX style has support for short (year only) citations.  This
% is done by having the citations actually look like
%         \citename{name-info, }year
% The LaTeX style has to have the following
%     \let\@internalcite\cite
30 %     \def\cite{\def\citename##1{##1}\@internalcite}
%     \def\shortcite{\def\citename##1{}\@internalcite}
%     \def\@biblabel#1{\def\citename##1{##1}[#1]\hfill}
% which makes \shortcite the macro for short citations.

35 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Changes made by SMS for thesis style
%   no emphasis on "et al."
%   "Ph.D." includes periods (not "PhD")
%   moved year to immediately after author's name
40 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ENTRY
  { address
```

```
      author
      booktitle
      chapter
      edition
      editor
      howpublished
      institution
      journal
      key
      month
      note
      number
      organization
      pages
      publisher
      school
      series
      title
      type
      volume
      year
    }
    {}
    { label extra.label sort.label }

INTEGERS { output.state before.all mid.sentence after.sentence after.block }

FUNCTION {init.state.consts}
{ #0 'before.all :=
  #1 'mid.sentence :=
  #2 'after.sentence :=
  #3 'after.block :=
}

STRINGS { s t }

FUNCTION {output.nonnull}
{ 's :=
  output.state mid.sentence =
    { ", " * write$ }
    { output.state after.block =
        { add.period$ write$
          newline$
          "\newblock " write$
        }
        { output.state before.all =
            'write$
            { add.period$ " " * write$ }
          if$
        }
      if$
      mid.sentence 'output.state :=
    }
  if$
  s
}
```

```
FUNCTION {output}
{ duplicate$ empty$
    'pop$
    'output.nonnull
  if$
}

FUNCTION {output.check}
{ 't :=
  duplicate$ empty$
    { pop$ "empty " t * " in " * cite$ * warning$ }
    'output.nonnull
  if$
}

FUNCTION {output.bibitem}
{ newline$

  "\bibitem[" write$
  label write$
  "]{" write$

  cite$ write$
  "}" write$
  newline$
  ""
  before.all 'output.state :=
}

FUNCTION {fin.entry}
{ add.period$
  write$
  newline$
}

FUNCTION {new.block}
{ output.state before.all =
    'skip$
    { after.block 'output.state := }
  if$
}

FUNCTION {new.sentence}
{ output.state after.block =
    'skip$
    { output.state before.all =
        'skip$
        { after.sentence 'output.state := }
      if$
    }
  if$
}

FUNCTION {not}
{   { #0 }
    { #1 }
  if$
```

```
}

FUNCTION {and}
{   'skip$
    { pop$ #0 }
  if$
}

FUNCTION {or}
{   { pop$ #1 }
    'skip$
  if$
}

FUNCTION {new.block.checka}
{ empty$
    'skip$
    'new.block
  if$
}

FUNCTION {new.block.checkb}
{ empty$
  swap$ empty$
  and
    'skip$
    'new.block
  if$
}

FUNCTION {new.sentence.checka}
{ empty$
    'skip$
    'new.sentence
  if$
}

FUNCTION {new.sentence.checkb}
{ empty$
  swap$ empty$
  and
    'skip$
    'new.sentence
  if$
}

FUNCTION {field.or.null}
{ duplicate$ empty$
    { pop$ "" }
    'skip$
  if$
}

FUNCTION {emphasize}
{ duplicate$ empty$
    { pop$ "" }
    { "{\em " swap$ * "}" * }
```

```
    if$
}

INTEGERS { nameptr namesleft numnames }

FUNCTION {format.names}
{ 's :=
  #1 'nameptr :=
  s num.names$ 'numnames :=
  numnames 'namesleft :=
    { namesleft #0 > }

    { s nameptr "{ff~}{vv~}{ll}{, jj}" format.name$ 't :=

      nameptr #1 >
        { namesleft #1 >
            { ", " * t * }
            { numnames #2 >
                { "," * }
                'skip$
              if$
              t "others" =
                { " et~al." * }
                { " and " * t * }
              if$
            }
          if$
        }
        't
      if$
      nameptr #1 + 'nameptr :=
      namesleft #1 - 'namesleft :=
    }
  while$
}

FUNCTION {format.authors}
{ author empty$
    { "" }
    { author format.names }
  if$
}

FUNCTION {format.editors}
{ editor empty$
    { "" }
    { editor format.names
      editor num.names$ #1 >
        { ", editors" * }
        { ", editor" * }
      if$
    }
  if$
}

FUNCTION {format.title}
{ title empty$
```

```
      { "" }

      { title "t" change.case$ }

275   if$
    }

    FUNCTION {n.dashify}
    { 't :=
280   ""
      { t empty$ not }
      { t #1 #1 substring$ "-" =
        { t #1 #2 substring$ "--" = not
          { "--" *
285         t #2 global.max$ substring$ 't :=
          }
          {   { t #1 #1 substring$ "-" = }
              { "-" *
                t #2 global.max$ substring$ 't :=
290           }
            while$
          }
        if$
        }
295     { t #1 #1 substring$ *
        t #2 global.max$ substring$ 't :=
        }
      if$
      }
300   while$
    }

    FUNCTION {format.date}
    { year empty$
305   { month empty$
        { "" }
        { "there's a month but no year in " cite$ * warning$
          month
        }
310     if$
      }
      { month empty$
        { "" }
        { month }
315     if$
      }
    if$
    }

320 FUNCTION {format.btitle}
    { title emphasize
    }

    FUNCTION {tie.or.space.connect}
325 { duplicate$ text.length$ #3 <
      { "~" }
      { " " }
```

```
    if$
    swap$ * *
}

FUNCTION {either.or.check}
{ empty$
    'pop$
    { "can't use both " swap$ * " fields in " * cite$ * warning$ }
  if$
}

FUNCTION {format.bvolume}
{ volume empty$
    { "" }
    { "volume" volume tie.or.space.connect
      series empty$
        'skip$
        { " of " * series emphasize * }
      if$
      "volume and number" number either.or.check
    }
  if$
}

FUNCTION {format.number.series}
{ volume empty$
    { number empty$
        { series field.or.null }
        { output.state mid.sentence =
            { "number" }
            { "Number" }
          if$
          number tie.or.space.connect
          series empty$
            { "there's a number but no series in " cite$ * warning$ }
            { " in " * series * }
          if$
        }
      if$
    }
    { "" }
  if$
}

FUNCTION {format.edition}
{ edition empty$
    { "" }
    { output.state mid.sentence =
        { edition "l" change.case$ " edition" * }
        { edition "t" change.case$ " edition" * }
      if$
    }
  if$
}

INTEGERS { multiresult }
```

```
FUNCTION {multi.page.check}
{ 't :=
  #0 'multiresult :=
    { multiresult not
      t empty$ not
      and
    }
    { t #1 #1 substring$
      duplicate$ "-" =
      swap$ duplicate$ "," =
      swap$ "+" =
      or or
        { #1 'multiresult := }
        { t #2 global.max$ substring$ 't := }
      if$
    }
  while$
  multiresult
}

FUNCTION {format.pages}
{ pages empty$
    { "" }
    { pages multi.page.check
        { "pages" pages n.dashify tie.or.space.connect }
        { "page" pages tie.or.space.connect }
      if$
    }
  if$
}

FUNCTION {format.year.label}
{ year extra.label *
}

FUNCTION {format.vol.num.pages}
{ volume field.or.null
  number empty$
    'skip$
    { "(" number * ")" * *
      volume empty$
        { "there's a number but no volume in " cite$ * warning$ }
        'skip$
      if$
    }
  if$
  pages empty$
    'skip$
    { duplicate$ empty$
        { pop$ format.pages }
        { ":" * pages n.dashify * }
      if$
    }
  if$
}

FUNCTION {format.chapter.pages}
```

```
{ chapter empty$
    'format.pages
    { type empty$
        { "chapter" }
        { type "l" change.case$ }
      if$
      chapter tie.or.space.connect
      pages empty$
        'skip$
        { ", " * format.pages * }
      if$
    }
  if$
}

FUNCTION {format.in.ed.booktitle}
{ booktitle empty$
    { "" }
    { editor empty$
        { "In " booktitle emphasize * }
        { "In " format.editors * ", " * booktitle emphasize * }
      if$
    }
  if$
}

FUNCTION {empty.misc.check}
{ author empty$ title empty$ howpublished empty$
  month empty$ year empty$ note empty$
  and and and and and

  key empty$ not and

    { "all relevant fields are empty in " cite$ * warning$ }
    'skip$
  if$
}

FUNCTION {format.thesis.type}
{ type empty$
    'skip$
    { pop$
      type "t" change.case$
    }
  if$
}

FUNCTION {format.tr.number}
{ type empty$
    { "Technical Report" }
    'type
  if$
  number empty$
    { "t" change.case$ }
    { number tie.or.space.connect }
  if$
}
```

```
FUNCTION {format.article.crossref}
{ key empty$
    { journal empty$
        { "need key or journal for " cite$ * " to crossref " * crossref *
          warning$
          ""
        }
        { "In {\em " journal * "\/}" * }
      if$
    }
    { "In " key * }
  if$
  " \cite{" * crossref * "}" *
}

FUNCTION {format.crossref.editor}
{ editor #1 "{vv~}{ll}" format.name$
  editor num.names$ duplicate$
  #2 >
    { pop$ " et~al." * }
    { #2 <
        'skip$
        { editor #2 "{ff }{vv }{ll}{ jj}" format.name$ "others" =
            { " et~al." * }
            { " and " * editor #2 "{vv~}{ll}" format.name$ * }
          if$
        }
      if$
    }
  if$
}

FUNCTION {format.book.crossref}
{ volume empty$
    { "empty volume in " cite$ * "'s crossref of " * crossref * warning$
      "In "
    }
    { "Volume" volume tie.or.space.connect
      " of " *
    }
  if$
  editor empty$
  editor field.or.null author field.or.null =
  or
    { key empty$
        { series empty$
            { "need editor, key, or series for " cite$ * " to crossref " *
              crossref * warning$
              "" *
            }
            { "{\em " * series * "\/}" * }
          if$
        }
        { key * }
      if$
    }
```

```
      { format.crossref.editor * }
    if$
    " \cite{" * crossref * "}" *
  }

  FUNCTION {format.incoll.inproc.crossref}
  { editor empty$
    editor field.or.null author field.or.null =
    or
      { key empty$
          { booktitle empty$
              { "need editor, key, or booktitle for " cite$ * " to crossref " *
                crossref * warning$
                ""
              }
              { "In {\em " booktitle * "\/}" * }
            if$
          }
          { "In " key * }
        if$
      }
      { "In " format.crossref.editor * }
    if$
    " \cite{" * crossref * "}" *
  }

  FUNCTION {article}
  { output.bibitem
    format.authors "author" output.check
    new.block
    format.year.label "year" output.check
    new.block
    format.title "title" output.check
    new.block
    crossref missing$
      { journal emphasize "journal" output.check
        format.vol.num.pages output
        format.date output
      }
      { format.article.crossref output.nonnull
        format.pages output
      }
    if$
    new.block
    note output
    fin.entry
  }

  FUNCTION {book}
  { output.bibitem
    author empty$
      { format.editors "author and editor" output.check }
      { format.authors output.nonnull
        crossref missing$
          { "author and editor" editor either.or.check }
          'skip$
        if$
```

```
      }
    if$
    new.block
    format.year.label "year" output.check
    new.block
    format.btitle "title" output.check
    crossref missing$
      { format.bvolume output
        new.block
        format.number.series output
        new.sentence
        publisher "publisher" output.check
        address output
      }
      { new.block
        format.book.crossref output.nonnull
      }
    if$
    format.edition output
    format.date output
    new.block
    note output
    fin.entry
}

FUNCTION {booklet}
{ output.bibitem
    format.authors output
    new.block
    format.year.label "year" output.check
    new.block
    format.title "title" output.check
    howpublished address new.block.checkb
    howpublished output
    address output
    format.date output
    new.block
    note output
    fin.entry
}

FUNCTION {inbook}
{ output.bibitem
    author empty$
      { format.editors "author and editor" output.check }
      { format.authors output.nonnull
        crossref missing$
          { "author and editor" editor either.or.check }
          'skip$
        if$
      }
    if$
    format.year.label "year" output.check
    new.block
    new.block
    format.btitle "title" output.check
    crossref missing$
```

```
      { format.bvolume output
        format.chapter.pages "chapter and pages" output.check
        new.block
        format.number.series output
        new.sentence
        publisher "publisher" output.check
        address output
      }
      { format.chapter.pages "chapter and pages" output.check
        new.block
        format.book.crossref output.nonnull
      }
    if$
    format.edition output
    format.date output
    new.block
    note output
    fin.entry
  }

  FUNCTION {incollection}
  { output.bibitem
    format.authors "author" output.check
    new.block
    format.year.label "year" output.check
    new.block
    format.title "title" output.check
    new.block
    crossref missing$
      { format.in.ed.booktitle "booktitle" output.check
        format.bvolume output
        format.number.series output
        format.chapter.pages output
        new.sentence
        publisher "publisher" output.check
        address output
        format.edition output
        format.date output
      }
      { format.incoll.inproc.crossref output.nonnull
        format.chapter.pages output
      }
    if$
    new.block
    note output
    fin.entry
  }

  FUNCTION {inproceedings}
  { output.bibitem
    format.authors "author" output.check
    new.block
    format.year.label "year" output.check
    new.block
    format.title "title" output.check
    new.block
    crossref missing$
```

```
      { format.in.ed.booktitle "booktitle" output.check
        format.bvolume output
        format.number.series output
        format.pages output
        address empty$
          { organization publisher new.sentence.checkb
            organization output
            publisher output
            format.date output
          }
          { address output.nonnull
            format.date output
            new.sentence
            organization output
            publisher output
          }
        if$
      }
      { format.incoll.inproc.crossref output.nonnull
        format.pages output
      }
    if$
    new.block
    note output
    fin.entry
}

FUNCTION {conference} { inproceedings }

FUNCTION {manual}
{ output.bibitem
  author empty$
    { organization empty$
        'skip$
        { organization output.nonnull
          address output
        }
      if$
    }
    { format.authors output.nonnull }
  if$
  format.year.label "year" output.check
  new.block
  new.block
  format.btitle "title" output.check
  author empty$
    { organization empty$
        { address new.block.checka
          address output
        }
        'skip$
      if$
    }
    { organization address new.block.checkb
      organization output
      address output
    }
```

```
    if$
    format.edition output
    format.date output
    new.block
    note output
    fin.entry
}

FUNCTION {mastersthesis}
{ output.bibitem
    format.authors "author" output.check
    new.block
    format.year.label "year" output.check
    new.block
    format.title "title" output.check
    new.block
    "Master's thesis" format.thesis.type output.nonnull
    school "school" output.check
    address output
    format.date output
    new.block
    note output
    fin.entry
}

FUNCTION {misc}
{ output.bibitem
    format.authors output
    new.block
    format.year.label output
    new.block
    title howpublished new.block.checkb
    format.title output
    howpublished new.block.checka
    howpublished output
    format.date output
    new.block
    note output
    fin.entry
    empty.misc.check
}

FUNCTION {phdthesis}
{ output.bibitem
    format.authors "author" output.check
    new.block
    format.year.label "year" output.check
    new.block
    format.btitle "title" output.check
    new.block
    "{Ph.D.} thesis" format.thesis.type output.nonnull
    school "school" output.check
    address output
    format.date output
    new.block
    note output
    fin.entry
```

```
}

FUNCTION {proceedings}
{ output.bibitem
  editor empty$
    { organization output }
    { format.editors output.nonnull }
  if$
  new.block
  format.year.label "year" output.check
  new.block
  format.btitle "title" output.check
  format.bvolume output
  format.number.series output
  address empty$
    { editor empty$
        { publisher new.sentence.checka }
        { organization publisher new.sentence.checkb
          organization output
        }
      if$
      publisher output
      format.date output
    }
    { address output.nonnull
      format.date output
      new.sentence
      editor empty$
        'skip$
        { organization output }
      if$
      publisher output
    }
  if$
  new.block
  note output
  fin.entry
}

FUNCTION {techreport}
{ output.bibitem
  format.authors "author" output.check
  new.block
  format.year.label "year" output.check
  new.block
  format.title "title" output.check
  new.block
  format.tr.number output.nonnull
  institution "institution" output.check
  address output
  format.date output
  new.block
  note output
  fin.entry
}

FUNCTION {unpublished}
```

```
{ output.bibitem
  format.authors "author" output.check
  new.block
  format.year.label "year" output.check
  new.block
  format.title "title" output.check
  new.block
  note "note" output.check
  format.date output
  fin.entry
}

FUNCTION {default.type} { misc }

MACRO {jan} {"January"}

MACRO {feb} {"February"}

MACRO {mar} {"March"}

MACRO {apr} {"April"}

MACRO {may} {"May"}

MACRO {jun} {"June"}

MACRO {jul} {"July"}

MACRO {aug} {"August"}

MACRO {sep} {"September"}

MACRO {oct} {"October"}

MACRO {nov} {"November"}

MACRO {dec} {"December"}

MACRO {acmcs} {"ACM Computing Surveys"}

MACRO {acta} {"Acta Informatica"}

MACRO {cacm} {"Communications of the ACM"}

MACRO {ibmjrd} {"IBM Journal of Research and Development"}

MACRO {ibmsj} {"IBM Systems Journal"}

MACRO {ieeese} {"IEEE Transactions on Software Engineering"}

MACRO {ieeetc} {"IEEE Transactions on Computers"}

MACRO {ieeetcad}
 {"IEEE Transactions on Computer-Aided Design of Integrated Circuits"}

MACRO {ipl} {"Information Processing Letters"}
```

```
MACRO {jacm} {"Journal of the ACM"}

MACRO {jcss} {"Journal of Computer and System Sciences"}

MACRO {scp} {"Science of Computer Programming"}

MACRO {sicomp} {"SIAM Journal on Computing"}

MACRO {tocs} {"ACM Transactions on Computer Systems"}

MACRO {tods} {"ACM Transactions on Database Systems"}

MACRO {tog} {"ACM Transactions on Graphics"}

MACRO {toms} {"ACM Transactions on Mathematical Software"}

MACRO {toois} {"ACM Transactions on Office Information Systems"}

MACRO {toplas} {"ACM Transactions on Programming Languages and Systems"}

MACRO {tcs} {"Theoretical Computer Science"}

READ

FUNCTION {sortify}
{ purify$
  "l" change.case$
}

INTEGERS { len }

FUNCTION {chop.word}
{ 's :=
  'len :=
  s #1 len substring$ =
    { s len #1 + global.max$ substring$ }
    's
  if$
}

INTEGERS { et.al.char.used }

FUNCTION {initialize.et.al.char.used}
{ #0 'et.al.char.used :=
}

EXECUTE {initialize.et.al.char.used}

FUNCTION {format.lab.names}
{ 's :=
  s num.names$ 'numnames :=

  numnames #1 =
    { s #1 "{vv }{ll}" format.name$ }
    { numnames #2 =
        { s #1 "{vv }{ll }and " format.name$ s #2 "{vv }{ll}" format.name$ *
        }
```

```
          { s #1 "{vv }{ll }\bgroup et al.\egroup " format.name$ }
        if$
      }
    if$

}

FUNCTION {author.key.label}
{ author empty$
    { key empty$

        { cite$ #1 #3 substring$ }

        { key #3 text.prefix$ }
      if$
    }
    { author format.lab.names }
  if$
}

FUNCTION {author.editor.key.label}
{ author empty$
    { editor empty$
        { key empty$

            { cite$ #1 #3 substring$ }

            { key #3 text.prefix$ }
          if$
        }
        { editor format.lab.names }
      if$
    }
    { author format.lab.names }
  if$
}

FUNCTION {author.key.organization.label}
{ author empty$
    { key empty$
        { organization empty$

            { cite$ #1 #3 substring$ }

            { "The " #4 organization chop.word #3 text.prefix$ }
          if$
        }
        { key #3 text.prefix$ }
      if$
    }
    { author format.lab.names }
  if$
}

FUNCTION {editor.key.organization.label}
{ editor empty$
    { key empty$
```

```
        { organization empty$
1070
            { cite$ #1 #3 substring$ }

            { "The " #4 organization chop.word #3 text.prefix$ }
          if$
1075      }
          { key #3 text.prefix$ }
        if$
      }
      { editor format.lab.names }
1080  if$
}

FUNCTION {calc.label}
{ type$ "book" =
1085  type$ "inbook" =
  or
    'author.editor.key.label
    { type$ "proceedings" =
        'editor.key.organization.label
1090    { type$ "manual" =
            'author.key.organization.label
            'author.key.label
          if$
        }
1095    if$
    }
  if$
  duplicate$

1100  "\protect\citename{" swap$ * "}" *
  year field.or.null purify$ *
  'label :=
  year field.or.null purify$ *

1105  sortify 'sort.label :=
}

FUNCTION {sort.format.names}
{ 's :=
1110  #1 'nameptr :=
  ""
  s num.names$ 'numnames :=
  numnames 'namesleft :=
    { namesleft #0 > }
1115    { nameptr #1 >
        { "   " * }
        'skip$
      if$

1120    s nameptr "{vv{ } }{ll{ }}{  ff{ }}{  jj{ }}" format.name$ 't :=

      nameptr numnames = t "others" = and
        { "et al" * }
        { t sortify * }
1125    if$
```

```
      nameptr #1 + 'nameptr :=
      namesleft #1 - 'namesleft :=
    }
  while$
}

FUNCTION {sort.format.title}
{ 't :=
  "A " #2
    "An " #3
      "The " #4 t chop.word
    chop.word
  chop.word
  sortify
  #1 global.max$ substring$
}

FUNCTION {author.sort}
{ author empty$
    { key empty$
        { "to sort, need author or key in " cite$ * warning$
          ""
        }
        { key sortify }
      if$
    }
    { author sort.format.names }
  if$
}

FUNCTION {author.editor.sort}
{ author empty$
    { editor empty$
        { key empty$
            { "to sort, need author, editor, or key in " cite$ * warning$
              ""
            }
            { key sortify }
          if$
        }
        { editor sort.format.names }
      if$
    }
    { author sort.format.names }
  if$
}

FUNCTION {author.organization.sort}
{ author empty$
    { organization empty$
        { key empty$
            { "to sort, need author, organization, or key in " cite$ * warning$
              ""
            }
            { key sortify }
          if$
        }
```

```
          { "The " #4 organization chop.word sortify }
        if$
      }
      { author sort.format.names }
    if$
}

FUNCTION {editor.organization.sort}
{ editor empty$
    { organization empty$
        { key empty$
            { "to sort, need editor, organization, or key in " cite$ * warning$
              ""
            }
            { key sortify }
          if$
        }
        { "The " #4 organization chop.word sortify }
      if$
    }
    { editor sort.format.names }
  if$
}

FUNCTION {presort}

{ calc.label
  sort.label
  "    "
  *
  type$ "book" =

  type$ "inbook" =
  or
    'author.editor.sort
    { type$ "proceedings" =
        'editor.organization.sort
        { type$ "manual" =
            'author.organization.sort
            'author.sort
          if$
        }
      if$
    }
  if$

  *

  "    "
  *
  year field.or.null sortify
  *
  "    "
  *
  title field.or.null
  sort.format.title
  *
```

```
  #1 entry.max$ substring$
  'sort.key$ :=
}

ITERATE {presort}

SORT

STRINGS { longest.label last.sort.label next.extra }

INTEGERS { longest.label.width last.extra.num }

FUNCTION {initialize.longest.label}
{ "" 'longest.label :=
  #0 int.to.chr$ 'last.sort.label :=
  "" 'next.extra :=
  #0 'longest.label.width :=
  #0 'last.extra.num :=
}

FUNCTION {forward.pass}
{ last.sort.label sort.label =
    { last.extra.num #1 + 'last.extra.num :=
      last.extra.num int.to.chr$ 'extra.label :=
    }
    { "a" chr.to.int$ 'last.extra.num :=
      "" 'extra.label :=
      sort.label 'last.sort.label :=
    }
  if$
}

