Yaxx Installation Guide

Table of Contents

1 Overview

This file describes the installation process for *yaxx* (Yet Another X-ray Xtractor). *Yaxx* is a Perl script that facilitates batch spectral processing of X-ray data using Perl open source tools and commonly available astronomical software (CIAO, SAS, HEASOFT).

2 Requirements

Operating system

Yaxx has been tested on these operating systems:

- Linux: RedHat (FC4)
- Solaris: Version 2.8.

Support for MacOS X is awaiting a patch to a bug in the CIAO *dmcoords* tool. Parties interested in using *yaxx* on MacOS X should contact the author directly about a possible workaround.

Disk space

The build and installation process requires approximately 200 Mb of free disk space (due mostly to a few perl modules used by *yaxx*). After installation and testing, one can remove the top-level `build` and `pkgs` directories to free up around 110 Mb of disk space.

Perl

Perl version 5.8.0 or newer. All non-standard packages used by *yaxx* are included in this distribution.

CIAO

CIAO version 3.3 or subsequent patch releases of 3.3 (for instance 3.3.0.1).

LaTeX

This is required if LaTeX-generated postscript fit summaries are desired.

3 Installation

To install and test *yaxx*, do the following steps.

Enter the CIAO environment

Source the appropriate initialization script (as instructed in the CIAO startup thread) to enter the CIAO analysis environment. For `csh` or `tcsh` do:

```
source <CIAO_HOME>/bin/ciao.csh
```

Note that if the FTOOLS package is used within the same session as CIAO then the FTOOLS initialization must be done *before* CIAO initialization. Read the CIAO and FTOOLS section further details.

Check perl version

Make sure your perl has version 5.8.0 or newer by checking the output of entering ’perl -v’ at your command prompt:

```
perl -v
```

Change dir to install directory

Change dir to the directory where you want the *yaxx* installation to reside. The only requirement is that sufficient disk space be available. For example if you do

```
cd /path/to/install
```

then the *yaxx* package will be installed in `/path/to/install/yaxx/`.

Untar the source

Untar the *yaxx* source tree by doing the following:

```
gunzip --stdout yaxx.tar.gz | tar xvf -
```

This will create a 'yaxx-<version>' directory and put the contents there.

Make link and change directory to the yaxx directory

```
ln -s yaxx-<version> yaxx
cd yaxx
```

Set up the analysis package initialization

Yaxx depends on external analysis packages such as CIAO (required) and SAS and HEAsoft (for *XMM*). To minimize potential package conflicts, *yaxx* does the analysis environment initialization internally (in the correct order) and expects that the user has *not* done any initialization. It is necessary therefore to tell *yaxx* how to initialize each of the analysis packages, typically by sourcing an initialization script.

The environment setup is specified in the System configuration file `yaxx.cfg` (described in the reference manual. Edit this file and change the `environment` parameters to be appropriate for your system setup. The initial installation and testing uses the *Chandra* analysis thread, which only requires setting the CIAO environment:

```
<environment CIAO>
  shell  csh
  script source /soft/ciao/bin/ciao.csh  <<--- CHANGE THIS
</environment>
```

Users doing *XMM* analysis will have to adjust the HEAsoft and SAS enviroments as well.

Make the required libraries and modules

This is the longest step and can take over an hour on a Solaris machine, but on a modern linux box it should finish within 20 minutes. At the command prompt enter:

```
make
```

This runs the perl script `install_yaxx.pl` which puts the necessary perl modules in the `yaxx-perl` directory of the source distribution.

If any library or module fails to compile or pass its internal tests, the install script will ask if you want to quit or continue. See the Known Issues section if this occurs. Depending on the severity of the failure, it may be possible to push through and still have a working system. Please email the developer with details and a copy of the installation log `install.log` if any modules do not compile.

Test *yaxx*

Test *yaxx* by doing:

```
make test
```

You should see that *yaxx* does not stop with any error messages and finishes with the message (except for the time):

```
<2006-Nov-05 18:44:04> *** SUCCESS for Obsid=3102  Ccdid=ACIS-S3  Srcid=1

Creating HTML report index
```

If the test step does not succeed, please email the developer with details and a copy of the log file which is created in `Test/install/Log`. Note that on some systems the *make* program may indicate a failure, but if *yaxx* ends with the above message then everything should be OK.

The final product is a report (`Test/install/obs3102/src1/report.ps`) which summarizes the fitting. There will also be a file `Test/install/report_index.html` which can be viewed with your favorite web browser, e.g.

```
firefox Test/install/report_index.html
```

Create an executable in your path or an alias

The 'make' step also creates a customized shell script 'yaxx' in the source directory which should be used to actually start *yaxx* for analysis. Copy this file to a directory which is in your PATH (e.g. ~/bin) and do 'rehash' (if you are running csh or tcsh).

A less desirable but valid option is to make an alias and put this in your shell initialization script (e.g. `~/.cshrc`):

```
alias yaxx /path/to/install/yaxx/yaxx
```

4 Known issues

PGPLOT Version 2.19

PGPLOT is known to fail its self-tests on some Solaris machines even though the code is actually perfectly OK. This is a known issue (see the PGPLOT module documentation) and results in errors like:

```
============== Running test1.p ==============
Can't load 'blib/arch/auto/PGPLOT/PGPLOT.so' for module PGPLOT: ld.so.1: perl5.8.6: ....
 at test1.p line 3
Compilation failed in require at test1.p line 3.
BEGIN failed--compilation aborted at test1.p line 3.
```

This module specifies the order of load libraries in a way that is incompatible with the Solaris compilers. We have patched this in the *yaxx* distribution and submitted a bug report to the PGPLOT developer.

ExtUtils::F77 Version 1.15 and recent Solaris compilers

The Solaris compiler suite *Studio10* is not compatible with ExtUtils::F77 version 1.15 and earlier. We have patched this in the *yaxx* distribution and submitted a bug report to the ExtUtils::F77 developer.

Perl installation must be compiled with the current compiler suite

One issue arose because the compiler suite had been upgraded without recompiling Perl. This can lead to obscure errors when building the modules, in particular PGPLOT.

This came up on two Solaris systems. In these cases the issue was resolved without having to recompile perl by reverting to an older compiler with a command similar to:

```
set path = ( /opt/SUNWspro_6.1/bin $path )
```

Details will vary with your installation.

Compress::Zlib

On one linux machine (Fedora Core 4 with Perl 5.8.6) the Compress::Zlib library failed a self-test. The installation process was continued with no further issues and was ultimately successful.