FUNCTION {reverse.pass}
{ next.extra "b" =
    { "a" 'extra.label := }
    'skip$
  if$
  label extra.label * 'label :=
  label width$ longest.label.width >
    { label 'longest.label :=
      label width$ 'longest.label.width :=
    }
    'skip$
  if$
  extra.label 'next.extra :=
}

EXECUTE {initialize.longest.label}

ITERATE {forward.pass}

REVERSE {reverse.pass}

FUNCTION {begin.bib}

{ et.al.char.used
    { "\newcommand{\etalchar}[1]{$^{#1}$}" write$ newline$ }
```

```
      'skip$
    if$
    preamble$ empty$
1300
      'skip$
      { preamble$ write$ newline$ }
    if$

1305  "\begin{thebibliography}{" "}" * write$ newline$

}

EXECUTE {begin.bib}
1310
EXECUTE {init.state.consts}

ITERATE {call.type$}

1315 FUNCTION {end.bib}
{ newline$
  "\end{thebibliography}" write$ newline$
}

1320 EXECUTE {end.bib}
```

3 doc/figure1.pd

```
#N canvas 0 0 169 102 36;
#N canvas 0 0 1905 1014 chaos 1;
#X obj 164 356 *~;
#X obj 164 410 +~;
 5 #X obj 164 464 mtof~;
#X obj 164 518 osc~;
#X obj 381 356 *~;
#X obj 381 410 +~;
#X obj 381 464 mtof~;
10 #X obj 381 518 osc~;
#X obj 616 100 block~ 2;
#X obj 164 100 table x 2;
#X obj 381 100 table y 2;
#N canvas 0 576 212 78 receive 0;
15 #X obj 72 184 tabreceive~ y;
#X obj 444 184 tabreceive~ x;
#X obj 72 268 outlet~;
#X obj 444 268 outlet~;
#X connect 0 0 2 0;
20 #X connect 1 0 3 0;
#X restore 164 282 pd receive;
#N canvas 0 754 212 76 send 0;
#X obj 80 108 inlet~;
#X obj 80 192 tabsend~ x;
25 #X obj 456 108 inlet~;
#X obj 456 192 tabsend~ y;
#X connect 0 0 1 0;
#X connect 2 0 3 0;
#X restore 164 592 pd send;
30 #X obj 244 410 r f_x;
```

```
#X obj 244 356 r m_x;
#X obj 461 356 r m_y;
#X obj 461 410 r f_y;
#X obj 204 174 tabreceive~ y;
#X obj 506 174 tabreceive~ x;
#X obj 204 228 outlet~;
#X obj 506 228 outlet~;
#X obj 204 646 inlet~;
#X obj 204 700 tabsend~ x;
#X obj 506 646 inlet~;
#X obj 506 700 tabsend~ y;
#X connect 0 0 1 0;
#X connect 1 0 2 0;
#X connect 2 0 3 0;
#X connect 3 0 12 0;
#X connect 4 0 5 0;
#X connect 5 0 6 0;
#X connect 6 0 7 0;
#X connect 7 0 12 1;
#X connect 11 0 0 0;
#X connect 11 1 4 0;
#X connect 13 0 1 1;
#X connect 14 0 0 1;
#X connect 15 0 4 1;
#X connect 16 0 5 1;
#X connect 17 0 19 0;
#X connect 18 0 20 0;
#X connect 21 0 22 0;
#X connect 23 0 24 0;
#X restore 172 168 pd chaos;
#X floatatom 80 92 4 0 0 2 f_x - -;
#X floatatom 200 92 4 0 0 2 f_y - -;
#X floatatom 320 92 4 0 0 2 m_x - -;
#X floatatom 440 92 4 0 0 2 m_y - -;
#X obj 212 248 dac~;
```

4 doc/lac2013.sty

```
%
% lac2013.sty
%
\typeout{}
\typeout{LAC2013 Proceedings style -- Sep 2012}
%\typeout{switch back to A4-Letter format}
%\typeout{LAC2012 Proceedings style -- Jan 2012}
%\typeout{updated for US-Letter format}
%\typeout{LAC2012 Proceedings style -- October 2011}
%\typeout{only slightly different from}
%\typeout{LAC2011 Proceedings style -- October 2010}
%\typeout{only slightly different from}
%\typeout{LAC2010 Proceedings style -- November 2009}
%\typeout{only slightly different from}
%\typeout{LAC2009 Proceedings style -- October 2008}
%\typeout{only slightly different from}
%\typeout{LAC2008 Proceedings style -- September 2007}
%\typeout{only slightly different from}
%\typeout{LAC2007 Proceedings style -- September 2006}
```

```
%\typeout{only slightly different from}
%\typeout{LAC2005 Proceedings style -- November 2005}
%\typeout{only slightly different from}
%\typeout{Coling 2004 Proceedings style -- February 2004}
%\typeout{only slightly different from}
%\typeout{ACL-COLING 1998 Proceedings style -- March 31st 1998}
\typeout{}
%
% ---------------------------------------------------------------------
%
% This is the LaTeX style file for LAC2012 (lac2012.sty).
% It is an almost unmodified copy of colacl.sty, the style file
% of Coling (International Conference on Computational Linguistics).
% Many thanks to Coling for their permission to use it!
% The original may be found here:
% http://www.issco.unige.ch/coling2004/coling2004downloads.html
%
% ---------------------------------------------------------------------
%
% This is the LaTeX style file for ACL-COLING 1998.  It is based on
% a series of similar files prepared for previous conferences by
% Fernando Pereira, Paul Jacobs, Stuart Shieber, Peter F.
% Patel-Schneider and others.  Various changes have been made, chiefly
% to save space in the final output or remove redundant definitions.
%
% colacl.sty is designed for use as a package or option with the
% standard LaTeX article class, and the BibTeX style acl.bst.
%
% Author/title and citation formatting differs slightly from standard
% LaTeX; see AUTHOR FORMATS and CITATION FORMATS below for more
% information.
%
% This file is supplied as a hopefully convenient implementation of
% some of the "instructions for authors" repeated below.  It is not
% guaranteed to work in any given LaTeX installation or in conjunction
% with any given class, package or style, and it is not intended as
% a LaTeX tutorial.
%
% ---------------------------------------------------------------------
% Instructions for authors
%
%   (i) Maximum length of full papers: 8 pages.
%
%  (ii) Paper size: A4.
%
% (iii) Margins: set so that text lies within a rectangle 9in (23cm)
%       high and 6.5in (16.5cm) wide.
%
%  (iv) Body of text to be set in two columns.  Full-width figures
%       (i.e. using \begin{figure*}) and tables may be used if
%       necessary.
%
%   (v) Use standard fonts, e.g. Computer Modern Roman, Times Roman, no
%       smaller than 10pt.
%
%  (vi) No page numbering.
%
```

```
% Items (iii), (iv) and (vi) are handled by this file, and should
% therefore not be overridden by resetting \textwidth, \textheight,
% \pagestyle etc. in your document, or by using styles or packages
% which have the same effect.
%
% ---------------------------------------------------------------------
% To convert papers prepared with colaclsub.sty to the final format
% for use with colacl.sty:
%
% (1) Remove commands specific to the original submission format
%     (\type, \subject, \contact, \conference, \makeidpage).
%
% (2) Replace \summary{...} with an abstract, using the normal
%     abstract environment, placed after \maketitle.
%
% A simple document template:
%
%   \documentclass[11pt,letterpaper]{article}
%   \usepackage{lac2012}
%   \title{...}
%   \author{...}              % see below for possible formats
%   \begin{document}
%   \maketitle
%   \begin{abstract}
%   ...                       % contents of abstract
%   \end{abstract}
%   ...                       % contents of article
%   \bibliographystyle{acl}  % use acl.bst
%   \bibliography{...}
%   \end{document}
%
% Users of obsolete LaTeX versions can try:
%
%   \documentstyle[lac2012]{article}    % or [11pt,lac2012]
%   \title{...}
%   ...
%
%
% ---------------------------------------------------------------------
% AUTHOR FORMATS
%
% Author information can be set in various styles.
%
% For several authors from the same institution:
% \author{Author 1 \and ... \and Author n \\
%         Address line \\ ... \\ Address line}
% if the names do not fit well on one line use
%         Author 1 \\ {\bf Author 2} \\ ... \\ {\bf Author n} \\
%
% For authors from different institutions:
% \author{Author 1 \\ Address line \\  ... \\ Address line
%         \And  ... \And
%         Author n \\ Address line \\ ... \\ Address line}
%
% To start a separate "row" of authors use \AND, as in
% \author{Author 1 \\ Address line \\  ... \\ Address line
%         \AND
```

```
%          Author 2 \\ Address line \\ ... \\ Address line \And
%          Author 3 \\ Address line \\ ... \\ Address line}
%
% If the title and author information does not fit in the area allocated,
% place \setlength\titlebox{<new height>} after \usepackage{colacl},
% where <new height> can be something larger than 2.0in
%
% ---------------------------------------------------------------------
% CITATION FORMATS
%
% Three possible citation formats:
% "\cite{...}"      produces a citation like "(Author, 1999)"
% "\shortcite{...}" produces a citation like "(1999)"
% "\newcite{...}"   produces a citation like "Author (1999)"
%
% All three take an optional argument which can be used to add page
% references, etc.:
% "\newcite[1--6]{...}" produces a citation like "Author (1999, 1--6)"
%
% ---------------------------------------------------------------------
% IF IT DOESN'T WORK
%
% The error message "File `colacl.sty' not found." indicates that this
% file has not been installed in a location which is visible to your
% LaTeX.  Try putting it in the same directory as your paper, and
% running LaTeX there.  Consult your `Local Guide' documentation or
% your system administrator to find out how LaTeX searches for input
% files.
%
% "\documentclass..." is a LaTeX2e declaration.  An error message
% "Undefined control sequence." followed by a line ending in
% "\documentclass" indicates that you have used this with an obsolete
% LaTeX installation.  Use the "\documentstyle" variant shown above.
%
% As a last resort, forget about colacl.sty and simply copy the
% following lines (uncommented, obviously) into your document before
% the "\begin{document}":
%
% \setlength\topmargin{0.0in}
% \setlength\oddsidemargin{-0.0in}
% \setlength\textheight{9.0in}
% \setlength\textwidth{6.5in}
% \setlength\columnsep{0.25in}
% \setlength\headheight{0pt}
% \setlength\headsep{0pt}
% \thispagestyle{empty}
% \pagestyle{empty}
% \flushbottom
% \twocolumn
% \sloppy
%
% Some interactions with other packages may still occur.  In order to
% remove the page number from the first page, you may have to place the
% "\thispagestyle{empty}" command immediately after "\maketitle".
%
% ---------------------------------------------------------------------
% NOTE:  Some laser printers have a serious problem printing TeX output.
```

```
% These printing devices, commonly known as "write-white" laser
% printers, tend to make characters too light.  To get around this
% problem, a darker set of fonts must be created for these devices.
%
% ---------------------------------------------------------------------
% Physical page layout - slightly modified from IJCAI by pj
% Physical page layout - slightly modified by rg for LAC
\newlength\myavierh
\setlength\myavierh{297mm}
%\message{The page height is \the\paperheight}
\ifdim \paperheight = \myavierh
\message{ !! Using ISO-A4 paper}
%% A4paper
\setlength\topmargin{0.0in}
\setlength\oddsidemargin{-0.0in}
\setlength\textheight{9.5in}   %%sm was 9.0in
\setlength\textwidth{6.5in}
\setlength\columnsep{0.25in}
\newlength\titlebox
\setlength\titlebox{2.0in}              % was 2.25in
\setlength\headheight{0pt}
\setlength\headsep{0pt}
\setlength\footskip{0pt}                % irrelevant when no footers.

\else
%% US-Letter
\message{ !! Using US-Letter paper}
\setlength{\pdfpagewidth}{8.5in}
\setlength{\pdfpageheight}{11in}
\setlength\topmargin{0.0in}
\setlength\oddsidemargin{-0.2in}
\setlength\textheight{8.8in}
\setlength\textwidth{6.9in}
\setlength\columnsep{0.4in}
\newlength\titlebox
\setlength\titlebox{2.1in}
\setlength\headheight{0pt}
\setlength\headsep{0pt}
\setlength\footskip{0pt}      % irrelevant when no footers.
\fi

\pagestyle{empty}                       % no page numbers
\thispagestyle{empty}                   % no page numbers
\flushbottom
\twocolumn
\sloppy

% We're never going to need a table of contents, so just flush it to
% save space --- suggested by drstrip@sandia-2
\def\addcontentsline#1#2#3{}

% ---------------------------------------------------------------------
% Title stuff, taken from deproc.

\def\maketitle{%
  \par%
  \begingroup%
```

```
      \def\thefootnote{\fnsymbol{footnote}}%
      \def\@makefnmark{\rlap{$^{\@thefnmark}$\hss}}%
      % no paragraph breaks in \thanks
      \long\def\@makefntext##1{%
                  \parindent 1em\noindent%
                  \hbox to 1em{$^{\@thefnmark}$}##1}
      \twocolumn[\@maketitle] \@thanks%
  \endgroup%
  \setcounter{footnote}{0}%
  \let\maketitle\relax\let\@maketitle\relax%
  \gdef\@thanks{}\gdef\@author{}\gdef\@title{}%
  \let\thanks\relax}

% some vertical space removed here: skip above and below title
%
\def\@maketitle{%
  \vbox to \titlebox{%
    \hsize\textwidth\linewidth\hsize%
    \vskip 0.125in minus 0.05in%
    \centering{\Large\bf \@title \par}%
    \vskip 0.2in plus 0.1fil minus 0.1in
    {\def\and{\unskip\enspace{\rm and}\enspace}%
     \def\And{\end{tabular}\hss \egroup \hskip 1in plus 2fil
              \hbox to 0pt\bgroup\hss \begin{tabular}[t]{c}\bf}%
     \def\AND{\end{tabular}\hss\egroup \hfil\hfil\egroup
              \vskip 0.25in plus 1fil minus 0.125in
              \hbox to \linewidth\bgroup\large \hfil\hfil
              \hbox to 0pt\bgroup\hss \begin{tabular}[t]{c}\bf}
    \hbox to \linewidth \bgroup\large \hfil\hfil
    \hbox to 0pt\bgroup\hss \begin{tabular}[t]{c}\bf\@author
                            \end{tabular}\hss\egroup
    \hfil\hfil\egroup}
  \vskip 0.3in plus 2fil minus 0.1in
}}

% ------------------------------------------------------------------------
% abstract, changed for LAC2005

% \renewenvironment{abstract}{\section*{\centerline{Abstract}}}{}
\renewenvironment{abstract}{\section*{Abstract}}{}

% ------------------------------------------------------------------------
% keywords, added for LAC2005

\def\keywords{\section*{Keywords}}

% ------------------------------------------------------------------------
% bibliography and citations

% most of cite format is from aclsub.sty by SMS

% don't box citations, separate with ; and a space
% Replaced for multiple citations (pj)
% don't box citations and also add space, semicolon between multiple
% citations
%
\def\@citex[#1]#2{\if@filesw\immediate\write\@auxout{\string\citation{#2}}\fi
```

```
305     \def\@citea{}\@cite{\@for\@citeb:=#2\do
          {\@citea\def\@citea{; }\@ifundefined
            {b@\@citeb}{{\bf ?}\@warning
             {Citation `\@citeb' on page \thepage \space undefined}}%
       {\csname b@\@citeb\endcsname}}}{#1}}
310
    % Allow short (name-less) citations, when used in
    % conjunction with a bibliography style that creates labels like
    %       \citename{<names>, }<year>
    %
315 \let\@internalcite\cite
    \def\cite{\def\citename##1{##1, }\@internalcite}
    \def\longcite{\def\citename##1{##1, }\@internalcite}

    \def\shortcite{\def\citename##1{}\@internalcite}
320 \def\newcite{\def\citename##1{{##1}\ (}\@internalciteb}

    % Macros for \newcite, which leaves name in running text, and is
    % otherwise like \shortcite.
    %
325 \def\@citexb[#1]#2{\if@filesw\immediate\write\@auxout{\string\citation{#2}}\fi
      \def\@citea{}\@newcite{\@for\@citeb:=#2\do
        {\@citea\def\@citea{;\penalty\@m\ }\@ifundefined
           {b@\@citeb}{{\bf ?}\@warning
           {Citation `\@citeb' on page \thepage \space undefined}}%
330 % gjr: hbox causes too many bad linebreaks
    %\hbox{\csname b@\@citeb\endcsname}}}{#1}}
         {\csname b@\@citeb\endcsname}}}{#1}}

    \def\@internalciteb{%
335   \@ifnextchar [{\@tempswatrue\@citexb}{\@tempswafalse\@citexb[]}}

    \def\@newcite#1#2{{#1\if@tempswa, #2\fi)}}

    % gjr: no labels in this bibliography style
340 % \def\@biblabel#1{\def\citename##1{##1}[#1]\hfill}
    \def\@biblabel#1{}

    % use square brackets after all
    %%% More changes made by SMS (originals in latex.tex)
345 % Use parentheses instead of square brackets in the text.
    % \def\@cite#1#2{({#1\if@tempswa , #2\fi})}

    % Don't put a label in the bibliography at all.  Just use the unlabeled format
    % instead.
350 % gjr: removed \@mkboth -- no headers here.
    % gjr: reduced vertical space between entries (plus was .33em)
    %

    \def\thebibliography#1{\section*{References}\list
355  {[\arabic{enumi}]}{\settowidth\labelwidth{[#1]}\leftmargin\labelwidth
     \advance\leftmargin\labelsep
     \usecounter{enumi}}
     \def\newblock{\hskip .11em plus .11em minus .07em}
     \sloppy\clubpenalty4000\widowpenalty4000
360  \sfcode`\.=1000\relax}
    \let\endthebibliography=\endlist
```

```
% \def\thebibliography#1{%
%   \section*{References}
%   \list{}{\setlength{\labelwidth}{0pt}
%           \setlength{\leftmargin}{\parindent}
%           \setlength{\itemsep}{0.11ex plus 0.11ex}
%           \setlength{\parsep}{0ex}
%           \setlength{\itemindent}{-\parindent}}
%   \def\newblock{\hskip .11em plus .11em minus -.07em}
%   \sloppy\clubpenalty4000\widowpenalty4000
%   \sfcode`\.=1000\relax}
% \let\endthebibliography=\endlist

% Allow for a bibliography of sources of attested examples
\def\thesourcebibliography#1{%
  \section*{Sources of Attested Examples}
  \list{}{\setlength{\labelwidth}{0pt}
          \setlength{\leftmargin}{\parindent}
          \setlength{\itemsep}{0.11ex plus 0.11ex}
          \setlength{\parsep}{0ex}
          \setlength{\itemindent}{-\parindent}}
  \def\newblock{\hskip .11em plus .11em minus -.07em}
  \sloppy\clubpenalty4000\widowpenalty4000
  \sfcode`\.=1000\relax}
\let\endthesourcebibliography=\endlist

\def\@lbibitem[#1]#2{\item[]\if@filesw
      { \def\protect##1{\string ##1\space}\immediate
        \write\@auxout{\string\bibcite{#2}{#1}}\fi\ignorespaces}}

\def\@bibitem#1{\item\if@filesw \immediate\write\@auxout
       {\string\bibcite{#1}{\the\c@enumi}}\fi\ignorespaces}

% ---------------------------------------------------------------------
% Section headings with less space

\def\section{%
    \@startsection{section}{1}{\z@}%
                  {-2.0ex plus -0.5ex minus -0.3ex}%
                  {0.8ex plus 0.3ex minus 0.2ex}%
                  {\large\bf\raggedright}}
\def\subsection{%
    \@startsection{subsection}{2}{\z@}%
                  {-1.4ex plus -0.4ex minus -0.2ex}%
                  {0.6ex plus 0.2ex minus 0.1ex}%
                  {\normalsize\bf\raggedright}}
\def\subsubsection{%
    \@startsection{subsubsection}{3}{\z@}%
                  {-0.8ex plus -0.3ex minus -0.1ex}%
                  {0.4ex plus 0.1ex minus 0.1ex}%
                  {\normalsize\bf\raggedright}}
\def\paragraph{%
    \@startsection{paragraph}{4}{\z@}%
                  {-0.8ex plus -0.3ex minus -0.1ex}%
                  {-1em}%
                  {\normalsize\bf}}
\def\subparagraph{%
```

```
    \@startsection{subparagraph}{5}{\parindent}%
420                 {0.4ex plus 0.3ex minus 0.1ex}%
                    {-1em}%
                    {\normalsize\bf}}


425 % ------------------------------------------------------------------
    % Footnotes

    %\footnotesep 6.65pt %
    %\skip\footins 9pt plus 4pt minus 2pt
430 %\def\footnoterule{\kern-3pt \hrule width 5pc \kern 2.6pt }
    %\setcounter{footnote}{0}

    % ------------------------------------------------------------------
    % Lists and paragraphs
435
    \setlength\parindent{1em}

    \leftmargin 2em \leftmargini\leftmargin \leftmarginii 2em
    \leftmarginiii 1.5em \leftmarginiv 1.0em \leftmarginv .5em \leftmarginvi .5em
440 \labelwidth\leftmargini\advance\labelwidth-\labelsep \labelsep 5pt

    % ------------------------------------------------------------------
    % Floats (figures, tables etc.)
    %
445 % Allow a larger proportion of the column/page to be taken up with
    % floats than the standard classes.  Also discourage the creation of
    % columns/pages containing only floats.

    % Maximum fraction of the page that can be occupied by floats:
450 %
    \renewcommand\topfraction{.9}
    \renewcommand\bottomfraction{.5}
    \renewcommand\dbltopfraction{.9}        % 2-column floats

455 % Minimum fraction of page that can be occupied by text:
    %
    \renewcommand\textfraction{.1}

    % Maximum fraction of a page that can be occupied by floats before a
460 % separate float page is produced:
    %
    \renewcommand\floatpagefraction{0.9}
    \renewcommand\dblfloatpagefraction{.9}  % 2-column floats

465 % ------------------------------------------------------------------
    %
    % Since we're using two columns, lines are short and we can get away
    % with less vertical space between lines, within lists and around
    % various kinds of display.
470 %
    % Normally, these parameters are set in the size option to the class
    % file (standard definitions are in classes.dtx).  Here we want to
    % accommodate 10pt, 11pt and 12pt, so we wrap the definitions in
    % \ifcase.
475 %
```

```
% \normalsize
%
\ifcase\@ptsize%
    \renewcommand{\normalsize}{%                                    10pt
        \@setsize\normalsize{11.3pt}\xpt\@xpt%
        \abovedisplayskip 10\p@\@plus2\p@\@minus5\p@%
        \abovedisplayshortskip\z@\@plus3\p@%
        \belowdisplayshortskip 4\p@\@plus3\p@\@minus3\p@%
        \belowdisplayskip\abovedisplayskip%
        \let\@listi\@listI}%
\or%
    \renewcommand{\normalsize}{%                                    11pt
        \@setsize\normalsize{12.6pt}\xipt\@xipt%
        \abovedisplayskip11\p@\@plus2\p@\@minus4\p@%
        \abovedisplayshortskip\z@\@plus3\p@%
        \belowdisplayshortskip5\p@\@plus3\p@\@minus2\p@%
        \belowdisplayskip\abovedisplayskip%
        \let\@listi\@listI}%
\or%
    \renewcommand{\normalsize}{%                                    12pt
        \@setsize\normalsize{13pt}\xiipt\@xiipt%
        \abovedisplayskip 11\p@ \@plus3\p@ \@minus5\p@%
        \abovedisplayshortskip \z@ \@plus3\p@%
        \belowdisplayshortskip 5\p@ \@plus3\p@ \@minus2\p@%
        \belowdisplayskip\abovedisplayskip%
        \let\@listi\@listI}%
\fi

% \small
%
\ifcase\@ptsize%
    \renewcommand{\small}{%                                         10pt
        \@setsize\small{10.5pt}\ixpt\@ixpt%
        \abovedisplayskip 8\p@ \@plus3\p@ \@minus3\p@%
        \abovedisplayshortskip \z@ \@plus2\p@%
        \belowdisplayshortskip 3\p@ \@plus2\p@ \@minus2\p@%
        \belowdisplayskip\abovedisplayskip%
        \def\@listi{\leftmargin\leftmargini%
                    \topsep 3.5\p@ \@plus1.5\p@ \@minus1.5\p@%
                    \parsep 1.5\p@ \@plus\p@ \@minus\p@%
                    \itemsep \parsep}}%
\or%
    \renewcommand{\small}{%                                         11pt
        \@setsize\small{11.3pt}\xpt\@xpt%
        \abovedisplayskip 9\p@ \@plus2\p@ \@minus4\p@%
        \abovedisplayshortskip \z@ \@plus3\p@%
        \belowdisplayshortskip 5\p@ \@plus2.5\p@ \@minus2.5\p@%
        \belowdisplayskip\abovedisplayskip%
        \def\@listi{\leftmargin\leftmargini%
                    \topsep 5\p@ \@plus2\p@ \@minus2\p@%
                    \parsep 2\p@ \@plus2\p@ \@minus\p@%
                    \itemsep \parsep}}%
\or%
    \renewcommand{\small}{%                                         12pt
        \@setsize\small{12pt}\xipt\@xipt%
        \abovedisplayskip 9\p@ \@plus3\p@ \@minus4\p@%
```

```
        \abovedisplayshortskip \z@ \@plus3\p@%
        \belowdisplayshortskip 5\p@ \@plus2.5\p@ \@minus2\p@%
        \belowdisplayskip\abovedisplayskip%
        \def\@listi{\leftmargin\leftmargini%
                    \topsep 5.5\p@ \@plus2.5\p@ \@minus2.5\p@%
                    \parsep 4\p@ \@plus1.5\p@ \@minus\p@%
                    \itemsep \parsep}}%
  \fi


  % \footnotesize
  %
  \ifcase\@ptsize
      \renewcommand{\footnotesize}{%                                  10pt
          \@setsize\footnotesize{9.3pt}\viiipt\@viiipt%
          \abovedisplayskip 5\p@ \@plus2\p@ \@minus3\p@%
          \abovedisplayshortskip \z@ \@plus\p@%
          \belowdisplayshortskip 2.5\p@\@plus\p@\@minus2\p@%
          \belowdisplayskip\abovedisplayskip%
          \def\@listi{\leftmargin\leftmargini%
                      \topsep 2.5\p@ \@plus\p@ \@minus\p@%
                      \parsep 1.5\p@ \@plus\p@ \@minus\p@%
                      \itemsep \parsep}}%
  \or%
      \renewcommand{\footnotesize}{%                                  11pt
          \@setsize\footnotesize{10.3pt}\ixpt\@ixpt%
          \abovedisplayskip 7\p@ \@plus2\p@ \@minus4\p@%
          \abovedisplayshortskip \z@ \@plus\p@%
          \belowdisplayshortskip 3\p@ \@plus2\p@ \@minus2\p@%
          \belowdisplayskip\abovedisplayskip%
          \def\@listi{\leftmargin\leftmargini%
                      \topsep 3\p@ \@plus2\p@ \@minus2\p@%
                      \parsep 2\p@ \@plus\p@ \@minus\p@%
                      \itemsep \parsep}}%
  \or%
      \renewcommand{\footnotesize}{%                                  12pt
          \@setsize\footnotesize{11pt}\xpt\@xpt%
          \abovedisplayskip 9\p@ \@plus2\p@ \@minus4\p@%
          \abovedisplayshortskip \z@ \@plus3\p@%
          \belowdisplayshortskip 5\p@ \@plus3\p@ \@minus3\p@%
          \belowdisplayskip\abovedisplayskip%
          \def\@listi{\leftmargin\leftmargini%
                      \topsep 4.5\p@ \@plus2\p@ \@minus2\p@%
                      \parsep 3\p@ \@plus\p@ \@minus\p@%
                      \itemsep \parsep}}%
  \fi


  % \large
  %
  \ifcase\@ptsize%
      \renewcommand{\large}{\@setsize\large{13pt}\xiipt\@xiipt}%   10pt
  \or%
      \renewcommand{\large}{\@setsize\large{13pt}\xiipt\@xiipt}%   11pt
  \or%
      \renewcommand{\large}{\@setsize\large{16pt}\xivpt\@xivpt}%   12pt
  \fi
```

```
590
    % \Large
    %
    \ifcase\@ptsize%
        \renewcommand{\Large}{\@setsize\Large{16pt}\xivpt\@xivpt}%  10pt
595  \or%
        \renewcommand{\Large}{\@setsize\Large{16pt}\xivpt\@xivpt}%  11pt
     \or%
        \renewcommand{\Large}{\@setsize\Large{16pt}\xivpt\@xivpt}%  12pt
     \fi
600
    % Leave \scriptsize, \tiny, \huge, \Huge unchanged?

    %
    % Float separations, single and double-column
605 %
    \ifcase\@ptsize%
        \setlength\floatsep{10\p@ \@plus 2\p@ \@minus 2\p@}%       10pt
        \setlength\textfloatsep{16\p@ \@plus 2\p@ \@minus 4\p@}%
        \setlength\intextsep{10\p@ \@plus 2\p@ \@minus 2\p@}%
610     \setlength\dblfloatsep{10\p@ \@plus 2\p@ \@minus 2\p@}%
        \setlength\dbltextfloatsep{16\p@ \@plus 2\p@ \@minus 4\p@}%
     \or%
        \setlength\floatsep{10\p@ \@plus 2\p@ \@minus 2\p@}%       11pt
        \setlength\textfloatsep{16\p@ \@plus 2\p@ \@minus 4\p@}%
615     \setlength\intextsep{10\p@ \@plus 2\p@ \@minus 2\p@}%
        \setlength\dblfloatsep{10\p@ \@plus 2\p@ \@minus 2\p@}%
        \setlength\dbltextfloatsep{16\p@ \@plus 2\p@ \@minus 4\p@}%
     \or%
        \setlength\floatsep{12\p@ \@plus 3\p@ \@minus 3\p@}%       12pt
620     \setlength\textfloatsep{18\p@ \@plus 2\p@ \@minus 4\p@}%
        \setlength\intextsep{12\p@ \@plus 3\p@ \@minus 3\p@}%
        \setlength\dblfloatsep{12\p@ \@plus 2\p@ \@minus 4\p@}%
        \setlength\dbltextfloatsep{18\p@ \@plus 2\p@ \@minus 4\p@}%
     \fi
625
    %
    % Top-level list in \normalsize text
    %
    \ifcase\@ptsize%
630     \def\@listi{\leftmargin\leftmargini%                        10pt
                    \topsep   6\p@ \@plus2\p@ \@minus2\p@%
                    \parsep   2\p@ \@plus0.5\p@ \@minus\p@%
                    \itemsep  2.5\p@ \@plus\p@ \@minus0.5\p@}%
     \or%
635     \def\@listi{\leftmargin\leftmargini%                        11pt
                    \topsep   8\p@ \@plus2\p@ \@minus2\p@%
                    \parsep   3\p@ \@plus1.5\p@ \@minus\p@%
                    \itemsep  3\p@ \@plus1.5\p@ \@minus\p@}%
     \or%
640     \def\@listi{\leftmargin\leftmargini%                        12pt
                    \topsep   9\p@ \@plus3\p@    \@minus4\p@%
                    \parsep   4\p@   \@plus2\p@ \@minus\p@%
                    \itemsep  4\p@   \@plus2\p@ \@minus\p@}%
     \fi
645 \let\@listI\@listi
```

```
%
% Embedded lists unchanged.
%


% ---------------------------------------------------------------------
% End of colacl.sty
% ---------------------------------------------------------------------
```

5 doc/lyapunov-fm.bib

```
@article{Slater98,
author={Slater, Dan},
year={1998},
title={Chaotic {S}ound {S}ynthesis},
journal={Computer Music Journal},
volume={22},
number={2},
pages={12-19}
}

@article{Dewdney91,
author={Dewdney, A. K.},
year={1991},
title={Mathematical {R}ecreations: {L}eaping into {L}yapunov {S}pace},
journal={Scientific American},
volume={265},
pages={178-180}
}

@book{Elert07,
author={Elert, Glenn},
year={2007},
title={The Chaos Hypertextbook},
publisher={\url{http://hypertextbook.com/chaos/}}
}

@book{Falconer03,
author={Falconer, Kenneth},
year={2003},
title={Fractal Geometry: Mathematical Foundations and Applications, Second
    ↪ Edition},
publisher={Wiley}
}

@book{SoftRockEP,
author={ClaudiusMaximus},
year={2005},
title={Soft~Rock~EP},
publisher={\url{http://archive.org/details/ ClaudiusMaximus\_-\_Soft\_Rock\_EP}}
}

@book{SoftRockDVD,
author={ClaudiusMaximus},
year={2006},
title={Soft~Rock~DVD},
publisher={\url{http://archive.org/details/ ClaudiusMaximus\_-\_Soft\_Rock\_DVD
```

```
    ↪ }}
}

@book{OpenGL,
author={Segal, Mark},
author={Akeley, Kurt},
year={2013},
title={The OpenGL Graphics System: A Specification},
publisher={\url{http://www.opengl.org/registry/doc/ glspec43.core.20130214.pdf}}
}

@book{GLSL,
author={Kessenich, John},
author={Baldwin, Dave},
author={Rost, Randi},
author={LunarG},
year={2013},
title={The OpenGL Shading Language},
publisher={\url{http://www.opengl.org/registry/doc/ GLSLangSpec.4.30.8.pdf}}
}

@book{GLUT,
author={Kilgard, Mark J.},
year={1996},
title={The OpenGL Utility Toolkit (GLUT) Programming Interface},
publisher={\url{http://www.opengl.org/documentation/ specs/glut/glut-3.spec.pdf
    ↪ }}
}

@book{JACK,
author={Davis, Paul},
year={2013},
title={The JACK Audio Connection Kit},
publisher={\url{http://jackaudio.org}}
}
```

6 doc/lyapunov-fm.tex

```
\documentclass[11pt,a4paper]{article}
\usepackage{lac2013}
\usepackage{amsmath}
\usepackage{graphicx}
\usepackage{float}
\usepackage{dblfloatfix}
\usepackage{subfigure}
%\usepackage{hyperref}
\newcommand{\url}[1]{\texttt{#1}}
\sloppy
\newenvironment{contentsmall}{\small}

\DeclareMathOperator{\inc}{I}
\DeclareMathOperator{\wrap}{\%}
\DeclareMathOperator{\SR}{SR}

\title{Lyapunov Space of Coupled FM Oscillators}
\author
{Claude Heiland-Allen
```

```
20  \\ claude@mathr.co.uk
    }

    \begin{document}
    \maketitle
25
    \begin{abstract}
    \begin{contentsmall}
    Consider two coupled oscillators, each modulating the other's
    frequency. This system is governed by four parameters: the base
30  frequency and modulation index for each oscillator. For some
    parameter values the system becomes unstable. The Lyapunov
    exponent is used to measure the instability. Images of the
    parameter space are generated, with the number crunching implemented on graphics
    hardware using OpenGL. The mouse position over the displayed
35  image is linked to realtime audio output, creating an audio-visual browser
    for the 4D parameter space.
    \end{contentsmall}
    \end{abstract}

40  \keywords{
    \begin{contentsmall}
    chaos, DSP, GPU
    \end{contentsmall}
    }
45
    \begin{figure}[H]
    \centering
    \includegraphics[width=\textwidth]{example.png}
    \caption{Example output.}
50  \label{example}
    \end{figure}

    \section{Introduction}
    Soft Rock EP~\cite{SoftRockEP} and Soft Rock DVD~\cite{SoftRockDVD}
55  explored the transitions between order and chaos in coupled FM
    oscillators. A more recent continuation of
    this project is to make a map of the parameter space of coupled FM oscillators ↙
        ↳ on a perceptually
    relevant level and use it in live performance, choosing parameters on the basis
    of desired sound character.
60
    A bifurcation diagram produced by an analogue Moog
    synthesizer~\cite{Slater98} and images of Lyapunov fractals~\cite{Dewdney91}
    were inspiration to apply the latter technique to the parameter space
    of coupled FM oscillators in the digital realm.
65
    \newpage

    \section{Formulation}

70  \subsection{Coupled FM Oscillators}
    \begin{figure}[H]
    \centering
    \includegraphics[width=0.4\textwidth]{figure1.png}
    \caption{Coupled FM oscillators in Pure-data.}
75  \label{fig:pd}
```

```
\end{figure}
Consider the two coupled oscillators in Figure~\ref{fig:pd}.
Pure-data's model of interconnected components each with their
own internal state maps poorly to GPU architecture.  Considering
the whole system as one and flattening the internal state into
a single phase space vector leads to the following formulation
as a mutual recurrence relation:
\begin{equation}\begin{aligned}
x_{n+1} &= \wrap(x_{n} + \inc(f_x + m_x \cos (2 \pi y_{n-d}))) \\
y_{n+1} &= \wrap(y_{n} + \inc(f_y + m_y \cos (2 \pi x_{n-d})))
\end{aligned}\label{eq:osc}\end{equation}
where
\begin{equation*}\wrap(t) = t - \lfloor t \rfloor ,\quad \inc(t) = \frac{440}{\SR} 2^{\frac{t - 69}{12}}\end{equation*}
Here $x_n$, $y_n$ is the phase of each oscillator at time step $n$,
$d$ is a delay measured in samples, $f_x$, $f_y$ is the base frequency
of each oscillator as a MIDI note number, and $m_x$, $m_y$ is the
modulation index of each oscillator as a MIDI note number.  $\wrap(t)$
performs wrapping into $[0,1)$, with $\lfloor t \rfloor$ being the
flooring operation (the largest integer not greater than $t$).

The four-dimensional parameter space vector will be written
\begin{equation*}a=(f_x,f_y,m_x,m_y)\end{equation*}
and the $(2d+2)$-dimensional phase space vector
\begin{equation*}z=(x_n,y_n,x_{n-1},y_{n-1},\ldots,x_{n-d},y_{n-d})\end{equation*}
with sample rate $\SR = 48000$.  For reasons
explained in Section~\ref{sec:glissues}, $d=1$ will be fixed.

\subsection{Lyapunov Exponents}

Lyapunov exponents can be used to measure the stability (or otherwise)
of a dynamical system.
A good introduction is found in Chapter~4.3 \emph{Lyapunov Exponent}~\cite{Elert07}.
The definition is covered in Chapter~13.7 \emph{Liapounov exponents and entropies}~\cite{Falconer03}
which also relates it to measures of fractal dimension.

The Lyapunov exponent $\lambda$ measures divergence in phase space:
\begin{equation*}\left| z_1(t) - z_0(t) \right| \approx e^{\lambda t} \left| z_1(0) - z_0(0) \right|\end{equation*}
\begin{equation}\lambda = \lim_{\begin{array}{c} t \to \infty \\
z_1(0) \to z_0(0)\end{array}} \frac{1}{t} \log{\frac{\left|z_1(t) - z_0(t)\right|}{\left|z_1(0) - z_0(0)\right|}}\label{eq:lexp}\end{equation}
An attracting orbit has $\lambda < 0$ and a divergent (chaotic) orbit
has $\lambda > 0$.

A modified norm is required to take into account
the wrapping of phase into $[0,1)$:
\begin{equation*}\left|z\right|_{\wrap} = \sqrt{\sum_{i} {\left(\min(\wrap(z_i), 1 - \wrap(z_i))\right)^2}}\end{equation*}
For example the distance between $0.1$ and $0.9$ is properly $0.2$ (not $0.8$)
because $0.1$ can be phase-unwrapped to $1.1$.

\subsection{Viewing Planes}
```

```
An image is 2D, which requires choosing a subset of the 4D parameter
space to visualize. Two particular planes were chosen:
\begin{equation}\begin{aligned}
A_+(a_0, r_0) &= a_0 + r_0 \left(\begin{array}{rr}
1 & 0 \\
\phantom{-}1 & 0 \\
0 & 1 \\
0 & \phantom{-}1 \end{array}\right)\left(\begin{array}{c}u \\
v \end{array}\right) \\
A_-(a_0, r_0) &= a_0 + r_0 \left(\begin{array}{rr}
 1 &  0 \\
-1 &  0 \\
 0 &  1 \\
 0 & -1 \end{array}\right)\left(\begin{array}{c}u \\
v \end{array}\right)\end{aligned}\label{eq:view}\end{equation}
where $(u,v)$ is the coordinates of the pixel, $a_0$ is the
centre of the view, and $r_0$ is the radius of the view.

These planes were chosen because they are simple, while still being
flexible enough to explore the whole 4D space.
The $A_+$ plane varies both oscillators in the same direction, while
the $A_-$ plane varies each oscillator in opposite directions. To
center on a particular target point $(f_x, f_y, m_x, m_y)$ one might
use the $A_+$ plane to center on the midpoint
\begin{equation*}\left(\frac{f_x + f_y}{2}, \frac{f_x + f_y}{2}, \frac{m_x + m_y
    ↳ }{2}, \frac{m_x + m_y}{2}\right)\end{equation*}
and then switch to the $A_-$ plane to break the $(x,y)$ symmetry.

\section{Implementation}

The implementation uses OpenGL~\cite{OpenGL} and OpenGL Shading Language~\cite{
    ↳ GLSL}
for computation and graphical rendering, GLUT~\cite{GLUT} for windowing
and input event handling, and JACK~\cite{JACK} for audio output.

\subsection{Introduction to Modern OpenGL}

Modern OpenGL has a programmable shader pipeline. Vertex attributes are read
from vertex buffers and processed by vertex shaders. The outputs
of the vertex shader (called varyings) are further manipulated by an
optional geometry shader stage. Geometry shaders can output a different
vertex count to their input count, whereas vertex shaders are one-in
one-out. The result of the geometry shader can be captured into another
vertex buffer using transform feedback. Following the geometry
shader the primitives (points or triangles) are rasterized, and
varyings interpolated across each primitive. Finally a fragment
shader takes these values and computes the colour at that pixel. The
output of a fragment shader can be captured by attaching a texture to
a framebuffer.

\subsection{Computation Overview}

To render an image a texture is first filled with $(u,v)$ coordinates
using a framebuffer object and a fragment shader. This texture is copied
to a vertex buffer object, interleaved with an initial phase space vector
$z=(0,0,0,0)$ and Lyapunov exponent statistics vector $l=(0,0,0,0)$ for
each point.
```

```
Using a vertex shader, $a$ is calculated from $(u,v)$ using Equation~\ref{eq:view}, and then
a rough estimate of the Lyapunov exponent is computed using
Equation~\ref{eq:lexp} by perturbing $z_1(0) = z_0(0) + \delta$ with
$\delta$ small and performing $t = 256$ iterations of Equation~\ref{eq:osc}.
The first few repetitions are discarded, along with those resulting in $-\infty$,
and the rough $\lambda$ estimates are accumulated in $l$.

Between each repetition the working set is compacted using a geometry
shader.  Points whose mean Lyapunov exponent estimate changed
very little during the previous step are plotted and removed from the
working set.  The other points are kept to be refined
further, directing the computational effort on the points that
need it most:~those slow to converge.

To ensure user interface responsiveness, the computation is amortized
over several frames.  The target frame period is divided by the measured
time for one repetition to compute how many repetitions to perform that
frame.  The repetitions-per-frame increases as the working set becomes
smaller.

\subsection{Noise Increases Stability}

At the end of each repetition $z_1$ is kept instead of $z_0$.  This
effectively adds a small amount of noise, counter-intuitively
increasing stability.  Noise allows more of the phase space to be
explored, and makes it more likely for the perturbed orbit to reach an
attracting part of the phase space.

\subsection{Dither Increases Quality}

To reduce grid sampling artifacts, $(u,v)$ is perturbed within the bounds
of its corresponding pixel before calculating the $a$ parameter vector
for each repetition.

\section{Results}

\begin{figure*}[ht!]
\centering
\begin{tabular}{cc}
\subfigure[$A_+((120, 120, 0, 0), 72)$]{\includegraphics[width=3in]{example01.png}\label{ex:01}} &
\subfigure[$A_-((120, 120, 0, 0), 72)$]{\includegraphics[width=3in]{example02.png}\label{ex:02}} \\
\subfigure[$A_+((95.2, 95.2, 32.6, 32.6), 4.5)$]{\includegraphics[width=3in]{example03.png}\label{ex:03}} &
\subfigure[$A_-((95.2, 95.2, 32.6, 32.6), 4.5)$]{\includegraphics[width=3in]{example04.png}\label{ex:04}} \\
\subfigure[$A_+((151.57, 151.57, 1.64, 1.64), 0.07)$]{\includegraphics[width=3in]{example05.png}\label{ex:05}} &
\subfigure[$A_-((151.57, 151.57, 1.64, 1.64), 0.07)$]{\includegraphics[width=3in]{example06.png}\label{ex:06}} \\
\subfigure[$A_-((117.0, 148.4, 20.4, 2.7), 1.8)$]{\includegraphics[width=3in]{example07.png}\label{ex:07}} &
\subfigure[$A_+((103.65, 108.41, 33.42, 10.93), 0.14)$]{\includegraphics[width=3
```

```
    ↪ in]{example09.png}\label{ex:09}}
\end{tabular}
\caption{Example images.  Darker shades are stable, lighter shades chaotic.}
\label{fig:examples1}
\end{figure*}
%\begin{figure}[ht]
%\centering
%\subfigure[$A_-((117.0, 148.4, 20.4, 2.7), 1.8)$]{\includegraphics[width=3in]{↙
    ↪ example07.png}\label{ex:07}}
%\subfigure[$A_-((141.46, 146.22, 22.76, 0.27), 0.14)$]{\includegraphics[width=3↙
    ↪ in]{example08.png}\label{ex:08}}
%\subfigure[$A_+((103.65, 108.41, 33.42, 10.93), 0.14)$]{\includegraphics[width↙
    ↪ =3in]{example09.png}\label{ex:09}}
%\subfigure[$A_-((89.8, 137.5, -17.5, -7.1), 3.7)$]{\includegraphics[width=3in]{↙
    ↪ example10.png}\label{ex:10}}
%\caption{More examples.}
%\label{fig:examples2}
%\end{figure}

\subsection{Examples}

Figure~\ref{ex:01} shows the initial view on starting the interactive
browser.  Low frequencies to the left are stable even at high
modulation index away from the central axis.  High frequencies to the
right become chaotic at progressively lower modulation index.
\subref{ex:02} shows the $A_-$ plane at the same location.
\subref{ex:03} shows bands alternating between stability and
chaos.  The bands become distorted and collapse as the modulation
index and frequency increase.  \subref{ex:04} shows its $A_-$
plane, bands become rings.
When the frequency is greatly increased, the shapes become more
intricate.  \subref{ex:05} exhibits spirals of stability, with
similar spirals in the $A_-$ plane in \subref{ex:06}.

When $f_x=f_y$ and $m_x=m_y$ the $A_+$ plane has mirror symmetry about its
horizontal axis, and the $A_-$ plane has two-fold rotational symmetry
about its centre.
Breaking the symmetry and setting $f_x \ne f_y$ or $m_x \ne m_y$ leads
to diverse forms.  In particular Figure~\ref{ex:09} has shapes that
resemble those of Lyapunov space images of the logistic map.

\subsection{Interactive Explorer}

The implementation is an interactive audio-visual explorer for the parameter
space of coupled FM oscillators.
Clicking with the mouse zooms the view about the clicked point.
The left button (or scroll up) zooms in, the right button (or scroll
down) zooms out, the middle button centers the view on the target point.
Pressing the TAB key toggles between the $A_+$ and $A_-$ planes in
Equation~\ref{eq:view}, and F11 toggles full screen operation.

While the GPU simulates and analyses one oscillator pair per pixel, the
CPU simulates one oscillator pair with $a$ determined from the pixel
under the mouse pointer.  The image acts as a map,
a reference frame for chosing parameters to audition by moving the mouse.

\section{Conclusions}
```

```
\subsection{Original Intent}

Earlier experiments used one Pure-data batch mode instance per CPU
core each sending analysis data to a realtime Pure-data instance.
The analysis used various methods (including FFT for spectral
statistics and the sigmund external for pitch tracking) to classify
points into pitched (ordered, stable) or unpitched (chaotic, unstable)
with measures of distortion or noisiness. Sadly this approach proved
impractical as it achieved only tens of pixels per second, even
with a fast multi-core CPU, and porting these signal analysis algorithms
to massively-parallel programmable graphics hardware seemed to be too
difficult.

\subsection{OpenGL Issues \label{sec:glissues}}

The current implementation is hardcoded with delay $d=1$ and would be
very awkward to generalize. OpenGL architecture limits each vertex
attribute to four components with the maximum number of attributes
typically limited to sixteen. This totals 64 floats per vertex, 6 of
which are needed for the pixel coordinates and Lyapunov exponent
statistics accumulation. Therefore using OpenGL imposes a limit
$d < 28$. For comparison the original experiments in
\emph{Soft Rock EP} used Pure-data's default block size of 64, with $d = 32$.
Moreover, increasing $d$ increases video memory consumption. With
the maximum $d = 27$, browsing at $1920 \times 1080$ resolution would
require over 1GB.

Future work on this project will look into using OpenCL, which
provides a heterogenous CPU and GPU computation framework, in the hope that
it will avoid the inherent awkwardness of abusing OpenGL shaders to
perform calculations.

\subsection{Audio Issues}

While the implementation works as intended, with $d=1$
the sound is nowhere near as rich and varied as with $d=32$. With
small $d$ there is much more very high frequency content in
interesting-looking regions. There seem to be few if any regions of the
parameter space with both interesting appearance and palatable audio
frequencies at $d=1$, while with high $d$ there are
parameters that generate sounds that fluctuate intermittently between
smooth tones and noise. Visualization with high $d$ has not been
possible so far, so whether their neighbourhoods
look as interesting as they sound remains an openquestion .

Unfortunately, heavy use of the GPU in the interactive browser can
block the operating system for too long and cause audible glitches
(JACK xruns). This situation may change as free drivers continue to
improve, allowing use of the browser in a live situation.

\subsection{Pretty Pictures}

Despite these shortcomings, I think the images look good.
I plan to render a selection at high resolution and print
postcards and posters. For huge images it is possible to divide the image plane↙
    ↳ into tiles and
```

```
compute each tile in succession, finally combining the pieces
into one large picture.

There is also scope for video work, moving and rotating the viewing
plane through the 4D parameter space, with different shapes forming
and collapsing over time.  Rough benchmarks take 5-10 seconds
per frame at $1920 \times 1080$, so it seems sensible to wait until faster ↙
    ↳ cheaper
graphics cards become available.

\section{Obtaining the Implementation}

The implementation was written on GNU/Linux Debian Wheezy running on a quad-core↙
    ↳ AMD64 processor
with NVIDIA GTX~550Ti graphics card using
proprietary drivers.  The source code is available at:
\url{http://code.mathr.co.uk/lyapunov-fm}


\section{Acknowledgements}

Thanks to the anonymous reviewers for their constructive criticism
on a number of issues, and to Rob Canning, Adnan Hadzi, and
Joanne Seale for their helpful feedback on earlier versions of this
paper.

\bibliographystyle{acl}
\bibliography{lyapunov-fm}

\end{document}
```

7 doc/Makefile

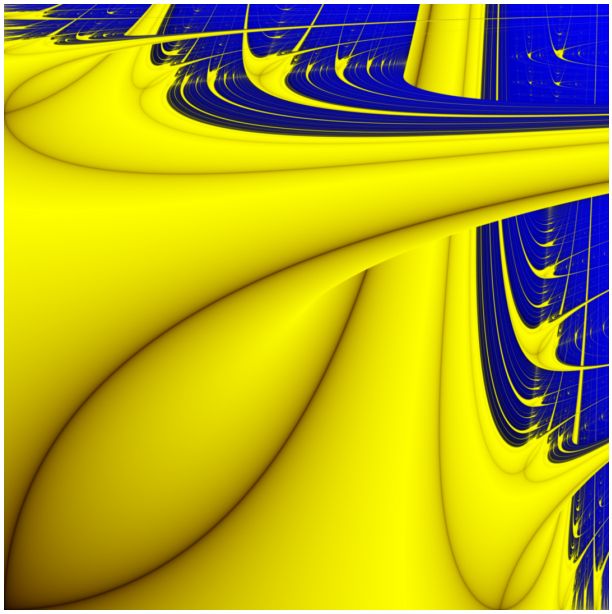
```
PAPER=lyapunov-fm

all: $(PAPER).pdf

clean: cleantmp
	rm -f $(PAPER).pdf

cleantmp:
	rm -f $(PAPER).aux
	rm -f $(PAPER).bbl
	rm -f $(PAPER).blg
	rm -f $(PAPER).log

%.pdf: %.tex %.bib
	pdflatex "$<"
	bibtex `basename "$<" .tex`
	pdflatex "$<"
	pdflatex "$<"

.PHONY: all clean
```

8 doc/slides/bifurcation2d.c

```c
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

#define W (1080*4)
#define H (1080*4)
#define N 2500
#define A 1

int main(int argc, char **argv) {
  float lmin =  1000;
  float lmax = -1000;
  printf("P6\n%d %d\n255\n", W, H);
  for (int y = H - 1; y >= 0; --y) {
    for (int x = 0; x < W; ++x) {
      float l = 0;
      int n = 0;
      for (int dx = 0; dx < A; ++ dx) {
        for (int dy = 0; dy < A; ++ dy) {
          float rB = 2 + (x + dx / (float) A) * 2 / W;
          float rA = 2 + (y + dy / (float) A) * 2 / H;
          float z = 0.5;
          for (int i = 0; i < 250; ++i) {
            z = rA * z * (1 - z);
            z = rB * z * (1 - z);
          }
          for (int i = 0; i < N; ++i) {
            z = rA * z * (1 - z);
            float dz = fabsf(rA * (1 - 2 * z));
            if (dz > 0) {
              l += logf(dz);
              n += 1;
            }
            z = rB * z * (1 - z);
            dz = fabsf(rB * (1 - 2 * z));
            if (dz > 0) {
              l += logf(dz);
              n += 1;
            }
          }
        }
      }
      float v = -100;
      if (n != 0) {
        l /= n;
        v = l;
        lmin = fminf(lmin, l);
        lmax = fmaxf(lmax, l);
      }
      float fr, fg, fb;
      if (v <= 0) {
        fr = (1 + tanhf(v / 4)) * 255;
        fg = (1 + tanhf(v / 3)) * 255;
        fb = 0;
      } else {
        fr = 0;
        fg = 0;
```

```
        fb = tanhf(3 * v) * 255;
      }
/*
      int cr = fminf(fmaxf((255 * 0.6 * (1.0 - tanhf(4.0 * (v - 0.3333))) - 64) ↙
          ↳ * 255.0 / 191, 0), 255);
      int cg = fminf(fmaxf((255 * (2.0 * v * v - 2.5 * v + 1) - 64) * 255.0 / ↙
          ↳ 191, 0), 255);
      int cb = fminf(fmaxf((255 * v * v - 64) * 255.0 / 191, 0), 255);
*/
#define O(c) putchar(fminf(fmaxf(c, 0), 255))
      O(fr);O(fg);O(fb);
    }
  }
  fprintf(stderr, "%f %f\n", lmin, lmax);
  return 0;
}
```

9 doc/slides/bifurcation-2d.png

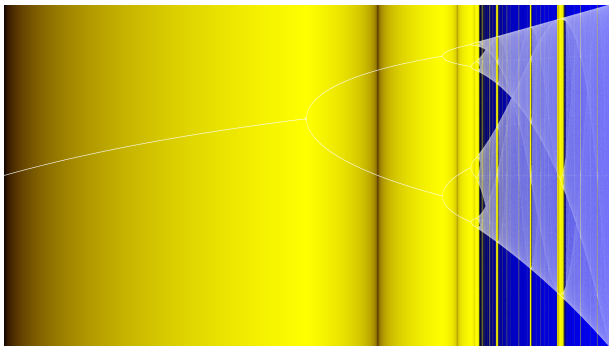


10 doc/slides/bifurcation.c

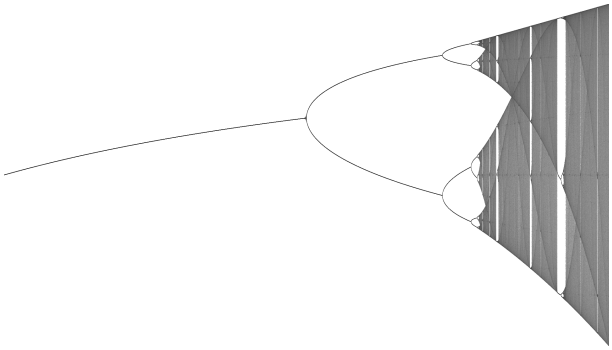
```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

#define W 1920
#define H 1080
#define N 50000
#define A 16

int main(int argc, char **argv) {
  int *m = calloc(1, W * H * sizeof(int));
  unsigned char *grey = calloc(1, W * H);
  unsigned char *rgb = calloc(1, W * H * 3);
```

```
  for (int x = 0; x < W; ++x) {
    float l = 0;
    int n = 0;
    for (int dx = 0; dx < A; ++ dx) {
      float r = 2 + (x + dx / (float) A) * 2 / W;
      float z = 0.5;
      for (int i = 0; i < 500; ++i) {
        z = r * z * (1 - z);
      }
      for (int i = 0; i < N; ++i) {
        z = r * z * (1 - z);
        float dz = fabsf(r * (1 - 2 * z));
        if (dz > 0) {
          l += logf(dz);
          n += 1;
        }
        float dither = rand() / (float) RAND_MAX - 0.5f;
        int y = fminf(fmaxf(H * (1 - z) + dither, 0), H - 1);
        m[y * W + x]++;
      }
    }
    if (n != 0) {
      l /= n;
    } else {
      l = -100;
    }
    float v = 0.5 + 0.5 * tanh(3.0 * l);
    int cr = fminf(fmaxf((255 * 0.6 * (1.0 - tanh(4.0 * (v - 0.3333))) - 30) * ↙
        ↳ 255.0 / 225, 0), 255);
    int cg = fminf(fmaxf((255 * (2.0 * v * v - 2.5 * v + 1) - 30) * 255.0 / 225,↙
        ↳  0), 255);
    int cb = fminf(fmaxf((255 * v * v - 30) * 255.0 / 225, 0), 255);
    for (int y = 0; y < H; ++ y) {
      float ca = log2f(1 + m[y * W + x]) / log2f(1 + A * N);
      grey[y * W + x] = 255 * (1 - ca);
      rgb[3 * (y * W + x) + 0] = 255 * ca + (1 - ca) * cr;
      rgb[3 * (y * W + x) + 1] = 255 * ca + (1 - ca) * cg;
      rgb[3 * (y * W + x) + 2] = 255 * ca + (1 - ca) * cb;
    }
  }
  printf("P5\n%d %d\n255\n", W, H);
  fwrite(grey, W * H, 1, stdout);
  printf("P6\n%d %d\n255\n", W, H);
  fwrite(rgb, W * H * 3, 1, stdout);
  return 0;
}
```

11 doc/slides/bifurcation-colour.png



12 doc/slides/bifurcation-grey.png



13 doc/slides/colourize.c

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char **argv) {
  int count = 4096 * 4096;
  unsigned short *pgm = malloc(count * 2);
  FILE *f = fopen(argv[1], "rb");
  fseek(f, -count * 2, SEEK_END);
  fread(pgm, count * 2, 1, f);
  fclose(f);
  unsigned short mi = 65535, ma = 0;
  for (int i = 0; i < count; ++i) {
    if (mi > pgm[i]) mi = pgm[i];
    if (ma < pgm[i]) ma = pgm[i];
  }
  unsigned char *ppm = malloc(count * 3);
  int k = 0;
  for (int i = 0; i < count; ++i) {
    float g = pgm[i];
    if (g > 65536/2) {
      g -= 65536/2;
      g /= (ma - 65536/2);
```

```
      g *= 0.5;
      g += 0.5;
    } else {
      g -= mi;
      g /= (65536/2 - mi);
      g *= 0.5;
    }
    float x = 0.6 * (1.0 - tanh(4.0 * (g - 1.0/3.0)));
    float y = 2.0 * g * g - 2.5 * g + 1.0;
    float z = g * g;
    x -= 0.25; x /= 0.75; x = fmin(fmax(x, 0), 1); x = pow(x, 0.25);
    y -= 0.25, y /= 0.75; y = fmin(fmax(y, 0), 1); y = pow(y, 0.25);
    z -= 0.25, z /= 0.75; z = fmin(fmax(z, 0), 1); z = pow(z, 0.25);
    ppm[k++] = 255 * x;
    ppm[k++] = 255 * y;
    ppm[k++] = 255 * z;
  }
  f = fopen(argv[2], "wb");
  fprintf(f, "P6\n4096 4096\n255\n");
  fwrite(ppm, count * 3, 1, f);
  fclose(f);
  return 0;
}
```

14 doc/slides/inc-equation.png

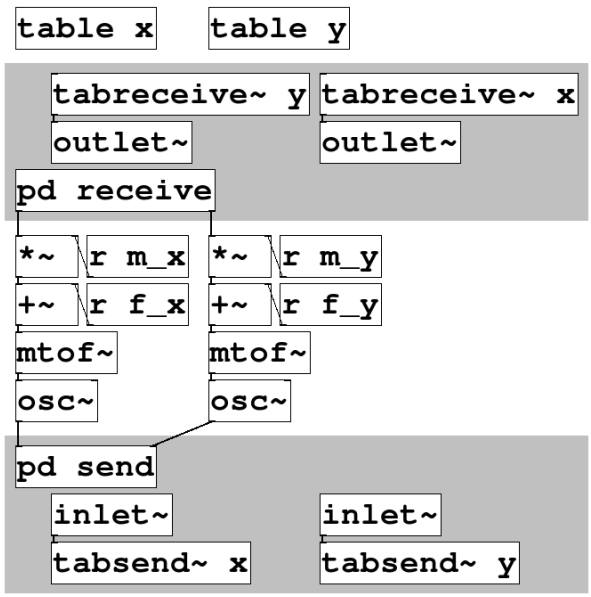
$$\operatorname{inc}(n) = \frac{440}{\mathrm{SR}} 2^{\frac{n-69}{12}}$$

15 doc/slides/maths.tex

```
\documentclass[11pt,a4paper]{article}
\usepackage{amsmath}
\usepackage{mathtools}

\begin{document}

\[
\begin{aligned}
\begin{matrix*}[r]
x_{j+1} \\
y_{j+1}
\end{matrix*} &=
\operatorname{wrap}\left(
\begin{matrix*}[r]
x_j \\
y_j
\end{matrix*} +
\operatorname{inc}\left(
\begin{matrix*}[r]
f_x \\
f_y
```

```
\end{matrix*} +
\begin{matrix*}[r]
m_x \\
m_y
\end{matrix*} \cos\left(2 \pi
\begin{matrix*}[r]
y_{j-d} \\
x_{j-d}
\end{matrix*} \right) \right) \right)
\\
\operatorname{inc}\left(n\right) &=
\mbox{\small$\frac{440}{\operatorname{SR}}$} 2^{\mbox{\Large$\frac{n - 69}{12}$}}
\end{aligned}\\
\mbox{\Huge$\infty$}
\]

\end{document}
```

16 doc/slides/puredata.png



17 doc/slides/recurrence-equation.png

$$\begin{matrix} x_{j+1} \\ y_{j+1} \end{matrix} = \operatorname{wrap}\left(\begin{matrix} x_j \\ y_j \end{matrix} + \operatorname{inc}\left(\begin{matrix} f_x \\ f_y \end{matrix} + \begin{matrix} m_x \\ m_y \end{matrix} \cos\left(2\pi \begin{matrix} y_{j-d} \\ x_{j-d} \end{matrix}\right)\right)\right)$$

18 doc/slides/slides.hs

```
{-# LANGUAGE NoMonomorphismRestriction #-}
module Main (main) where

import Diagrams.Prelude
```

```
import Diagrams.Backend.Cairo.CmdLine (defaultMain)

import Control.Monad (forM_)
import Data.List (intersperse)
import Data.Maybe (fromJust)
import System.Environment (withArgs)

main = do
  forM_ (diagrams slides `zip` [100..]) $ \(d, i) ->
    withArgs ["-o", "o/" ++ tail (show i ++ ".png"), "-h", "1080"] $
      defaultMain d

diagrams = map style . presentation

slides =
  ( "Lyapunov FM"
  ,[( "Background"
     ,[( "Recurrence Relations"
       ,[( "Recurrence"
         , [ "Next state depends on current state."
           , "z(t+1) = F(z(t))"
       ]),("Phase Space"
         , [ "Internal state that varies over time."
           , "z(0) determines all future z(t)."
       ]),("Parameter Space"
         , [ "Constant over time."
           , "Family of recurrences."
       ]),("Example: Logistic Map"
         , [ "z(t+1) = r z(t) (1 - z(t))"
           , "1D phase space: z(t)"
           , "1D parameter space: r"
       ])]
     ),( "Behaviour Of Recurrences"
       ,[( "Orbits"
         , [ "The set of points reached from z(0)."
           , "{ z(0), z(1), z(2), ... }"
       ]),("Attractors"
         , [ "Orbits may become periodic."
           , "Many points can reach one attractor."
       ]),("Stability"
         , [ "Nearby points are pulled closer together."
       ]),("Chaos"
         , [ "Nearby points are pushed further apart."
           , "Butterfly effect."
       ]),("Lyapunov Exponent"
         , [ "Quantifies butterfly effect."
           , "< 0 : stable"
           , "> 0 : chaotic"
       ])]
     ),( "Mapping Behaviours"
       ,[( "1D Logistic Map"
         , [ "z(t+1) = F(z(t))"
           , "F(x) = r x (1 - x)"
           , "1D phase space: z(t)"
           , "1D parameter space: r"
       ]),("1D Bifurcation Diagram"
         , [ "#LogisticBifurcation"
```

```
        ]),("1D Lyapunov Space"
          , [ "#LogisticLyapunov1D"
        ]),("2D Logistic Map"
          , [ "z(t+1) = G(F(z(t)))"
            , "F(x) = A x (1 - x)"
            , "G(x) = B x (1 - x)"
            , "1D phase space: z(t)"
            , "2D parameter space: (A,B)"
        ]),("2D Lyapunov Space"
          , [ "#LogisticLyapunov2D"
        ])])
    ]),("Implementation"
      ,[( "Coupled FM Oscillators"
        ,[( "Pure-data Implementation"
          , [ "#Pure-data"
        ]),("Recurrence Relation"
          , [ "x(t+1) = F(fx, mx, x(t), y(t-d))"
            , "y(t+1) = F(fy, my, y(t), x(t-d))"
            , "F(f,m,u,v) = wrap(u + I(f + m cos(2 pi v)))"
            , "I(n) = 440/SR 2^((n - 69)/12) "
        ]),("4D Parameter Space"
          , [ "Base frequency (oscillator X)"
            , "Base frequency (oscillator Y)"
            , "Modulation index (oscillator X)"
            , "Modulation index (oscillator Y)"
        ])]
      ),( "GPU Overview"
        ,[( "OpenGL Pipeline"
          , [ "#OpenGLPipeline"
        ]),("OpenGL Glossary"
          , [ "shader: stage in the pipeline"
            , "program: a linked set of shaders"
            , "uniform: parameters for a program"
            , "attribute: per-vertex shader input"
            , "varying: data passed between shaders"
            , "texture: 2D array stored in GPU memory"
            , "buffer: 1D array stored in GPU memory"
        ]),("Hardware Limits"
          , [ "16 attributes"
            , "4 components per attribute"
            , "2(d+1) components for phase space vector"
            , "5 components needed elsewhere"
            , "d <= 28 with optimal packing"
        ]),("Buffer Memory Layout"
          , [ "#Memory"
        ]),("Vertex Attribute Pointers"
          , [ "#Pointers"
        ])]
      ),( "GPU Details"
        ,[( "Computation Structure"
          , [ "#Compute"
        ]),( "Fill RGBA Texture with uv00"
          , [ "#Fill"
        ]),("Copy RG Channels to Buffer"
          , [ "#Copy"
        ]),("Initialize State"
          , [ "#Init"
```

```
    --       ]),("Step: Unpack State"
120 --         , ["#Unpack"
           ]),("Step Recurrence"
             , ["#Step"
    --       ]),("Step: Pack State"
    --         , ["#Pack"
125        ]),("Prune Working Set"
             , ["Memory bandwidth limits speed"
               , "Some points converge faster"
               , "Geometry shader culls converged points"
               , "Working set becomes leaner"
130        ]),("Plot Points"
             , ["Render to framebuffer texture"
               , "gl_FragColor = raw L statistics"
           ]),("Colour Image"
             , ["Read raw L statistics from texture"
135            , "gl_FragColor = F(1)"
               , "stable -> yellow"
               , "chaotic -> blue"
           ])]
       ),("Lyapunov Exponents"
140      ,[("Informal Definition"
             , ["|Delta z(t)| = exp(L t) |Delta z(0)|"
               , "L < 0 stable; nearby values pulled closer"
               , "L > 0 chaotic; nearby values pushed apart"
           ]),("Formal Definition"
145        , ["L = 1/t log |Delta z(t)|/|Delta z(0)|"
             , "Take limits as t -> infinity and Delta z(0) -> 0"
           ]),("Calculation in 1D"
             , ["Inner limit become derivative"
               , "Recurrence derivative is product of step derivatives"
150            , "Log of product is sum of logs"
           ]),("Calculation in >1D"
             , ["Inner limit becomes Jacobian"
               , "Outer limit defines a matrix"
               , "Log of matrix eigenvalues are Lyapunov spectrum"
155            , "Maximal Lyapunov exponent measures stability"
           ]),("Numerical Calculation"
             , ["Pick a point on the attractor z(0)"
               , "Pick a nearby point w(0)"
               , "Run recurrence on both until t large"
160            , "Compute L = 1/t log |z(t)-w(t)|/|z(0)-w(0)|"
               , "Repeat, averaging all the results"
           ])])
     ]),("Conclusions"
        ,[("Examples"
165        ,[("d = 0"
             , ["#Example0"
           ]),("d = 1"
             , ["#Example1"
           ]),("d = 4"
170          , ["#Example4"
           ]),("d = 16"
             , ["#Example16"
           ])]
       ),("Evaluation"
175      ,[("Good Points"
```

```
          , [ "Operation at 0 <= d <= 27"
            , "Experimental fractional d (interpolation)"
            , "Batch mode for offline rendering"
            , "Tiled rendering of huge images"
        ]),("Bad Points"
          , [ "JACK xruns with heavy GPU load"
            , "Free drivers might be better than proprietary?"
            , "Perceptual (ir)relevance of Lyapunov exponent"
            , "Tiled rendering random seed glitches"
            , "Long video render times even with fast hardware"
        ])]
      ),( "Ends"
        ,[( "Links"
          , [ "http://mathr.co.uk/lyapunov-fm"
            , "claude@mathr.co.uk"
  ])])])])

images =
  [("LogisticBifurcation", d'bifur)
  ,("LogisticLyapunov1D", d'bifurc)
  ,("LogisticLyapunov2D", d'bifurc2)
  ,("Pure-data", d'pd)
  ,("OpenGLPipeline", d'opengl)
  ,("Compute", d'compute)
  ,("Memory", d'memory)
  ,("Pointers", d'pointers)
  ,("Fill", d'fill)
  ,("Copy", d'copy)
  ,("Init", d'init)
  ,("Unpack", d'unpack)
  ,("Step", d'step)
  ,("Pack", d'pack)
  ,("Example0", hcat [image "0_pos.png" 9 9, strutX 1, image "0_neg.png" 9 9] # centerXY)
  ,("Example1", hcat [image "1_pos.png" 9 9, strutX 1, image "1_neg.png" 9 9] # centerXY)
  ,("Example4", hcat [image "4_pos.png" 9 9, strutX 1, image "4_neg.png" 9 9] # centerXY)
  ,("Example16", hcat [image "16_pos.png" 9 9, strutX 1, image "16_neg.png" 9 9] # centerXY)
  ]

presentation (t, ss) = ([], mempty) : ([t], bullets (map fst ss)) : concatMap (section [t]) ss

section ts (t, ss) = (t:ts, bullets (map fst ss)) : concatMap (subsection (t:ts)) ss

subsection ts (t, ss) = (t:ts, bullets (map fst ss)) : concatMap (slide (t:ts)) ss

slide ts (t, ss) = [(t:ts, bullets ss)]

bullet ('#':im) = fromJust $ im `lookup` images
bullet s = llabel s (0, 0) # scale 1.5 # translate (r2(-11.5, 0))
```

```
225 bullets ss = (vcat . intersperse (llabel "" (0,0)) . map bullet) ss # alignT # ↙
      ↳ translate (r2(0, 3))

    style (ts, d)
      = (d <> (frame ts # centerXY))
      # font "LMSans10"
230   # lw 0.05
      # lc black
      # lineCap LineCapRound
      # lineJoin LineJoinRound
      # fc white
235   # centerXY
    -- # pad 1.1
      # bg white

    frame [] = mconcat
240   [ alignedText 0 0.5 "Lyapunov Space of" # scale 1.5 # translate (r2(0.5, 14)) ↙
        ↳ # fc black # bold
    , alignedText 0 0.5 "Coupled FM Oscillators" # scale 1.5 # translate (r2(0.5, ↙
        ↳ 12.5)) # fc black # bold
    , alignedText 0 0.5 "Claude Heiland-Allen" # scale 1.5 # translate (r2(0.5, ↙
        ↳ 10)) # fc black
    , alignedText 0 0.5 "Linux Audio Conference 2013" # scale 1.5 # translate (r2↙
        ↳ (0.5, 7.5)) # fc black
    , box (0, 0) (24, 18)
245   ]
    frame (title:ts) = mconcat $
      [ alignedText 0 0.5 t # scale 0.5 # translate (r2(0.5, 17.25 - i))# fc black |↙
        ↳ (t, i) <- reverse ts `zip` [0..] ] ++
      [ alignedText 0 0.5 title # scale 1.5 # translate (r2(0.5, 14)) # fc black # ↙
        ↳ bold
    , alignedText 0 0.5 "Claude Heiland-Allen" # scale 0.5 # translate (r2(0.5, ↙
        ↳ 0.75)) # fc black
250   , alignedText 1 0.5 "Linux Audio Conference 2013" # scale 0.5 # translate (r2↙
        ↳ (23.5, 0.75)) # fc black
    , box (0, 0) (24, 18)
      ]

    label s p = bound (text s # scale 0.75) # translate (r2 p) # fc black
255
    llabel s p = bound (alignedText 0 0.5 s # scale 0.75) # translate (r2 p) # fc ↙
      ↳ black

    rlabel s p = bound (alignedText 1 0.5 s # scale 0.75) # translate (r2 p) # fc ↙
      ↳ black

260 bound d = d `atop` phantom' (square 0.5)

    phantom' d = phantom d `asTypeOf` d

    withEnvelope' d d' = withEnvelope (d `asTypeOf` d') d'
265
    line p q = fromVertices [p2 p, p2 q]

    triangle p q r = fromVertices [p2 p, p2 q, p2 r, p2 p] # fc black

270 box (x0, y0) (x1, y1) = mconcat
```

```
    [ line (x0, y0) (x1, y0)
    , line (x1, y0) (x1, y1)
    , line (x1, y1) (x0, y1)
    , line (x0, y1) (x0, y0)
    ]

xarrow (x0, y0) (x1, y1) = mconcat
    [ line (x0, y0) (x1 - 0.25, y1)
    , triangle (x1, y1) (x1 - 0.5, y1 + 0.25) (x1 - 0.5, y1 - 0.25)
    ]

yarrow (x0, y0) (x1, y1) = mconcat
    [ line (x0, y0) (x1, y1 - 0.25)
    , triangle (x1, y1) (x1 - 0.25, y1 - 0.5) (x1 + 0.25, y1 - 0.5)
    ]

nxarrow (x0, y0) (x1, y1) = mconcat
    [ line (x0, y0) (x1 + 0.25, y1)
    , triangle (x1, y1) (x1 + 0.5, y1 + 0.25) (x1 + 0.5, y1 - 0.25)
    ]

nyarrow (x0, y0) (x1, y1) = mconcat
    [ line (x0, y0) (x1, y1 + 0.25)
    , triangle (x1, y1) (x1 - 0.25, y1 + 0.5) (x1 + 0.25, y1 + 0.5)
    ]

d'fill = mconcat
    [ label "u" (9, 3.5)
    , label "v" (1.5, 8.5)
    , label "R" (17.5, 11.5)
    , label "G" (18.5, 11.5)
    , label "B" (19.5, 11.5)
    , label "A" (20.5, 11.5)
    , label "u" (17.5, 10.5), box (17, 10) (18, 11)
    , label "v" (18.5, 10.5), box (18, 10) (19, 11)
    , label "0" (19.5, 10.5), box (19, 10) (20, 11)
    , label "0" (20.5, 10.5), box (20, 10) (21, 11)
    , yarrow (2, 5) (2, 12)
    , xarrow (3, 4) (15, 4)
    , box (3, 5) (15, 12)
    , box (8, 7) (8.5, 7.5)
    , line (8, 7.5) (17, 11)
    , line (8.5, 7) (21, 10)
    ] # centerXY

d'copy = grid
    [ [ cell "11", cell "21", hdots, cell "w1" ]
    , [ cell "12", cell "22", hdots, cell "w2" ]
    , [ vdots', vdots', ddots, vdots' ]
    , [ cell "1h", cell "2h", hdots, cell "wh" ]
    ] # centerXY
  ==== strutY 1 ==== nyarrow (0, 2) (0, 0) ==== strutY 1 ====
  hcat [cell' "11", cell' "21", hdots, cell' "w1", cell' "12", hdots, cell' "w2
      ↪ ", cell' "13", hdots, cell' "wh" ] # centerXY
  where vdots' = centerXY vdots <> phantom' (centerXY (box (0,0) (4,2)))

```

```
hdots = mconcat [ dot (0.5, 0.5), dot (1, 0.5), dot (1.5, 0.5), phantom' (box ↙
    ↳ (0, 0) (2, 1)) ]

vdots = mconcat [ dot (0.5, 0.5), dot (0.5, 1), dot (0.5, 1.5), phantom' (box ↙
    ↳ (0, 0) (1, 2)) ]

ddots = mconcat [ dot (0.5, 1.5), dot (1, 1), dot (1.5, 0.5), phantom' (box (0, ↙
    ↳ 0) (2, 2)) ]

dot p = circle 0.1 # fc black # translate (r2 p)

cell [x,y] = hcat [sub ['u',x,y], sub ['v',x,y], sub "0", sub "0"]

cell' [x,y] = hcat [sub ['u',x,y], sub ['v',x,y]]

chunk _ [] = []
chunk n xs = let (ys, zs) = splitAt n xs in ys : chunk n zs

grid = vcat . map (centerXY . hcat . map centerXY)

d'init = ((vcat
  [ label "in" (6, 0.5)
  , strutY 7 # translate (r2(6,0))
  , vec2 "q" "u" "v"
  ] # centerXY) ||| strutX 1 ||| xarrow (0,0) (2, 0) ||| strutX 1 ||| (vcat
  [ label "out" (6, 0.5)
  , strutY 1 # translate (r2(6,0))
  , vec4' "p0" "0" "0" "0" "0"
  , vec4' "p1" "0" "0" "0" "0"
  , vdots # translate (r2(7.5, 0))
  , vec4' "pN" "0" "0" "0" "0"
  , vec4 "l" "0" "0" "0" "0"
  , vec2 "q" "u" "v"
  ] # centerXY)) # centerXY

d'unpack = ((vcat
  [ label "in" (6, 0.5)
  , strutY 2.5 # translate (r2(6,0))
  , vec4' "p0" "x0" "y0" "x1" "y1"
  , strutY 0.5 # translate (r2(6,0))
  , vdots # translate (r2(7.5, 0))
  , strutY 0.5 # translate (r2(6,0))
  , vec4' "pN" "xd-1" "yd-1" "xd" "yd"
  , strutY 0.5 # translate (r2(6,0))
  ] # centerXY) ||| strutX 1 ||| xarrow (0,0) (2, 0) ||| strutX 1 ||| (vcat
  [ label "out" (6, 0.5)
  , strutY 1 # translate (r2(6,0))
  , float "j" "0"
  , vec2A "p[D]" "x0" "y0"
  , vec2A'       "x1" "y1"
  , vdots # translate (r2(7.5, 0))
  , vec2A'       "xd-1" "yd-1"
  , vec2A'       "xd" "yd"
  ] # centerXY) # centerXY === strutY 2 === label "" (0, 0.5)) # centerXY

d'pack = ((vcat
  [ label "in" (6, 0.5)
```

```
    , strutY 1 # translate (r2(6,0))
    , float "j" "j"
    , vec2A "p[D]" "x0" "y0"
    , vec2A'       "x1" "y1"
    , vdots # translate (r2(7.5, 0))
    , vec2A'       "xd-1" "yd-1"
    , vec2A'       "xd" "yd"
    ] # centerXY) ||| strutX 1 ||| xarrow (0,0) (2, 0) ||| strutX 1 ||| (vcat
    [ label "out" (6, 0.5)
    , strutY 2.5 # translate (r2(6,0))
    , vec4' "p0" "xj" "yj" "xj+1" "yj+1"
    , strutY 0.5 # translate (r2(6,0))
    , vdots # translate (r2(7.5, 0))
    , strutY 0.5 # translate (r2(6,0))
    , vec4' "pN" "xj-2" "yj-2" "xj-1" "yj-1"
    , strutY 0.5 # translate (r2(6,0))
    ] # centerXY)) # centerXY === strutY 2 ===
    label "subscripts are wrapped into [0,D)" (0,0.5)

memrowlayout = memrowvec

memrowvec = hcat $
  map subV4 ["p0", "p1"] ++ [hdots, subV4 "pN", subV4 "1", subV2 "q"]

memrowodd = hcat $
  map sub ["x0", "y0", "x1", "y1", "x2", "y2", "x3", "y3"] ++
  [hdots] ++ map sub ["xd-1", "yd-1", "xd", "yd"] ++
  map ub ["1", "L", "LL", "", "u", "v"]

memroweven = hcat $
  map sub ["x0", "y0", "x1", "y1", "x2", "y2", "x3", "y3"] ++
  [hdots] ++ map sub ["xd", "yd", "  ", "  "] ++
  map ub ["1", "L", "LL", "", "u", "v"]

subV4 v = label' v (2, 0.5) <> box (0,0) (4,1)

subV2 v = label' v (1, 0.5) <> box (0,0) (2,1)

sub v = label' v (0.5, 0.5) <> box (0,0) (1,1)

ub v = label v (0.5, 0.5) <> box (0,0) (1,1)

float v x = mconcat
  [ label "float" (4, 0.5), phantom' (box (3, 0) (5, 1))
  , label v (6, 0.5)
  , label x (7.5, 0.5), box (7, 0) (8, 1)
  ]

vec2 v x y = mconcat
  [ label "vec2" (4, 0.5), phantom' (box (3, 0) (5, 1))
  , label v (5.5, 0.5)
  , label x (6.5, 0.5), box (6, 0) (7, 1)
  , label y (7.5, 0.5), box (7, 0) (8, 1)
  ]

vec2A v x y = mconcat
  [ label "vec2" (4, 0.5), phantom' (box (3, 0) (5, 1))
```

```
    , label v (6, 0.5)
    , label' x (7.5, 0.5), box (7, 0) (8, 1)
    , label' y (8.5, 0.5), box (8, 0) (9, 1)
440 ]

vec2A' x y = mconcat
  [ phantom' (box (3, 0) (5, 1))
  , label' x (7.5, 0.5), box (7, 0) (8, 1)
445 , label' y (8.5, 0.5), box (8, 0) (9, 1)
  ]

vec4 v x y z w = mconcat
  [ label "vec4" (4, 0.5), phantom' (box (3, 0) (5, 1))
450 , label v (5.5, 0.5)
  , label x (6.5, 0.5), box (6, 0) (7, 1)
  , label y (7.5, 0.5), box (7, 0) (8, 1)
  , label z (8.5, 0.5), box (8, 0) (9, 1)
  , label w (9.5, 0.5), box (9, 0) (10, 1)
455 ]

label' [s1] (x, y) = label [s1] (x, y)
label' (s1:s2) (x, y) = mconcat [label [s1] (x - 0.1875, y), label s2 (2 * x + ↙
    ↳ 0.5, 2 * y - 0.5) # scale 0.5]

460 label'' [s1] (x, y) = label [s1] (x, y)
label'' (s1:s2) (x, y) = mconcat [label [s1] (x, y), llabel s2 (2 * x + 0.5, 2 *↙
    ↳  y - 0.5) # scale 0.5]

vec4' v x y z w = mconcat
  [ label "vec4" (4, 0.5), phantom' (box (3, 0) (5, 1))
465 , label' v (5.5, 0.5)
  , label' x (6.5, 0.5), box (6, 0) (7, 1)
  , label' y (7.5, 0.5), box (7, 0) (8, 1)
  , label' z (8.5, 0.5), box (8, 0) (9, 1)
  , label' w (9.5, 0.5), box (9, 0) (10, 1)
470 ]

d'memory = vcat
  [ memrowvec # centerXY
  , strutY 1
475 , text "d odd" # scale 0.75 # fc black <> strutY 1
  , strutY 0.5
  , memrowodd # centerXY
  , strutY 1
  , text "d even" # scale 0.75 # fc black <> strutY 1
480 , strutY 0.5
  , memroweven # centerXY
  ]

d'pointers = vcat
485 [ label "stride" (12, 0.5) <> phantom' (box (0,0) (1,1))
  , xarrow (2, 0.5) (22, 0.5) <> (withEnvelope' (box (0,0) (1,1)) $ (line (2, ↙
    ↳ 0.5) (2, 0) <> line (22, 0.5) (22, 0)) # dashed)
  , memrowlayout # translate (r2(2, 0))
  , pointer "p0" 2 1
  , pointer "p1" 6 2
490 , vdots <> phantom' (box (0,0) (1,1))
```

```
    , pointer "pN" 12 5
    , pointer "l"  16 6
    , pointer "q"  20 7
    ] # centerXY

pointer l x y = withEnvelope' p (p <> q)
  where
    p = label' l (0.5, 0.5) <> xarrow (1, 0.5) (x, 0.5) <> phantom' (box (0,0) ↙
        ↳ (1,1))
    q = line (x, 0.5) (x, y) # dashed

d'step = vcat
  [ hcat [hdots, v "j-d-1", v "j-d", v "j-d+1", hdots, v "j-1", v "j", v "j+1", ↙
      ↳ v "j+2", hdots]
  , mconcat
      [ line (5, 2) (5, 1)
      , line (13, 2) (13, 1)
      , line (5, 1) (13, 1)
      , line (9, 1) (9, 0)
      , line (9, 0) (15, 0)
      , yarrow (15, 0) (15, 2)
      ]
  ] # centerXY === strutY 1 ===
  image "recurrence-equation.png" 16 2
  === strutY 0.5 ===
  image "inc-equation.png" 8 2

v' [a,b] s = label'' ([a]++s) (0.5, 1) <> label'' ([b]++s) (0.5, 0) <> phantom' ↙
    ↳ (box (0,-0.5) (2,1.5))

v s = v' "xy" s <> box (0,-0.5) (2,1.5)

d'pd = image "puredata.png" 9 9

d'opengl = mconcat
  [ label "vertex attributes" (-1, 1)
  , nyarrow (0, 0) (0, -2), llabel "vertex shader" (1, -1)
  , nyarrow (0, -3) (0, -5), llabel "geometry shader" (1, -4)
  , line (-0.5, -5.5) (-2, -5.5), yarrow (-2, -5.5) (-2, 0), rlabel "transform ↙
      ↳ feedback" (-3, -2)
  , nyarrow (0, -6) (0, -8), llabel "fragment shader" (1, -7)
  , label "fragment colour" (-1, -9)
  ]

d'compute = mconcat
  [ label "fill" (1, 6)
  , label "copy" (4, 6)
  , label "init" (7, 6)
  , label "colour" (4, 1)
  , label "plot" (7, 1)
  , label "step" (7, 3.5)
  , label "prune" (13, 3.5)
  , xarrow (0, 5) (2, 5)
  , xarrow (3, 5) (5, 5)
  , xarrow (6, 5) (8, 5)
  , nxarrow (8, 2) (6, 2)
  , nxarrow (5, 2) (3, 2)
```

```
    , nyarrow (8.5, 4.5) (8.5, 2.5)
    , line (9, 2) (11, 2)
    , line (11, 2) (11, 5)
    , nxarrow (11, 5) (9, 5)
    , phantom' (box (-1,-0.5) (14.5, 7))
    ] # centerXY

d'bifur =
  image "bifurcation-grey.png" 16 9 # centerXY
  === strutY 0.5 ===
  (label "2" (0, 0.5) <> label "r" (8, 0.5) <> label "4" (16, 0.5)) # centerXY

d'bifurc =
  image "bifurcation-colour.png" 16 9 # centerXY
  === strutY 0.5 ===
  (label "2" (0, 0.5) <> label "r" (8, 0.5) <> label "4" (16, 0.5)) # centerXY

d'bifurc2 =
  (((label "4" (0.5, 9) <> label "B" (0.5, 4.5) <> label "2" (0.5, 0)) # ↙
      ↳ centerXY)
   ||| strutX 0.5 ||| image "bifurcation-2d.png" 9 9 # centerXY) # centerXY
  === strutY 0.25 ===
  (phantom' (label " " (0.5, 0.5) # centerXY)
   ||| strutX 0.5 |||
   (label "2" (0, 0.5) <> label "A" (4.5, 0.5) <> label "4" (9, 0.5)) # centerXY
  ) # centerXY

dashed = dashing [0.125, 0.125] 0.1875
```

19 examples/cicada-2.sh

```
#!/bin/bash
set -e
s="cicada-2"
w=16
h=9
mkdir -p "${s}"
( cat <<-EOF
        lyapunov-fm/0
        geometry = 1280x720
        tiling   = 1x1+0+0
        delay    = 0.750000
        radius   = 0.262415
        note_x   = 93.291428
        note_y   = 106.493973
        index_x  = 5.687781
        index_y  = -38.408806
        matrix   = [ 1.000000, -1.000000, 0.000000, 0.000000 ; 0.000000, ↙
            ↳ 0.000000, 1.000000, -1.000000 ]
        render(${s}/small.pgm)
        geometry = 256x256
EOF
for y in $(seq $((h - 1)) -1 0)
do
  for x in $(seq 0 $((w - 1)))
  do
    cat <<-EOF
```

```
        tiling    = ${w}x${h}+${x}+${y}
        render(${s}/tile_${y}_${x}.pgm)
EOF
  done
done ) | ../src/lyapunov-fm 0.75 --batch
for y in $(seq 0 $((h - 1)))
do
  pnmcat -lr $(for x in $(seq 0 $((w - 1))) ; do echo "${s}/tile_${y}_${x}.pgm" ↙
      ↳ ; done) > "${s}/row_${y}.pgm"
done
pnmcat -tb $(for y in $(seq $((h - 1)) -1 0) ; do echo "${s}/row_${y}.pgm" ; ↙
    ↳ done) > "${s}/${s}.pgm"
pnmtopng -interlace -compression 9 -force "${s}/${s}.pgm" > "${s}/${s}.png"
convert "${s}/${s}.png" -normalize -geometry "1920x1080" -quality 90 "${s}/${s}.↙
    ↳ jpg"
```

20 examples/cicada-2-zoom.sh

```
#!/bin/bash
set -e
s="cicada--2-zoom"
w=32
h=18
mkdir -p "${s}"
( cat <<-EOF
        lyapunov-fm/0
        delay    = 0.750000
        note_x   = 93.291428
        note_y   = 106.493973
        index_x  = 5.687781
        index_y  = -38.408806
        matrix   = [ 1, -1, 0, 0 ; 0, 0, 1, -1 ]
        geometry = 240x240
EOF
for r in 0.25 0.5 1 2 4 8 16 32 64
do
  mkdir -p "${s}/${r}"
  echo "radius = ${r}"
  for y in $(seq $((h - 1)) -1 0)
  do
    for x in $(seq 0 $((w - 1)))
    do
      cat <<-EOF
        tiling = ${w}x${h}+${x}+${y}
        render(${s}/${r}/tile_${y}_${x}.pgm)
EOF
    done
  done
done ) | ../src/lyapunov-fm 0.75 --batch
for r in 0.25 0.5 1 2 4 8 16 32 64
do
  for y in $(seq 0 $((h - 1)))
  do
    pnmcat -lr $(for x in $(seq 0 $((w - 1))) ; do echo "${s}/${r}/tile_${y}_${x↙
        ↳ }.pgm" ; done) > "${s}/${r}/row_${y}.pgm"
  done
  pnmcat -tb $(for y in $(seq $((h - 1)) -1 0) ; do echo "${s}/${r}/row_${y}.pgm↙
```

```
      ↪ " ; done) > "${s}/${r}.pgm"
  pnmtopng -interlace -compression 9 -force "${s}/${r}.pgm" > "${s}/${r}.png"
  convert "${s}/${r}.png" -normalize -geometry "1920x1080" -quality 90 "${s}/${r↙
      ↪ }.jpg"
done
```

21 examples/cicada.sh

```
#!/bin/bash
set -e
s="cicada"
w=16
h=9
mkdir -p "${s}"
( cat <<-EOF
        lyapunov-fm/0
        geometry = 1280x720
        tiling   = 1x1+0+0
        delay    = 0.750000
        radius   = 0.262415
        note_x   = 93.291428
        note_y   = 106.493973
        index_x  = 5.687781
        index_y  = -38.408806
        matrix   = [ 1.000000, 1.000000, 0.000000, 0.000000 ; 0.000000, ↙
            ↪ 0.000000, 1.000000, 1.000000 ]
        render(${s}/small.pgm)
        geometry = 256x256
EOF
for y in $(seq $((h - 1)) -1 0)
do
  for x in $(seq 0 $((w - 1)))
  do
    cat <<-EOF
        tiling   = ${w}x${h}+${x}+${y}
        render(${s}/tile_${y}_${x}.pgm)
EOF
  done
done ) | ../src/lyapunov-fm 0.75 --batch
for y in $(seq 0 $((h - 1)))
do
  pnmcat -lr $(for x in $(seq 0 $((w - 1))) ; do echo "${s}/tile_${y}_${x}.pgm" ↙
      ↪ ; done) > "${s}/row_${y}.pgm"
done
pnmcat -tb $(for y in $(seq $((h - 1)) -1 0) ; do echo "${s}/row_${y}.pgm" ; ↙
    ↪ done) > "${s}/${s}.pgm"
pnmtopng -interlace -compression 9 -force "${s}/${s}.pgm" > "${s}/${s}.png"
convert "${s}/${s}.png" -normalize -geometry "1920x1080" -quality 90 "${s}/${s}.↙
    ↪ jpg"
```

22 examples/cicada-zoom.sh

```
#!/bin/bash
set -e
s="cicada-zoom"
w=32
```

```
h=18
mkdir -p "${s}"
( cat <<-EOF
        lyapunov-fm/0
        delay    = 0.750000
        note_x   = 93.291428
        note_y   = 106.493973
        index_x  = 5.687781
        index_y  = -38.408806
        matrix   = [ 1, 1, 0, 0 ; 0, 0, 1, 1 ]
        geometry = 240x240
EOF
for r in 0.25 0.5 1 2 4 8 16 32 64
do
  mkdir -p "${s}/${r}"
  echo "radius = ${r}"
  for y in $(seq $((h - 1)) -1 0)
  do
    for x in $(seq 0 $((w - 1)))
    do
      cat <<-EOF
        tiling = ${w}x${h}+${x}+${y}
        render(${s}/${r}/tile_${y}_${x}.pgm)
EOF
    done
  done
done ) | ../src/lyapunov-fm 0.75 --batch
for r in 0.25 0.5 1 2 4 8 16 32 64
do
  for y in $(seq 0 $((h - 1)))
  do
    pnmcat -lr $(for x in $(seq 0 $((w - 1))) ; do echo "${s}/${r}/tile_${y}_${x}.pgm" ; done) > "${s}/${r}/row_${y}.pgm"
  done
  pnmcat -tb $(for y in $(seq $((h - 1)) -1 0) ; do echo "${s}/${r}/row_${y}.pgm" ; done) > "${s}/${r}.pgm"
  pnmtopng -interlace -compression 9 -force "${s}/${r}.pgm" > "${s}/${r}.png"
  convert "${s}/${r}.png" -normalize -geometry "1920x1080" -quality 90 "${s}/${r}.jpg"
done
```

23 examples/crosswired.sh

```
#!/bin/bash
mkdir -p crosswired
w=16
h=12
( cat <<-EOF
        lyapunov-fm/0
        geometry = 1024x768
        tiling   = 1x1+0+0
        delay    = 0.75
        radius   = 0.5
        note_x   = 111.879204
        note_y   = 124.428627
        index_x  = 45.643826
        index_y  = 1.264403
```

```
        matrix   = [ 1, 1, 0, 0 ; 0, 0, 1, 1 ]
        render(crosswired/crosswired_preview.pgm)
        geometry = 600x600
EOF
for y in $(seq 0 $((h - 1)))
do
  for x in $(seq 0 $((w - 1)))
  do
    cat <<-EOF
        tiling   = ${w}x${h}+${x}+${y}
        render(crosswired/tile_${y}_${x}.pgm)
EOF
  done
done ) | ../src/lyapunov-fm 0.75 --batch
for y in $(seq 0 $((h - 1)))
do
  pnmcat -lr $(for x in $(seq 0 $((w - 1))) ; do echo "crosswired/tile_${y}_${x}.pgm" ; done) > "crosswired/row_${y}.pgm"
done
pnmcat -tb $(for y in $(seq $((h - 1)) -1 0) ; do echo "crosswired/row_${y}.pgm" ; done) > "crosswired/crosswired.pgm"
```

24 examples/delay.sh

```
#!/bin/bash
set -e
s="delay"
mkdir -p "${s}"
for d in $(seq 0 27)
do
  cat <<-EOF | ../src/lyapunov-fm ${d} --batch
        lyapunov-fm/0
        geometry = 1280x720
        tiling   = 1x1+0+0
        delay    = ${d}
        radius   = 48
        note_x   = 96
        note_y   = 96
        index_x  = 0
        index_y  = 0
        matrix   = [ 1, -1, 0, 0 ; 0, 0, 1, -1 ]
        render(${s}/${d}.pgm)
EOF
  pnmtopng -interlace -compression 9 -force "${s}/${d}.pgm" > "${s}/${d}.png"
done
```

25 fragmentarium/ab.frag

```
#include "Progressive2D.frag"

vec3 color(vec2 ab) {
        float rB = ab.x + 3.0;
        float rA = ab.y + 3.0;
        if (0.0 < rA && rA < 4.0 && 0.0 < rB && rB < 4.0) {
                float l = 0.0;
                float n = 0.0;
```

```
                float z = 0.5;
                float dz;
                for (int i = 0; i < 128; ++i) {
                        z = rA * z * (1.0 - z);
                        z = rB * z * (1.0 - z);
                }
                for (int i = 0; i < 1024; ++i) {
                        z = rA * z * (1.0 - z);
                        dz = abs(rA * (1.0 - 2.0 * z));
                        if (dz > 0.0) {
                                dz = log(dz);
                                if (-1000.0 < dz && dz < 1000.0) {
                                        l += dz;
                                        n += 1.0;
                                }
                        }
                        z = rB * z * (1.0 - z);
                        dz = abs(rB * (1.0 - 2.0 * z));
                        if (dz > 0.0) {
                                dz = log(dz);
                                if (-1000.0 < dz && dz < 1000.0) {
                                        l += dz;
                                        n += 1.0;
                                }
                        }
                }
                if (n > 0.0) {
                        l /= n;
                        if (l <= 0.01) {
                                return vec3(1.0 + tanh(l / 4.0), 1.0 + tanh(l / ↙
                                    ↳ 3.0), 0.0);
                        } else {
                                return vec3(0.0, 0.0, tanh(l * 3.0));
                        }
                } else {
                        return vec3(1.0, 1.0, 0.0);
                }
        } else {
                return vec3(0.0);
        }
}
```

26 .gitignore

```
-
src/*.glsl.c
src/lyapunov-fm
src/overlay
doc/lyapunov-fm.aux
doc/lyapunov-fm.bbl
doc/lyapunov-fm.blg
doc/lyapunov-fm.log
doc/lyapunov-fm.pdf
doc/videos
*.pgm
*.ppm
*.png
```

```
*.jpg
*.mkv
*.ogv
```

27 README

```
lyapunov-fm -- Lyapunov space of coupled FM oscillators
Copyright (C) 2013 Claude Heiland-Allen <claude@mathr.co.uk>
License: GPLv3+ http://www.gnu.org/licenses/gpl.html


Usage
-----

make -C src && ./src/lyapunov-fm

mouse left/scrollup: zoom in
mouse right/scrolldown: zoom out
mouse middle: recenter view
TAB: toggle viewing plane
F11: toggle full screen


Documentation
-------------

make -C doc && open doc/lyapunov-fm.pdf

Note: the figures for the paper are not included in this repository, see:
http://mathr.co.uk/lyapunov-fm/
```

28 src/audio.c

```
#include "audio.h"

static inline float mtoi(float m, float sr) {
  return 440.0f * powf(2.0f, (m - 69.0f) / 12.0f) / sr;
}

static inline float mix(float x, float y, float f) {
  return x * f + (1 - f) * y;
}
static inline float wmix(float x, float y, float f) {
  if (y < x) { return mix(x, y + 1, f); }
  else       { return mix(x, y, f); }
}

int audio_process(jack_nframes_t nframes, void *arg) {
  struct audio *s = (struct audio *) arg;
  jack_default_audio_sample_t *out[4];
  out[0] = (jack_default_audio_sample_t *) jack_port_get_buffer(s->port[0], ↙
      ↳ nframes);
  out[1] = (jack_default_audio_sample_t *) jack_port_get_buffer(s->port[1], ↙
      ↳ nframes);
  out[2] = (jack_default_audio_sample_t *) jack_port_get_buffer(s->port[2], ↙
      ↳ nframes);
```

```
  out[3] = (jack_default_audio_sample_t *) jack_port_get_buffer(s->port[3], ↙
      ↳ nframes);
  float *phase = s->state, df = s->df, sr = 48000.0f, m[2], p[4], Q[2];
  int j = s->j, d = s->d, dd = 2 * (s->d + 1);
  m[0] = s->mouse[0];
  m[1] = s->mouse[1];
  p[0] = s->param[0];
  p[1] = s->param[1];
  p[2] = s->param[2];
  p[3] = s->param[3];
  for (jack_nframes_t i = 0; i < nframes; ++i) {
#define P(w,t) phase[(2 * (t) + w + dd)%dd]
    Q[0] = fmodf(P(0,j) + mtoi(p[0] + p[2] * (out[0][i] = cosf(6.283185307179586↙
        ↳ f * wmix(P(1,j-d), P(1,j-d+1), df))), sr), 1.0f);
    Q[1] = fmodf(P(1,j) + mtoi(p[1] + p[3] * (out[1][i] = cosf(6.283185307179586↙
        ↳ f * wmix(P(0,j-d), P(0,j-d+1), df))), sr), 1.0f);
#define NOISE (0.00001f * (rand() / (float)RAND_MAX - 0.5f))
    P(0,j+1) = Q[0] + NOISE;
    P(1,j+1) = Q[1] + NOISE;
#undef NOISE
#undef P
    out[2][i] = m[0];
    out[3][i] = m[1];
    j = (j + 1) % (d + 1);
  }
  s->j = j;
  return 0;
}

void audio_do(struct audio *s, glm::vec2 &m, glm::vec4 &p) {
  s->mouse[0] = m[0];
  s->mouse[1] = m[1];
  s->param[0] = p[0];
  s->param[1] = p[1];
  s->param[2] = p[2];
  s->param[3] = p[3];
}

void audio_begin(struct audio *s, int d, float df) {
  s->mouse[0] = 0;
  s->mouse[1] = 0;
  s->param[0] = 48;
  s->param[1] = 48;
  s->param[2] = 0;
  s->param[3] = 0;
  s->d = d;
  s->df = df;
  s->j = 0;
  s->state = (float *) calloc(2 * (d + 1), sizeof(float));
  if ((s->client = jack_client_open("lyapunov-fm", JackNoStartServer, 0))) {
    jack_set_process_callback(s->client, audio_process, s);
    s->port[0] = jack_port_register(s->client, "output_1", ↙
        ↳ JACK_DEFAULT_AUDIO_TYPE, JackPortIsOutput, 0);
    s->port[1] = jack_port_register(s->client, "output_2", ↙
        ↳ JACK_DEFAULT_AUDIO_TYPE, JackPortIsOutput, 0);
    s->port[2] = jack_port_register(s->client, "output_3", ↙
        ↳ JACK_DEFAULT_AUDIO_TYPE, JackPortIsOutput, 0);
```

```
    s->port[3] = jack_port_register(s->client, "output_4", ↙
        ↳ JACK_DEFAULT_AUDIO_TYPE, JackPortIsOutput, 0);
    if (jack_activate(s->client)) {
      fprintf (stderr, "cannot activate JACK client\n");
    } else {
      const char **ports;
      if ((ports = jack_get_ports(s->client, NULL, NULL, JackPortIsPhysical | ↙
          ↳ JackPortIsInput))) {
        for (int i = 0; i < 2; ++ i) {
          if (! ports[i]) {
            break;
          }
          if (jack_connect(s->client, jack_port_name(s->port[i]), ports[i])) {
            fprintf(stderr, "cannot connect JACK output port\n");
          }
        }
        free(ports);
      }
    }
  }
}

void audio_end(struct audio *s) {
  if (s->client) {
    jack_client_close(s->client);
  }
}
```

29 src/audio.h

```
#ifndef AUDIO_H
#define AUDIO_H 1

#include <jack/jack.h>

struct audio {
  jack_client_t *client;
  jack_port_t *port[4];
  float mouse[2];
  float param[4];
  int d;
  float df;
  int j;
  float *state;
};

void audio_begin(struct audio *s, int d, float df);
void audio_end(struct audio *s);
void audio_do(struct audio *s, glm::vec2 &m, glm::vec4 &p);

#endif
```

30 src/batch.c

```
void batch_begin(struct batch *b) {
  b->version = -1;
```

```
  b->size = 0;
  view_init(&b->v, 1024, 576, 120, 120, 0, 0, 72, 0);
  b->size_specified = 0;
}

void batch_done(struct render *r, void *v) {
  (void) r;
  struct batch *b = (struct batch *) v;
  S.record.screenshot = 1;
  record_do(&S.record, &b->v);
  batch_do(b, &S.record);
}

void batch_do(struct batch *b, struct record *rec) {
  int len;
#define L if (-1 == getline(&b->line, &b->size, stdin)) { exit(0); } len = strlen(b->line); if (b->line[len - 1] == '\n') { b->line[len - 1] = 0; len--; } fprintf(stderr, "%s\n", b->line);
  char *rest = 0;
#define S0(str) if (0 == strncmp(str, b->line, strlen(str))) { rest = b->line + strlen(str);
#define S(str) } else S0(str)
#define F(fld) sscanf(rest, " = %f", &b->v.fld);
  L
  S0("lyapunov-fm") if (0 == strcmp("/0", rest)) { b->version = 0; L } else { fprintf(stderr, "bad version: %s\n", b->line); exit(1); }
  } while (1) {
    S0("geometry") sscanf(rest, " = %dx%d", &b->v.width, &b->v.height); b->size_specified = 1;
    S("tiling")    sscanf(rest, " = %dx%d+%d+%d", &b->v.tilesx, &b->v.tilesy, &b->v.tilex, &b->v.tiley);
    S("delay")     F(delay)
    S("radius")    F(radius)
    S("note_x")    F(center[0])
    S("note_y")    F(center[1])
    S("index_x")   F(center[2])
    S("index_y")   F(center[3])
    S("matrix")    sscanf(rest, " = [ %f, %f, %f, %f ; %f, %f, %f, %f ]"
      , &b->v.matrix[0][0], &b->v.matrix[0][1], &b->v.matrix[0][2], &b->v.matrix[0][3]
      , &b->v.matrix[1][0], &b->v.matrix[1][1], &b->v.matrix[1][2], &b->v.matrix[1][3]);
    S("render")    if (rest[0] == '(' && rest[strlen(rest) - 1] == ')') { rest[strlen(rest) - 1] = 0; if (rec->filename) { free(rec->filename); } rec->filename = strdup(rest + 1); break; }
    } else { fprintf(stderr, "warning: ignoring: %s\n", b->line);
    }
    L
  }
#undef S0
#undef S
#undef F
#undef L
  render_ondone(&S.render, batch_done, b);
  render_do(&S.render, &b->v);
}
```

```
void batch_reshape(struct batch *b, int w, int h) {
  view_reshape(&b->v, w, h);
}
```

31 src/batch.h

```
#ifndef BATCH_H
#define BATCH_H 1

struct batch {
  int version;
  char *line;
  size_t size;
  struct view v;
  int size_specified;
};

void batch_begin(struct batch *b);
void batch_do(struct batch *b, struct record *rec);

#endif
```

32 src/colour.c

```
#include "colour_vert.glsl.c"
#include "colour_frag.glsl.c"

struct colour {
  GLuint program;
  GLint mvp;
  GLint tex;
  GLint q0;
  GLuint vao;
  GLuint vbo;
};

void colour_begin(struct colour *s) {
  s->program = 0;
  s->mvp = -1;
  s->tex = -1;
  s->q0 = -1;
  s->vao = 0;
  s->vbo = 0;
  s->program = compile_program("colour", colour_vert, 0, colour_frag);
  s->mvp = glGetUniformLocation(s->program, "mvp");D;
  s->tex = glGetUniformLocation(s->program, "tex");D;
  s->q0 = glGetAttribLocation(s->program, "q0");D;
  glGenVertexArrays(1, &s->vao);D;
  glGenBuffers(1, &s->vbo);D;
}

void colour_end(struct colour *s) {
  glDeleteProgram(s->program);D;
  s->program = 0;
  glDeleteVertexArrays(1, &s->vao);D;
  s->vao = 0;
```

```
  glDeleteBuffers(1, &s->vbo);D;
  s->vbo = 0;
}

int cmp_float(const void *x, const void *y) {
  const float *a = (const float *) x;
  const float *b = (const float *) y;
  if (*a < *b) return -1;
  if (*a > *b) return  1;
  return 0;
}

void colour_do(struct colour *s, GLuint tex, int tex_size, struct view *v) {
  glBindTexture(GL_TEXTURE_2D, tex);D;
  glUseProgram(s->program);D;
  glm::mat4 mvp = glm::ortho(0.0f, 1.0f, 0.0f, 1.0f);
  glUniformMatrix4fv(s->mvp, 1, GL_FALSE, &mvp[0][0]);D;
  glUniform1i(s->tex, 0);D;
  GLfloat x = v->width  * 1.0 / tex_size;
  GLfloat y = v->height * 1.0 / tex_size;
  GLfloat quad[] = {
    0, 0,  0, 0,
    0, 1,  0, y,
    1, 0,  x, 0,
    1, 1,  x, y
  };
  glBindVertexArray(s->vao);D;
  glBindBuffer(GL_ARRAY_BUFFER, s->vbo);D;
  glBufferData(GL_ARRAY_BUFFER, 16 * sizeof(GLfloat), &quad, GL_STATIC_DRAW);D;
  glEnableVertexAttribArray(s->q0);D;
  glVertexAttribPointer(s->q0, 4, GL_FLOAT, GL_FALSE, 0, 0);D;
  glDrawArrays(GL_TRIANGLE_STRIP, 0, 4);D;
  glDisableVertexAttribArray(s->q0);D;
  glBindBuffer(GL_ARRAY_BUFFER, 0);D;
  glBindVertexArray(0);D;
  glUseProgram(0);D;
  glBindTexture(GL_TEXTURE_2D, 0);D;
}
```

33 src/colour_frag.glsl

```
uniform sampler2D tex;
smooth in vec2 t;
out layout(location = 0, index = 0) vec4 c;
void main() {
  vec3 g = texture(tex, t).xyz;
  float mean = 0.0;
  if (g.x > 0.0) { mean = g.y / g.x; }
  float umean = (g.y + 10.0) / (g.x + 1.0);
  float lmean = (g.y - 10.0) / (g.x + 1.0);
  float delta = abs(tanh(8.0 * umean) - tanh(8.0 * lmean));
  if (delta < 0.5 / 256.0) {
    c = vec4(vec3(0.5 + 0.5 * tanh(8.0 * mean)), 1.0);
  } else {
    c = vec4(vec3(0.5 + 0.5 * tanh(8.0 * mean)), 1.0) * vec4(1.0, 0.7, 0.7, 1.0)↙
        ↳ ;
  }
```

```
/*
  if (l <= -10.0) {
    c = vec4(0.0, 0.0, 0.0, 1.0);
  } else {
    c = 0.5 * mix(vec4(1.0 - tanh(l * 4.0), 1.0 - tanh(l * 3.0), 0.0, 2.0),
            vec4(0.0, 0.0, tanh(l * 3.0) + 1.0, 2.0), 0.5 + 0.5 * tanh(l));
  }
*/
}
```

34 src/colour_vert.glsl

```
uniform mat4 mvp;
in vec4 q0;
smooth out vec2 t;
void main() {
  t = q0.zw;
  gl_Position = mvp * vec4(q0.xy, 0.0, 1.0);
}
```

35 src/dvariable.c

```
void dvariable_begin(struct dvariable *s, int d) {
  s->d = d;
  s->packed = (d >> 1) + 1;
  s->nvaryings = s->packed + 2;
  s->attributes = (GLchar **) calloc(s->nvaryings, sizeof(GLchar *));
  s->varyings = (GLchar **) calloc(s->nvaryings, sizeof(GLchar *));
  for (int i = 0; i < s->packed; ++i) {
    s->attributes[i] = (GLchar *) calloc(1, 8);
    snprintf(s->attributes[i], 7, "p%d_i", i);
    s->varyings[i] = (GLchar *) calloc(1, 8);
    snprintf(s->varyings[i], 7, "p%d_o", i);
  }
  s->attributes[s->packed    ] = strdup("g_i");
  s->attributes[s->packed + 1] = strdup("q_i");
  s->varyings[s->packed    ] = strdup("g_o");
  s->varyings[s->packed + 1] = strdup("q_o");
  s->stride = (4 * (s->packed + 1) + 2) * sizeof(GLfloat);
  s->gptr = ((GLbyte *)0)+4*s->packed*sizeof(GLfloat);
  s->qptr = ((GLbyte *)0)+4*(s->packed+1)*sizeof(GLfloat);
}

void dvariable_setpointers(struct dvariable *s, GLuint program) {
  for (int i = 0; i < s->packed; ++i) {
    GLint p0 = glGetAttribLocation(program, s->attributes[i]);D;
    glEnableVertexAttribArray(p0);D;
    glVertexAttribPointer(p0, 4, GL_FLOAT, GL_FALSE, s->stride, ((GLbyte *)0)+ i↙
        ↳ *(4*sizeof(GLfloat)));D;
  }
  GLint g0 = glGetAttribLocation(program, "g_i");D;
  glEnableVertexAttribArray(g0);D;
  glVertexAttribPointer(g0, 4, GL_FLOAT, GL_FALSE, s->stride, s->gptr);D;
  GLint q0 = glGetAttribLocation(program, "q_i");D;
  glEnableVertexAttribArray(q0);D;
  glVertexAttribPointer(q0, 2, GL_FLOAT, GL_FALSE, s->stride, s->qptr);D;
```

```
}
```

36 src/dvariable.h

```
#ifndef DVARIABLE_H
#define DVARIABLE_H 1

struct dvariable {
  int d;
  int packed;
  int nvaryings;
  GLchar **attributes; // p%d_i, g_i, q_i
  GLchar **varyings;   // p%d_o, g_o, q_o
  int stride;
  void *gptr;
  void *qptr;
};

void dvariable_begin(struct dvariable *s, int d);
void dvariable_setpointers(struct dvariable *s, GLuint program);

#endif
```

37 src/fillc.c

```
#include "fillc_vert.glsl.c"
#include "fillc_frag.glsl.c"

struct fillc {
  GLuint program;
  GLint  mvp;
  GLint  p;
  GLuint vao;
  GLuint vbo;
};

void fillc_begin(struct fillc *s) {
  s->program = 0;
  s->mvp = -1;
  s->p = -1;
  s->vbo = 0;
  s->vao = 0;
  s->program = compile_program("fillc", fillc_vert, 0, fillc_frag);
  s->mvp = glGetUniformLocation(s->program, "mvp");D;
  s->p = glGetAttribLocation(s->program, "p");D;
  glGenVertexArrays(1, &s->vao);D;
  glGenBuffers(1, &s->vbo);D;
}

void fillc_end(struct fillc *s) {
  glDeleteProgram(s->program);D;
  s->program = 0;
  glDeleteVertexArrays(1, &s->vao);D;
  s->vao = 0;
  glDeleteBuffers(1, &s->vbo);D;
  s->vbo = 0;
```

```
}

void fillc_do(struct fillc *s, GLuint vbo1, GLuint fbo, GLuint tex, struct view ↙
    ↳ *v) {
  glBindFramebuffer(GL_FRAMEBUFFER, fbo);D;
  glFramebufferTexture(GL_FRAMEBUFFER, GL_COLOR_ATTACHMENT0, tex, 0);D;
  GLenum buffers[1] = { GL_COLOR_ATTACHMENT0 };
  glDrawBuffers(1, buffers);D;
  glViewport(0, 0, v->width, v->height);D;
  glUseProgram(s->program);D;
  float aspect = v->width * 1.0f / v->height;
  float tw = 2.0f * aspect / v->tilesy;
  float th = 2.0f          / v->tilesy;
  float tx = (v->tilex - v->tilesx / 2.0) * tw;
  float ty = (v->tiley - v->tilesy / 2.0) * th;
  glm::mat4 mvp = glm::ortho(tx, tx + tw, ty, ty + th);
  glUniformMatrix4fv(s->mvp, 1, GL_FALSE, &mvp[0][0]);D;
  GLfloat quad[] = {
    tx,      ty,
    tx,      ty + th,
    tx + tw, ty,
    tx + tw, ty + th
  };
  glBindVertexArray(s->vao);D;
  glBindBuffer(GL_ARRAY_BUFFER, s->vbo);D;
  glBufferData(GL_ARRAY_BUFFER, 8 * sizeof(GLfloat), &quad, GL_STATIC_DRAW);D;
  glVertexAttribPointer(s->p, 2, GL_FLOAT, GL_FALSE, 0, 0);D;
  glEnableVertexAttribArray(s->p);D;
  glDrawArrays(GL_TRIANGLE_STRIP, 0, 4);D;
  glDisableVertexAttribArray(s->p);D;
  glBindBuffer(GL_ARRAY_BUFFER, 0);D;
  glBindVertexArray(0);D;
  glUseProgram(0);D;
  // Copy texture to vertex buffer.
  glBindBuffer(GL_PIXEL_PACK_BUFFER, vbo1);D;
  glReadPixels(0, 0, v->width, v->height, GL_RG, GL_FLOAT, 0);D;
  glBindBuffer(GL_PIXEL_PACK_BUFFER, 0);D;
  glClearColor(0,0,0,0);D;
  glClear(GL_COLOR_BUFFER_BIT);D;
  glBindFramebuffer(GL_FRAMEBUFFER, 0);D;
}
```

38 src/fillc_frag.glsl

```
smooth in vec2 t;
out layout(location = 0, index = 0) vec4 o;
void main() {
  o = vec4(t, 0.0, 0.0);
}
```

39 src/fillc_vert.glsl

```
uniform mat4 mvp;
in  vec2 p;
smooth out vec2 t;
void main() {
```

```
  gl_Position = mvp * vec4(p, 0.0, 1.0);
  t = p;
}
```

40 src/init.c

```
struct init {
  GLuint program;
  GLint c;
  GLuint vao;
};

void init_begin(struct init *s, struct dvariable *d) {
  s->program = 0;
  s->c = -1;
  s->vao = 0;
  int len = 65536;
  char *src = (char *) calloc(1, len);
  int slen = 0;
#define L0(str) slen += snprintf(src + slen, len - slen - 1, str)
#define L1(fmt,arg) slen += snprintf(src + slen, len - slen - 1, fmt, arg)
  L0("in  vec2 c;\n");
  for (int i = 0; i < d->packed; ++i) {
    L1("flat out vec4 %s;\n", d->varyings[i]);
  }
  L0("flat out vec4 g_o;\n");
  L0("flat out vec2 q_o;\n");
  L0("void main() {\n");
  for (int i = 0; i < d->packed; ++i) {
    L1("  %s = vec4(0.0);\n", d->varyings[i]);
  }
  L0("  g_o = vec4(0.0);\n");
  L0("  q_o = c;\n");
  L0("}\n");
#undef L0
#undef L1
  s->program = compile_program("init", src, 0, 0, d->nvaryings, d->varyings);
  s->c = glGetAttribLocation(s->program, "c");D;
  glGenVertexArrays(1, &s->vao);D;
}

void init_end(struct init *s) {
  glDeleteProgram(s->program);D;
  s->program = 0;
  glDeleteVertexArrays(1, &s->vao);D;
  s->vao = 0;
}

void init_do(struct init *s, GLuint vbo0, GLuint vbo1, struct view *v) {
  glEnable(GL_RASTERIZER_DISCARD);D;
  glBindVertexArray(s->vao);D;
  glBindBuffer(GL_ARRAY_BUFFER, vbo0);D;
  glUseProgram(s->program);D;
  glVertexAttribPointer(s->c, 2, GL_FLOAT, GL_FALSE, 0, 0);D;
  glEnableVertexAttribArray(s->c);D;
  glBindBufferBase(GL_TRANSFORM_FEEDBACK_BUFFER, 0, vbo1);D;
  glBeginTransformFeedback(GL_POINTS);D;
```

```
  glDrawArrays(GL_POINTS, 0, v->width * v->height);D;
  glEndTransformFeedback();D;
  glBindBufferBase(GL_TRANSFORM_FEEDBACK_BUFFER, 0, 0);D;
  glUseProgram(0);D;
  glBindBuffer(GL_ARRAY_BUFFER, 0);D;
  glBindVertexArray(0);D;
  glDisable(GL_RASTERIZER_DISCARD);D;
}
```

41 src/interact.c

```
static struct {
  int which;
  struct view view;
} I;

void interact_mouse(int button, int state, int x, int y) {
  if (state == GLUT_DOWN) {
    glm::vec4 v;
    view_coord(&v, &I.view, x, y);
    if (button == GLUT_MIDDLE_BUTTON) { view_center(&I.view, v);
    } else if (button == GLUT_LEFT_BUTTON  || button == 3) { view_zoom(&I.view, ↙
        ↳ v, 1);
    } else if (button == GLUT_RIGHT_BUTTON || button == 4) { view_zoom(&I.view, ↙
        ↳ v, -1);
    }
    render_do(&S.render, &I.view);
  }
}

void interact_motion(int x, int y) {
  glm::vec4 v;
  view_coord(&v, &I.view, x, y);
  glm::vec2 m;
  m[0] = x / (float) I.view.width;
  m[1] = y / (float) I.view.height;
  audio_do(&S.audio, m, v);
}

void interact_automate(int x) {
  if (x < I.view.width) {
    interact_motion(x, I.view.height / 2);
    glutTimerFunc(40, interact_automate, x + 1);
  } else {
    interact_motion(0, I.view.height / 2);
    jack_transport_stop(S.audio.client);
  }
}

void interact_keyboard(unsigned char key, int x, int y) {
  (void) x;
  (void) y;
  if (key == 27 || key == 'q') {
    exit(0);
  }
  if (S.mode != 0) return;
  if (key == '\t') {
```

```
45      I.which = -I.which;
        view_plane(&I.view, I.which);
        render_do(&S.render, &I.view);
    /*
      } else if (key == 'a') {
50      S.record.screenshot = true;
        glutPostRedisplay();
        jack_transport_start(S.audio.client);
        glutTimerFunc(40, interact_automate, 0);
      } else if (key == 'l') {
55      S.animate.active = 1;
        S.render.view.which = 2;
        glutPostRedisplay();
    */
      } else if (key == 's') {
60      S.record.screenshot = true;
        glutPostRedisplay();
    /*
      } else if (key == 'r') {
        S.record.video = !S.record.video;
65      glutPostRedisplay();
      } else if (key == '@') {
        S.render.idle.f = 0;
        S.rendering = true;
    */
70    }
    }

    void interact_special(int key, int x, int y) {
      (void) x;
75    (void) y;
      if (key == GLUT_KEY_F11)                { glutFullScreenToggle(); return;
      } else if (key == GLUT_KEY_HOME)       { view_reset(&I.view, 96, 96, 0, 0, 48);↙
          ↪ view_plane(&I.view, I.which = 1);
      } else if (key == GLUT_KEY_PAGE_UP)    { view_zoom(&I.view,  0.1);
      } else if (key == GLUT_KEY_PAGE_DOWN) { view_zoom(&I.view, -0.1);
80    } else                                  { return; }
      render_do(&S.render, &I.view);
    }

    void interact_reshape(int w, int h) {
85    view_reshape(&I.view, w, h);
    }

    void interact_begin() {
      I.which = 1;
90    view_init(&I.view, 1024, 576, 96, 96, 0, 0, 48, 0);
      glutMouseFunc(interact_mouse);
      glutPassiveMotionFunc(interact_motion);
      glutKeyboardFunc(interact_keyboard);
      glutSpecialFunc(interact_special);
95  }
```

42 src/main.cc

```
#define _POSIX_C_SOURCE 199309L
```

```
#include <GL/glew.h>
#include <GL/freeglut.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <string.h>
#include <stdio.h>
#include <time.h>

#define D do{ int e = glGetError(); if (e != 0) { fprintf(stderr, "OpenGL error ↙
    ↳ %d in %s() (line %d)\n", e, __FUNCTION__, __LINE__); } }while(0)

#include "utility.c"
#include "shader.c"
#include "record.c"

#include "dvariable.h"
#include "dvariable.c"
#include "fillc.c"
#include "init.c"
#include "step.c"
#include "prune.c"
#include "plot.c"
#include "colour.c"

#include "state.c"
void idlecb();
#include "interact.c"
#include "render.c"
#include "view.c"
#include "audio.c"
#include "batch.c"

void reshapecb(int new_width, int new_height) {
  glViewport(0, 0, new_width, new_height);

  if (S.record.buffer) {
    free(S.record.buffer);
    S.record.buffer = 0;
  }

  render_reshape(&S.render, new_width, new_height);
  if (S.mode == 0) {
    interact_reshape(new_width, new_height);
    render_do(&S.render, &I.view);
  } else {
    batch_reshape(&S.batch, new_width, new_height);
    render_do(&S.render, &S.batch.v);
  }

  glutIdleFunc(idlecb);
}

void begincb(int d, float df) {
  render_begin(&S.render, d, df);
  if (S.mode == 0) {
    audio_begin(&S.audio, d, df);
    interact_begin();
```

```
  } else {
    batch_begin(&S.batch);
  }
}

void endcb() {
  render_end(&S.render);
  if (S.mode == 0) {
    audio_end(&S.audio);
  }
}

void displaycb() {
  render_display(&S.render);
}

void idlecb() {
  render_idle(&S.render);
}

int main(int argc, char **argv) {
  srand(time(NULL));
  float delay = 1;
  if (argc > 1) {
    delay = atof(argv[1]);
  }
  int d = ceilf(delay);
  float df = d - delay;
  int mode = 0;
  if (argc > 2) {
    mode = 1;
  }
  state_init(mode, delay);
  glutInitWindowSize(512, 288);
  glutInit(&argc, argv);
  glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE);
  glutCreateWindow(argv[0]);
  glewExperimental = GL_TRUE;
  glewInit();

  GLint n;
  glGetIntegerv(GL_MAX_VERTEX_ATTRIBS, &n);
  fprintf(stderr, "GL_MAX_VERTEX_ATTRIBS      = %d\n", n);
  glGetIntegerv(GL_MAX_VARYING_COMPONENTS, &n);
  fprintf(stderr, "GL_MAX_VARYING_COMPONENTS = %d\n", n);
  glGetIntegerv(GL_MAX_VARYING_VECTORS, &n);
  fprintf(stderr, "GL_MAX_VARYING_VECTORS     = %d\n", n);
  glGetIntegerv(GL_MAX_VARYING_FLOATS, &n);
  fprintf(stderr, "GL_MAX_VARYING_FLOATS      = %d\n", n);

  begincb(d, df);
  atexit(endcb);
  glutDisplayFunc(displaycb);
  glutReshapeFunc(reshapecb);

  if (mode != 0) {
    batch_do(&S.batch, &S.record);
```

```
  }
  glutMainLoop();
  return 0;
}
```

43 src/Makefile

```
SOURCES = \
        audio.c \
        audio.h \
        batch.c \
        batch.h \
        dvariable.h \
        dvariable.c \
        fillc.c \
        init.c \
        step.c \
        prune.c \
        plot.c \
        colour.c \
        interact.c \
        main.cc \
        record.c \
        render.c \
        render.h \
        shader.c \
        state.c \
        utility.c \
        view.c \
        view.h

SHADERS = \
        preamble.glsl.c \
        fillc_frag.glsl.c \
        fillc_vert.glsl.c \
        plot_vert.glsl.c \
        plot_frag.glsl.c \
        colour_vert.glsl.c \
        colour_frag.glsl.c

all: lyapunov-fm overlay

clean:
        -rm -f lyapunov-fm overlay $(SHADERS)

.SUFFIXES:
.PHONY: all clean

lyapunov-fm: $(SOURCES) $(SHADERS)
        g++ -ggdb -Wall -pedantic -Wextra -O3 -march=native -o lyapunov-fm main.↙
            ↳ cc -lGLEW -lglut -lGL -lrt -ljack

%.glsl.c: %.glsl s2c.sh
        bash s2c.sh $* < $< > $@

overlay: overlay.c
        gcc -std=c99 -ggdb -Wall -pedantic -Wextra -O3 -march=native -o overlay ↙
```

↳ overlay.c

44 src/overlay.c

```c
#include <stdio.h>
#include <stdlib.h>

static inline int min(int x, int y) { return x < y ? x : y; }
static inline int max(int x, int y) { return x > y ? x : y; }

int main(int argc, char **argv) {
  (void) argc;
  (void) argv;
  // read input header
  int w, h;
  if (2 != scanf("P5\n#%*[^\n]\n%d %d 255\n", &w, &h)) { return 1; }
  // allocate buffers
  unsigned char *pgm = calloc(1, w * h);
  if (! pgm) { return 1; }
  unsigned char *ppm = calloc(1, w * h * 3);
  if (! ppm) { return 1; }
  // read input image
  if (1 != fread(pgm, w * h, 1, stdin)) { return 1; }
  // copy image
  for (int i = 0, j = 0; i < w * h; ++i) {
    unsigned char g = pgm[i];
    ppm[j++] = g; ppm[j++] = g; ppm[j++] = g;
  }
  for (int f = 0; f < w; ++f) {
    // draw crosshair
    int x, y;
    y = h / 2;
    for (x = max(0, f - 8); x <= min(f + 8, w - 1); ++x) {
      if (x == f) { continue; }
      int j = 3 * (w * y + x);
      ppm[j++] = 255; ppm[j++] = 0; ppm[j++] = 0;
    }
    x = f;
    for (y = max(0, h/2 - 8); y <= min(h/2 + 8, h - 1); ++y) {
      if (y == h/2) { continue; }
      int j = 3 * (w * y + x);
      ppm[j++] = 255; ppm[j++] = 0; ppm[j++] = 0;
    }
    // write image
    fprintf(stdout, "P6\n%d %d\n255\n", w, h);
    fwrite(ppm, w * h * 3, 1, stdout);
    fflush(stdout);
    // erase crosshair
    y = h/2;
    for (x = max(0, f - 8); x <= min(f + 8, w - 1); ++x) {
      if (x == f) { continue; }
      int i = w * y + x;
      int j = 3 * i;
      unsigned char g = pgm[i];
      ppm[j++] = g; ppm[j++] = g; ppm[j++] = g;
    }
    x = f;
```

```
      for (y = max(0, h/2 - 8); y <= min(h/2 + 8, h - 1); ++y) {
        if (y == h/2) { continue; }
        int i = w * y + x;
        int j = 3 * i;
        unsigned char g = pgm[i];
        ppm[j++] = g; ppm[j++] = g; ppm[j++] = g;
      }
    }
    return 0;
}
```

45 src/plot.c

```
#include "plot_vert.glsl.c"
#include "plot_frag.glsl.c"

struct plot {
  GLuint program;
  GLint mvp;
  GLint g0;
  GLint q0;
  GLuint vao;
};

void plot_begin(struct plot *s, GLuint vbo, struct dvariable *d) {
  s->program = 0;
  s->mvp = -1;
  s->g0 = -1;
  s->q0 = -1;
  s->vao = 0;
  s->program = compile_program("plot", plot_vert, 0, plot_frag);
  s->mvp = glGetUniformLocation(s->program, "mvp");D;
  s->g0 = glGetAttribLocation(s->program, "g0");D;
  s->q0 = glGetAttribLocation(s->program, "q0");D;
  glGenVertexArrays(1, &s->vao);D;
  glBindVertexArray(s->vao);D;
  glBindBuffer(GL_ARRAY_BUFFER, vbo);D;
  glEnableVertexAttribArray(s->g0);D;
  glEnableVertexAttribArray(s->q0);D;
  glVertexAttribPointer(s->g0, 4, GL_FLOAT, GL_FALSE, d->stride, d->gptr);D;
  glVertexAttribPointer(s->q0, 2, GL_FLOAT, GL_FALSE, d->stride, d->qptr);D;
  glBindBuffer(GL_ARRAY_BUFFER, 0);D;
  glBindVertexArray(0);D;
}

void plot_end(struct plot *s) {
  glDeleteProgram(s->program);D;
  s->program = 0;
  glDeleteVertexArrays(1, &s->vao);D;
  s->vao = 0;
}

void plot_start(struct plot *s, GLuint fbo, GLuint tex, struct view *v) {
  glUseProgram(s->program);D;
  float aspect = v->width * 1.0f / v->height;
  float tw = 2.0f * aspect / v->tilesy;
  float th = 2.0f          / v->tilesy;
```

```
45    float tx = (v->tilex - v->tilesx / 2.0) * tw;
      float ty = (v->tiley - v->tilesy / 2.0) * th;
      glm::mat4 mvp = glm::ortho(tx, tx + tw, ty, ty + th);
      glUniformMatrix4fv(s->mvp, 1, GL_FALSE, &mvp[0][0]);D;
      glUseProgram(0);
50    glBindFramebuffer(GL_FRAMEBUFFER, fbo);D;
      glFramebufferTexture(GL_FRAMEBUFFER, GL_COLOR_ATTACHMENT0, tex, 0);D;
      GLenum buffers[1] = { GL_COLOR_ATTACHMENT0 };
      glDrawBuffers(1, buffers);D;
    }
55
    void plot_finish(struct plot *s) {
      (void) s;
      glBindFramebuffer(GL_FRAMEBUFFER, 0);D;
    }
60
    void plot_do(struct plot *s, int count) {
      glUseProgram(s->program);D;
      glBindVertexArray(s->vao);D;
      glDrawArrays(GL_POINTS, 0, count);D;
65    glBindVertexArray(0);D;
      glUseProgram(0);D;
    }
```

46 src/plot_frag.glsl

```
flat in vec3 g;
out layout(location = 0, index = 0) vec4 c;
void main() {
  c = vec4(g, 1.0);
5 }
```

47 src/plot_vert.glsl

```
uniform mat4 mvp;
in vec4 g0;
in vec2 q0;
flat out vec3 g;
5 void main() {
  g = g0.xyz;
  gl_Position = mvp * vec4(q0, 0.0, 1.0);
}
```

48 src/preamble.glsl

```
#version 330 core
precision highp float;
```

49 src/prune.c

```
struct prune {
  GLuint program;
  GLuint vao;
  GLuint query;
5 };
```

```
void prune_begin(struct prune *s, GLuint vbo, struct dvariable *d) {
  s->program = 0;
  s->vao = 0;
  s->query = 0;

  int len = 65536;
  char *src = (char *) calloc(1, len);
  int slen = 0;
#define L0(str) slen += snprintf(src + slen, len - slen - 1, str)
#define L1(fmt,arg) slen += snprintf(src + slen, len - slen - 1, fmt, arg)
#define L2(fmt,arg1,arg2) slen += snprintf(src + slen, len - slen - 1, fmt, arg1
    ↳ , arg2)

  for(int i = 0; i < d->packed; ++i) {
    L1("in vec4 p%d_i;\n", i);
  }
  L0("in vec4 g_i;\n");
  L0("in vec2 q_i;\n");
  for(int i = 0; i < d->packed; ++i) {
    L1("flat out vec4 p%d_v;\n", i);
  }
  L0("flat out vec4 g_v;\n");
  L0("flat out vec2 q_v;\n");
  L0("void main() {\n");
  for(int i = 0; i < d->packed; ++i) {
    L2("  p%d_v = p%d_i;\n", i, i);
  }
  L0("  g_v = g_i;\n");
  L0("  q_v = q_i;\n");
  L0("}\n");
  char *prune_vert = src;

  src = (char *) calloc(1, len);
  slen = 0;
  L0("layout(points) in;\n");
  L0("layout(points, max_vertices = 1) out;\n");
  for(int i = 0; i < d->packed; ++i) {
    L1("flat in vec4 p%d_v[1];\n", i);
  }
  L0("flat in vec4 g_v[1];\n");
  L0("flat in vec2 q_v[1];\n");
  for (int i = 0; i < d->packed; ++i) {
    L1("flat out vec4 p%d_o;\n", i);
  }
  L0("flat out vec4 g_o;\n");
  L0("flat out vec2 q_o;\n");
  L0("void main() {\n");
  L0("  float umean = (g_v[0].y + 10.0) / (g_v[0].x + 1.0);\n");
  L0("  float lmean = (g_v[0].y - 10.0) / (g_v[0].x + 1.0);\n");
  L0("  float delta = abs(tanh(8.0 * umean) - tanh(8.0 * lmean));\n");
  L0("  if (! (g_v[0].x >= 64.0 && delta < 0.5 / 256.0)) {\n");
  for (int i = 0; i < d->packed; ++i) {
    L2("  p%d_o = p%d_v[0];\n", i, i);
  }
  L0("    g_o = g_v[0];\n");
  L0("    q_o = q_v[0];\n");
```

```
    L0("    EmitVertex();\n");
    L0("    EndPrimitive();\n");
    L0("  }\n");
    L0("}\n");
    char *prune_geom = src;

#undef L0
#undef L1
#undef L2

    s->program = compile_program("prune", prune_vert, prune_geom, 0, d->nvaryings,
        d->varyings);
    glGenVertexArrays(1, &s->vao);D;
    glGenQueries(1, &s->query);D;
    glBindVertexArray(s->vao);D;
    glBindBuffer(GL_ARRAY_BUFFER, vbo);D;
    dvariable_setpointers(d, s->program);
    glBindBuffer(GL_ARRAY_BUFFER, 0);D;
    glBindVertexArray(0);D;
}

void prune_end(struct prune *s) {
    glDeleteProgram(s->program);D;
    s->program = 0;
    glDeleteVertexArrays(1, &s->vao);D;
    s->vao = 0;
    glDeleteQueries(1, &s->query);D;
    s->query = 0;
}

void prune_do(struct prune *s, GLuint vbo, GLuint *count) {
    glEnable(GL_RASTERIZER_DISCARD);
    glBindVertexArray(s->vao);D;
    glUseProgram(s->program);D;
    glBindBufferBase(GL_TRANSFORM_FEEDBACK_BUFFER, 0, vbo);D;
    glBeginQuery(GL_TRANSFORM_FEEDBACK_PRIMITIVES_WRITTEN, s->query);D;
    glBeginTransformFeedback(GL_POINTS);D;
    glDrawArrays(GL_POINTS, 0, *count);D;
    glEndTransformFeedback();D;
    glEndQuery(GL_TRANSFORM_FEEDBACK_PRIMITIVES_WRITTEN);D;
    glGetQueryObjectuiv(s->query, GL_QUERY_RESULT, count);D;
    glBindBufferBase(GL_TRANSFORM_FEEDBACK_BUFFER, 0, 0);D;
    glUseProgram(0);D;
    glBindVertexArray(0);D;
    glDisable(GL_RASTERIZER_DISCARD);
}
```

50 src/record.c

```
#include "view.h"

struct record {
    unsigned char *buffer;
    int bytes;
    int seqno;
    bool screenshot;
    bool video;
```

```
  char *filename;
};

void record_begin(struct record *s) {
  s->buffer = 0;
  s->bytes = 0;
  s->seqno = 0;
  s->screenshot = false;
  s->video = false;
  s->filename = 0;
}

void record_end(struct record *s) {
  if (s->buffer) {
    free(s->buffer);
    s->buffer = 0;
    s->bytes = 0;
  }
}

void record_write(FILE *out, unsigned char *buffer, struct view *v) {
  fprintf(out, "P5\n");
  fprintf(out, "# lyapunov-fm/0\n");
  fprintf(out, "# geometry = %dx%d\n", v->width, v->height);
  fprintf(out, "# tiling   = %dx%d+%d+%d\n", v->tilesx, v->tilesy, v->tilex, v->tiley);
  fprintf(out, "# delay    = %f\n", v->delay);
  fprintf(out, "# radius   = %f\n", v->radius);
  fprintf(out, "# note_x   = %f\n", v->center[0]);
  fprintf(out, "# note_y   = %f\n", v->center[1]);
  fprintf(out, "# index_x  = %f\n", v->center[2]);
  fprintf(out, "# index_y  = %f\n", v->center[3]);
  fprintf(out, "# matrix   = [ %f, %f, %f, %f ; %f, %f, %f, %f ]\n"
    , v->matrix[0][0], v->matrix[0][1], v->matrix[0][2], v->matrix[0][3]
    , v->matrix[1][0], v->matrix[1][1], v->matrix[1][2], v->matrix[1][3]
    );
  fprintf(out, "%d %d\n255\n", v->width, v->height);
  fflush(out);
  for (int y = v->height - 1; y >= 0; --y) {
    fwrite(buffer + y * v->width, v->width, 1, out);
  }
  fflush(out);
}

void record_do(struct record *s, struct view *v) {
  if (s->screenshot || s->video) {
    if (! s->buffer) {
      s->bytes = v->width * v->height;
      s->buffer = (unsigned char *) malloc(s->bytes);
    }
    glReadPixels(0, 0, v->width, v->height, GL_RED, GL_UNSIGNED_BYTE, s->buffer);
    if (s->screenshot) {
      char filename[1024];
      snprintf(filename, 1000, "%04d.pgm", s->seqno++);
      FILE *out = fopen(s->filename ? s->filename : filename, "wb");
      if (! out) {
```

```
        fprintf(stderr, "\aERROR: screenshot failed\n");
      } else {
        record_write(out, s->buffer, v);
        fclose(out);
      }
      s->screenshot = false;
    }
    if (s->video) {
      record_write(stdout, s->buffer, v);
    }
  }
}
```

51 src/render.c

```
void render_begin(struct render *s, int d, float df) {
  s->reshape_requested = 0;
  s->win_width = -1;
  s->win_height = -1;
  s->tex_size = -1;
  s->vbo_size = -1;
  s->tex = 0;
  s->fbo = 0;
  s->vbo[0] = 0;
  s->vbo[1] = 0;
  s->count = 0;
  s->steps = 0;
  s->samplerate = 48000.0;

  glClampColor(GL_CLAMP_VERTEX_COLOR, GL_FALSE);D;
  glClampColor(GL_CLAMP_READ_COLOR, GL_FALSE);D;
  glClampColor(GL_CLAMP_FRAGMENT_COLOR, GL_FALSE);D;
  glGenTextures(1, &s->tex);D;
  glBindTexture(GL_TEXTURE_2D, s->tex);D;
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);D;
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);D;
  glBindTexture(GL_TEXTURE_2D, 0);D;
  glGenFramebuffers(1, &s->fbo);D;
  glGenBuffers(2, s->vbo);D;

  dvariable_begin(&s->dvariable, d);
  fillc_begin(&s->fillc);
  init_begin(&s->init, &s->dvariable);
  step_begin(&s->step, s->vbo[0], &s->dvariable, df);
  prune_begin(&s->prune, s->vbo[1], &s->dvariable);
  plot_begin(&s->plot, s->vbo[1], &s->dvariable);
  colour_begin(&s->colour);

  s->idle.f = 0;
  s->done.f = 0;
  s->done.u = 0;
}

void render_end(struct render *s) {
  glDeleteTextures(1, &s->tex);D;
  glDeleteFramebuffers(1, &s->fbo);D;
  glDeleteBuffers(2, s->vbo);D;
```

```
  fillc_end(&s->fillc);
  init_end(&s->init);
  step_end(&s->step);
  prune_end(&s->prune);
  plot_end(&s->plot);
  colour_end(&s->colour);
}

void render_reshape(struct render *s, int w, int h) {
  s->win_width  = w;
  s->win_height = h;
  if (s->reshape_requested) {
    s->reshape_requested = 0;
    s->view.width  = w;
    s->view.height = h;
  }
}

void render_go(struct render *s);
void render_do(struct render *s, struct view *v) {
  if (v->width != s->win_width || v->height != s->win_height) {
    if (! s->reshape_requested)  {
      s->reshape_requested = 1;
      glutReshapeWindow(v->width, v->height);
    }
    return;
  }
  int new_count = v->width * v->height;
  int new_size = ceil2n(max(v->width, v->height));
  bool reallocate = new_size != s->tex_size;
  if (reallocate) {
    s->tex_size = new_size;
    glBindTexture(GL_TEXTURE_2D, s->tex);D;
    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA32F, s->tex_size, s->tex_size, 0, ↙
        ↳ GL_RGBA, GL_UNSIGNED_BYTE, 0);D;
    glBindTexture(GL_TEXTURE_2D, 0);D;
  }
  if (s->vbo_size != new_count) {
    s->vbo_size = new_count;
    for (int i = 0; i < 2; ++i) {
      glBindBuffer(GL_ARRAY_BUFFER, s->vbo[i]);D;
      glBufferData(GL_ARRAY_BUFFER, s->vbo_size * s->dvariable.stride, 0, ↙
          ↳ GL_DYNAMIC_COPY);D;
      glBindBuffer(GL_ARRAY_BUFFER, 0);D;
    }
  }
  view_copy(&s->view, v);
  s->count = s->view.width * s->view.height;
  s->steps = 0;
  s->total = 0;
  clock_gettime(CLOCK_MONOTONIC, &s->t_start);
  clock_gettime(CLOCK_MONOTONIC, &s->t_update);
  fillc_do(&s->fillc, s->vbo[1], s->fbo, s->tex, &s->view);
  init_do(&s->init, s->vbo[1], s->vbo[0], &s->view);
  step_start(&s->step, s->samplerate, &s->view);
  s->idle.f = render_go;
  glutIdleFunc(idlecb);
```

```
}

void render_go(struct render *s) {
  plot_start(&s->plot, s->fbo, s->tex, &s->view);

#define GO \
  s->total += s->count; \
  step_do(&s->step, s->vbo[1], s->count, s->steps++ > 128); \
  plot_do(&s->plot, s->count); \
  prune_do(&s->prune, s->vbo[0], &s->count);

  glFinish();
  struct timespec t0;
  clock_gettime(CLOCK_MONOTONIC, &t0);

  GO

  glFinish();
  struct timespec t1;
  clock_gettime(CLOCK_MONOTONIC, &t1);

  double dt = time_difference(&t1, &t0);

  for (double t = dt; t < 0.04 && s->count /* && s->total < 8192 * s->view.width
      ↪  * s->view.height */ ; t += dt) {
    GO
  }

#undef GO

  plot_finish(&s->plot);
  if (! (s->count /* && s->total < 8192 * s->view.width * s->view.height */)) {
    s->idle.f = 0;
  }
}

void render_idle(struct render *s) {
  if (s->idle.f) {
    s->idle.f(s);
  } else {
    glutIdleFunc(0);
  }
  glutPostRedisplay();
  glutReportErrors();
}

void render_display(struct render *s) {
  struct timespec t;
  clock_gettime(CLOCK_MONOTONIC, &t);
  if (time_difference(&t, &s->t_update) > 1 || ! s->count) {
    fprintf(stderr, "\r%8d%12d    %4.3f", s->count, s->total, time_difference(&t
        ↪ , &s->t_start)); fflush(stderr);
    clock_gettime(CLOCK_MONOTONIC, &s->t_update);
    if (! s->count) {
      fprintf(stderr, "\n");
    }
  }
```

```
  colour_do(&s->colour, s->tex, s->tex_size, &s->view);
  glutSwapBuffers();
  if (! s->idle.f) {
    if (s->done.f) {
      s->done.f(s, s->done.u);
    }
  }
}

void render_ondone(struct render *s, void (*f)(struct render *, void *), void *u↙
    ↳ ) {
  s->done.f = f;
  s->done.u = u;
}
```

52 src/render.h

```
#ifndef RENDER_H
#define RENDER_H 1

#include "view.h"

struct idlefunc_t { void (*f)(struct render *); };
struct donefunc_t { void (*f)(struct render *, void *); void *u; };

struct render {
  struct dvariable dvariable;
  struct prune prune;
  struct idlefunc_t idle;
  struct donefunc_t done;
  double samplerate;
  int reshape_requested;
  int win_width;
  int win_height;
  int tex_size;
  int vbo_size;
  GLuint tex;
  GLuint fbo;
  GLuint vbo[2];
  GLuint count;
  int steps;
  int total;
  struct view view;
  struct fillc fillc;
  struct init init;
  struct step step;
  struct plot plot;
  struct colour colour;
  struct timespec t_start;
  struct timespec t_update;
};

void render_begin(struct render *s, int d, float df);
void render_end(struct render *s);
void render_reshape(struct render *s, int w, int h);
void render_do(struct render *s, struct view *v);
void render_idle(struct render *s);
```

```
void render_display(struct render *s);
void render_ondone(struct render s, void (*f)(struct render *, void *), void *u)↙
    ↳ ;

#endif
```

53 src/s2c.sh

```
#!/bin/bash
echo "static const char $1[] ="
sed 's|\\|\\\\|g' |
sed 's|"|\\"|g' |
sed 's|^|"|' |
sed 's|$|\\n"|'
echo ";"
```

54 src/shader.c

```
#include "preamble.glsl.c"

void debug_program(GLuint program, const char *name, const GLchar *vert, const ↙
    ↳ GLchar *geom, const GLchar *frag) {
  if (program) {
    GLint linked = GL_FALSE;
    glGetProgramiv(program, GL_LINK_STATUS, &linked);D;
    if (linked != GL_TRUE) {
      fprintf(stderr, "%s: link failed\n", name);
    }
    int length;
    glGetProgramiv(program, GL_INFO_LOG_LENGTH, &length);D;
    char *buffer = (char *) malloc(length + 1);
    glGetProgramInfoLog(program, length, NULL, buffer);D;
    buffer[length] = 0;
    if (buffer[0]) {
      fprintf(stderr, "==== %s.log ====\n%s\n", name, buffer);
      if (vert) { fprintf(stderr, "---- %s.vert ----\n%s\n", name, vert); }
      if (geom) { fprintf(stderr, "---- %s.geom ----\n%s\n", name, geom); }
      if (frag) { fprintf(stderr, "---- %s.frag ----\n%s\n", name, frag); }
    }
    free(buffer);
  } else {
    fprintf(stderr, "%s: program failed\n", name);
  }
}

void compile_shader(GLint program, GLenum type, const GLchar *source) {
  const GLchar *sources[2] = { preamble, source };
  GLuint shader = glCreateShader(type);D;
  glShaderSource(shader, 2, sources, 0);D;
  glCompileShader(shader);D;
  glAttachShader(program, shader);D;
  glDeleteShader(shader);D;
}

GLint compile_program(const char *name, const GLchar *vert, const GLchar *geom, ↙
    ↳ const GLchar *frag, int nvaryings, GLchar **varyings) {
```

```
  GLint program = glCreateProgram();D;
  if (vert) { compile_shader(program, GL_VERTEX_SHADER   , vert); }
  if (geom) { compile_shader(program, GL_GEOMETRY_SHADER, geom); }
  if (frag) { compile_shader(program, GL_FRAGMENT_SHADER, frag); }
  if (nvaryings) {
    glTransformFeedbackVaryings(program, nvaryings, (const GLchar **) varyings, ↙
        ↳ GL_INTERLEAVED_ATTRIBS);D;
  }
  glLinkProgram(program);D;
  debug_program(program, name, vert, geom, frag);
  return program;
}

GLint compile_program(const char *name, const GLchar *vert, const GLchar *geom, ↙
    ↳ const GLchar *frag) {
  return compile_program(name, vert, geom, frag, 0, 0);
}
```

55 src/state.c

```
#include "render.h"
#include "view.h"
#include "audio.h"
#include "batch.h"

static struct {
  int mode;
  bool rendering;
  struct render render;
  struct record record;
  struct audio audio;
  struct batch batch;
} S;

void state_init(int mode, double delay) {
  S.mode = mode;
  batch_begin(&S.batch);
  view_init(&S.render.view, 1280, 720, 120, 120, 0, 0, 72, delay);
//  view_init(&S.render.view, 1800, 1200, 120, 120, 0, 0, 72, 0);
//  1400, 1000, 123.609177, 72.565994, 17.597776, -21.296762, 3, 1);

// 183.863355798875289793, 183.863355798875289793, 1.306958840076017037, ↙
    ↳ 1.306958840076017037, 0.05);
//  125.525291442871093750, 125.525291442871093750, -21.048765182495117188, ↙
    ↳ 2.741522789001464844, 1.125000000000000000);
//148.668530260743807503, 148.668530260743807503, 1.812859159382673724, ↙
    ↳ 1.812859159382673724, 0.198873782208716543/4);
//118.093467857212488070, 113.307444175989090240, 11.232046341461641603, ↙
    ↳ -32.543675944390649590, 4.242640687119273224);
  S.rendering = false;
}
```

56 src/step.c

```
struct step {
  GLuint program;
```

```
  GLint sr;
  GLint iters;
  GLint perturb;
  GLint dither;
  GLint model;
  GLint center;
  GLint radius;
  GLint keep;
  GLuint vao;
  float dithersize;
  int d;
  GLfloat *noise;
};

void step_begin(struct step *s, GLuint vbo, struct dvariable *d, float df) {
  s->program = 0;
  s->sr = -1;
  s->iters = -1;
  s->perturb = -1;
  s->dither = -1;
  s->model = -1;
  s->center = -1;
  s->radius = -1;
  s->keep = -1;
  s->vao = 0;
  s->dithersize = 0;
  s->d = d->d;

  int len = 65536;
  char *src = (char *) calloc(1, len);
  int slen = 0;
#define L0(str) slen += snprintf(src + slen, len - slen - 1, str)
#define L1(fmt,arg) slen += snprintf(src + slen, len - slen - 1, fmt, arg)
#define L2(fmt,arg1,arg2) slen += snprintf(src + slen, len - slen - 1, fmt, arg1↙
    ↳ , arg2)

L1("const int D = %d;\n", d->d + 1);
L1("const float df = %f;\n", df);
L0("uniform float sr;\n");
L0("uniform int iters;\n");
L1("uniform vec2 perturb[%d];\n", d->d + 1);
L0("uniform vec2 dither;\n");
L0("uniform bool keep;\n");
L0("uniform mat4 model;\n");
L0("uniform vec4 center;\n");
L0("uniform float radius;\n");
for (int i = 0; i < d->packed; ++i) {
  L1("in vec4 p%d_i;\n", i);
}
L0("in vec4 g_i;\n");
L0("in vec2 q_i;\n");
for (int i = 0; i < d->packed; ++i) {
  L1("flat out vec4 p%d_o;\n", i);
}
L0("flat out vec4 g_o;\n");
L0("flat out vec2 q_o;\n");
L0("float delta(vec2 l, vec2 r) {\n");
```

```
  L0("  vec2 d = l - r;\n");
  L0("  d = abs(d);\n");
  L0("  d = min(d, abs(d - vec2(1.0)));\n");
  L0("  return dot(d, d);\n");
  L0("}\n");
  L0("vec2 mtoi(vec2 m) {\n");
  L0("  return 440.0 * pow(vec2(2.0), (m - vec2(69.0)) / 12.0) / sr;\n");
  L0("}\n");
  L0("float wmix(float x, float y, float f) {\n");
  L0("  if (y < x) { return mix(x, y + 1.0, f); }\n");
  L0("  else       { return mix(x, y, f); }\n");
  L0("}\n");
  L0("vec2 wmix(vec2 x, vec2 y, float f) {\n");
  L0("  return vec2(wmix(x.x, y.x, f), wmix(x.y, y.y, f));\n");
  L0("}\n");
  L0("\n");
  L0("vec2 step(vec2 z, vec2 zz, vec4 a1) {\n");
  L0("  return fract(z.xy + mtoi(a1.xy + a1.zw * cos(6.283185307179586 * zz.yx)))
      ↪ ;\n");
  L0("}\n");
  L0("void main() {\n");
  L0("// state vector\n");
  L0("  vec2 l[D];\n");
  L0("  vec2 r[D];\n");
  L0("\n");
  L0("  // unpack state vector\n");
  for (int i = 0; i < d->packed; ++i) {
    L2("  r[%d] = p%d_i.xy;\n", 2 * i, i);
    if (2 * i < d->d) {
      L2("  r[%d] = p%d_i.zw;\n", 2 * i + 1, i);
    }
  }
  L0("\n");
  L0("  // perturb\n");
  L0("  for (int i = 0; i < D; ++i) {\n");
  L0("    l[i] = r[i] + perturb[i];\n");
  L0("  }\n");
  L0("\n");
  L0("  // maximum separation\n");
  L0("  float sM = float(D) * 0.25;\n");
  L0("  // initial separation\n");
  L0("  float s0 = 0.0;\n");
  L0("  for (int i = 0; i < D; ++i) {\n");
  L0("    s0 += delta(l[i], r[i]);\n");
  L0("  }\n");
  L0("\n");
  L0("  // parameter\n");
  L0("  vec4 a1 = model * (vec4(radius * (q_i + dither), 0.0, 0.0)) + center;\n");
  L0("\n");
  L0("  // step\n");
  L0("  vec3 f = vec3(g_i.xyz);\n");
  L0("  int j = 0;\n");
  L0("  for (int h = 0; h < iters/D + 1; ++h) {\n");
  L0("    if (s0 > sM * 0.25) { break; }\n");
  L0("    for (int i = 0; i < D; ++i) {\n");
  L0("      int jj = (j + 1) %% D;\n");
  L0("      int jjj = (j + 2) %% D;\n");
```

```
115   L0("      l[jj] = step(l[j], wmix(l[jj], l[jjj], df), a1);\n");
      L0("      r[jj] = step(r[j], wmix(r[jj], r[jjj], df), a1);\n");
      L0("      j = jj;\n");
      L0("    }\n");
      L0("\n");
120   L0("    // final separation\n");
      L0("    float s1 = 0.0;\n");
      L0("    for (int i = 0; i < D; ++i) {\n");
      L0("      s1 += delta(l[i], r[i]);\n");
      L0("    }\n");
125   L0("    // accumulate lyapunov exponent\n");
      L0("    if (keep && s0 > 0.0 && s1 > 0.0) {\n");
      L0("      float k = 0.5 * log(s1 / s0) / float(D);\n");
      L0("      if (! ( 10.0 >= k)) { k =  10.0; }\n");
      L0("      if (! (-10.0 <= k)) { k = -10.0; }\n");
130   L0("      f += vec3(1.0, k, k * k);\n");
      L0("    }\n");
      L0("    s0 = s1;\n");
      L0("  }\n");
      L0("  g_o = vec4(f, g_i.w);\n");
135   L0("\n");
      L0("  // pass through pixel coords\n");
      L0("  q_o = q_i;\n");
      L0("\n");
      L0("  // pack state vector\n");
140   L0("  // explore more orbits by picking the perturbed version\n");
      for (int i = 0; i < d->packed; ++i) {
        L2("  p%d_o.xy = l[(j + %d) %% D];\n", i, 2 * i);
        if (2 * i < d->d) {
          L2("  p%d_o.zw = l[(j + %d) %% D];\n", i, 2 * i + 1);
145     }
      }
      L0("\n");
      L0("}\n");

150   #undef L0
      #undef L1
      #undef L2

        GLchar *step_vert = src;
155
        s->program = compile_program("step", step_vert, 0, 0, d->nvaryings, d->↙
            ↳ varyings);
        s->sr = glGetUniformLocation(s->program, "sr");D;
        s->iters = glGetUniformLocation(s->program, "iters");D;
        s->perturb = glGetUniformLocation(s->program, "perturb");D;
160     s->noise = (GLfloat *) calloc(1, 2 * (s->d + 1) * sizeof(GLfloat));
        s->dither = glGetUniformLocation(s->program, "dither");D;
        s->model = glGetUniformLocation(s->program, "model");D;
        s->center = glGetUniformLocation(s->program, "center");D;
        s->radius = glGetUniformLocation(s->program, "radius");D;
165     s->keep = glGetUniformLocation(s->program, "keep");D;
        glGenVertexArrays(1, &s->vao);D;
        glBindVertexArray(s->vao);D;
        glBindBuffer(GL_ARRAY_BUFFER, vbo);D;
        dvariable_setpointers(d, s->program);D;
170     glBindBuffer(GL_ARRAY_BUFFER, 0);D;
```

```
  glBindVertexArray(0);D;
}

void step_end(struct step *s) {
  glDeleteProgram(s->program);D;
  s->program = 0;
  glDeleteVertexArrays(1, &s->vao);D;
  s->vao = 0;
}

void step_start(struct step *s, double samplerate, struct view *v) {
  glUseProgram(s->program);D;
  glUniform1f(s->sr, samplerate);D;
  glUniform1i(s->iters, 360);D;
  glUniformMatrix4fv(s->model, 1, GL_FALSE, &v->matrix[0][0]);D;
  glUniform4fv(s->center, 1, &v->center[0]);D;
  glUniform1f(s->radius, v->radius);D;
  s->dithersize = 2.0f / (v->height * v->tilesy);
  glUseProgram(0);D;
}

void step_do(struct step *s, GLuint vbo, GLuint count, int keep) {
  glEnable(GL_RASTERIZER_DISCARD);
  glBindVertexArray(s->vao);D;
  glUseProgram(s->program);
  glUniform1i(s->keep, keep);D;
#define NOISE (0.00001 * (rand() / (double) RAND_MAX - 0.5))
  for (int i = 0; i < 2 * (s->d + 1); ++i) {
    s->noise[i] = NOISE;
  }
  glUniform2fv(s->perturb, s->d + 1, s->noise);D;
#undef NOISE
#define NOISE ((rand() / (double) RAND_MAX - 0.5) * s->dithersize)
  glUniform2f(s->dither, NOISE, NOISE);D;
#undef NOISE
  glBindBufferBase(GL_TRANSFORM_FEEDBACK_BUFFER, 0, vbo);D;
  glBeginTransformFeedback(GL_POINTS);D;
  glDrawArrays(GL_POINTS, 0, count);D;
  glEndTransformFeedback();D;
  glBindBufferBase(GL_TRANSFORM_FEEDBACK_BUFFER, 0, 0);D;
  glUseProgram(0);D;
  glBindVertexArray(0);D;
  glDisable(GL_RASTERIZER_DISCARD);
}
```

57 src/utility.c

```
template <typename T> T& max(T& a, T& b) {
  return a > b ? a : b;
}

int ceil2n(int z) {
  int n = 1;
  while (0 < n && n < z) {
    n <<= 1;
  }
  return n;
```

```
}

double time_difference(const struct timespec *t1, const struct timespec *t0) {
  return (t1->tv_sec - t0->tv_sec) + (t1->tv_nsec - t0->tv_nsec) / 1000000000.0;
}
```

58 src/view.c

```
void view_init(struct view *v, int width, int height, double centerx, double centery, double centerz, double centerw, double radius, double delay) {
  v->width = width;
  v->height = height;
  v->matrix = glm::mat4(0.0f);
  v->matrix[0][0] = 1;
  v->matrix[0][1] = 1;
  v->matrix[1][2] = 1;
  v->matrix[1][3] = 1;
  v->center[0] = centerx;
  v->center[1] = centery;
  v->center[2] = centerz;
  v->center[3] = centerw;
  v->radius = radius;
  v->tilex = 0;
  v->tiley = 0;
  v->tilesx = 1;
  v->tilesy = 1;
  v->delay = delay;
}

void view_copy(struct view *dst, struct view *src) {
  dst->width = src->width;
  dst->height = src->height;
  dst->matrix = src->matrix;
  dst->center = src->center;
  dst->radius = src->radius;
  dst->tilex = src->tilex;
  dst->tiley = src->tiley;
  dst->tilesx = src->tilesx;
  dst->tilesy = src->tilesy;
  dst->delay = src->delay;
}

void view_reshape(struct view *v, int width, int height) {
  v->width = width;
  v->height = height;
}

void view_reset(struct view *v, double centerx, double centery, double centerz, double centerw, double radius) {
  v->center[0] = centerx;
  v->center[1] = centery;
  v->center[2] = centerz;
  v->center[3] = centerw;
  v->radius = radius;
}

void view_plane(struct view *v, int which) {
```

```
  v->matrix = glm::mat4(0.0f);
  v->matrix[0][0] = 1;
  v->matrix[0][1] = which;
  v->matrix[1][2] = 1;
  v->matrix[1][3] = which;
}

void view_coord(glm::vec4 *c, struct view *v, int px, int py) {
  glm::vec4 p = glm::vec4(float((px - v->width / 2.0) * 2.0 / v->height), float
      ↪ ((v->height / 2.0 - py) * 2.0 / v->height), 0.0f, 0.0f);
  *c = v->matrix * (v->radius * p) + v->center;
}

void view_center(struct view *v, glm::vec4 &c) {
  v->center = c;
}

void view_zoom(struct view *v, glm::vec4 &c, float z) {
  float g = pow(0.5, z);
  v->center = g * v->center + (1 - g) * c;
  v->radius *= g;
}

void view_zoom(struct view *v, float z) {
  view_zoom(v, v->center, z);
}
```

59 src/view.h

```
#ifndef VIEW_H
#define VIEW_H 1

struct view {
  int width;
  int height;
  glm::mat4 matrix;
  glm::vec4 center;
  float radius;
  int tilex, tiley, tilesx, tilesy;
  float delay;
};

void view_init(struct view *v, int width, int height, double centerx, double
    ↪ centery, double centerz, double centerw, double radius, double delay);
void view_copy(struct view *dst, struct view *src);
void view_reshape(struct view *v, int width, int height);
void view_reset(struct view *v, double centerx, double centery, double centerz,
    ↪ double centerw, double radius);
void view_plane(struct view *v, int which);
void view_coord(glm::vec4 *c, struct view *v, int px, int py);
void view_center(struct view *v, glm::vec4 &c);
void view_zoom(struct view *v, float z);
void view_zoom(struct view *v, glm::vec4 &c, float z);

#endif
```