GNU Data Language (GDL)

a free and open-source implementation of IDL

Sylwester Arabas
University of Warsaw, Poland

Alain Coulais
CNRS LERMA, Observatoire de Paris, France

Takeshi Enomoto
JAMSTEC, Yokohama, Japan

GDL development is led by Marc Schellens, the project founder

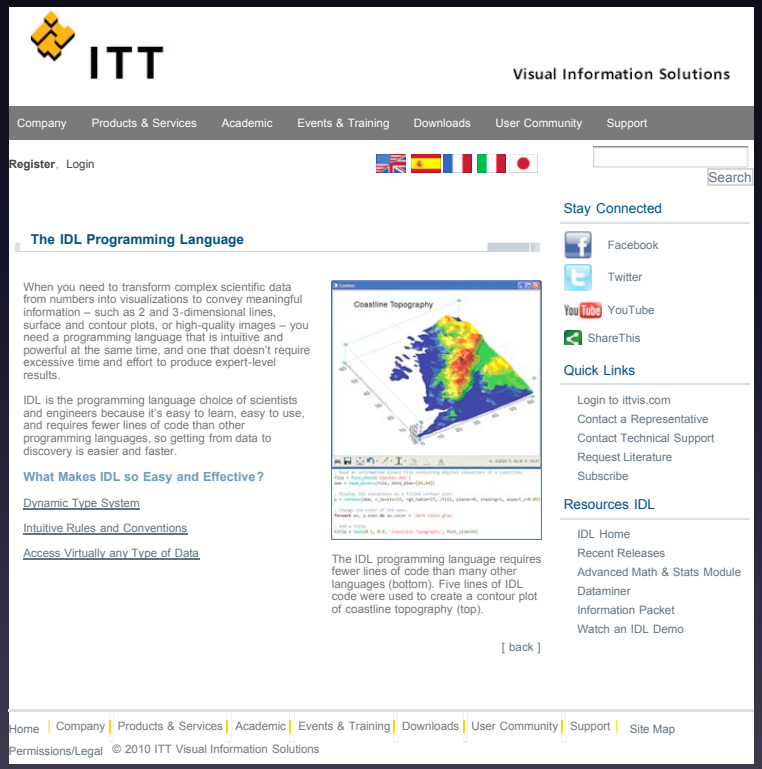
August 20th 2010, JAMSTEC, Yokohama, Japan

Plan of the talk

- About GDL
  - what's GDL (and IDL)
  - why to use GDL, who uses it and what for
  - how to get GDL
  - how to help and contribute
- Hands-on demo
  - running the GDL interpreter
  - syntax basics
  - opening an example dataset
  - simple array manipulation
  - basic plotting functionalities

before we begin with GDL: what's IDL

http://www.ittvis.com/



- IDL is a product and registered trade mark of ITT Visual Information Solutions
- IDL is a tool for data analysis and visualisation
- IDL is a programming language
- IDL package contains: interpreter/compiler, library routines, documentation and a development environment
- IDL is related with GDL as Matlab with Octave/Scilab, M$ Office with OO.org etc

IDL in Astronomy

- The situation
  - IDL is widely used in astronomy, since 1977, including a lot of Space Missions (official pipelines)
  - Several well known libraries (Astron, MPFIT, Solar Soft, ...) used as commodity by Scientists
  - Code is rather well independant of OS's (VMS, many Unix, OSX, Linux and also MS-win) and can live during very long time
  - Fast coding, nice for testing idea/modles/processing during data analysis

IDL in Astronomy

- The situation
  - IDL is widely used in astronomy, since 1977, including a lot of Space Missions (official pipelines)
  - Several well known libraries (Astron, MPFIT, Solar Soft, ...) used as commodity by Scientists
  - Code is rather well independant of OS's (VMS, many Unix, OSX, Linux and also MS-win) and can live during very long time
  - Fast coding, nice for testing idea/modles/processing during data analysis
- The problems ?
  - Dependence. IDL is not open source free software. We had serious warning in the past, some platforms (Unix, VMS) had desapear. Despite support by NASA, editor may desapear. Some choices by editor (widgets, limited graphical outputs, ...) upset users. Because of the large codebase, we cannot afford disparition!
  - No general purpose alternative. Now Python in Astronomy become a real alternative, but it was not he case 3 or 5 years ago.

IDL in Astronomy

- The situation
  - IDL is widely used in astronomy, since 1977, including a lot of Space Missions (official pipelines)
  - Several well known libraries (Astron, MPFIT, Solar Soft, ...) used as commodity by Scientists
  - Code is rather well independant of OS's (VMS, many Unix, OSX, Linux and also MS-win) and can live during very long time
  - Fast coding, nice for testing idea/modles/processing during data analysis
- The problems ?
  - Dependence. IDL is not open source free software. We had serious warning in the past, some platforms (Unix, VMS) had desapear. Despite support by NASA, editor may desapear. Some choices by editor (widgets, limited graphical outputs, ...) upset users. Because of the large codebase, we cannot afford disparition!
  - No general purpose alternative. Now Python in Astronomy become a real alternative, but it was not he case 3 or 5 years ago.
- Solution? Having a free clone.

About GDL: the project and its authors

- aim: develop an open-source drop-in replacement for IDL

About GDL: the project and its authors

- aim: develop an open-source drop-in replacement for IDL
- project founder and leader: Marc Schellens

About GDL: the project and its authors

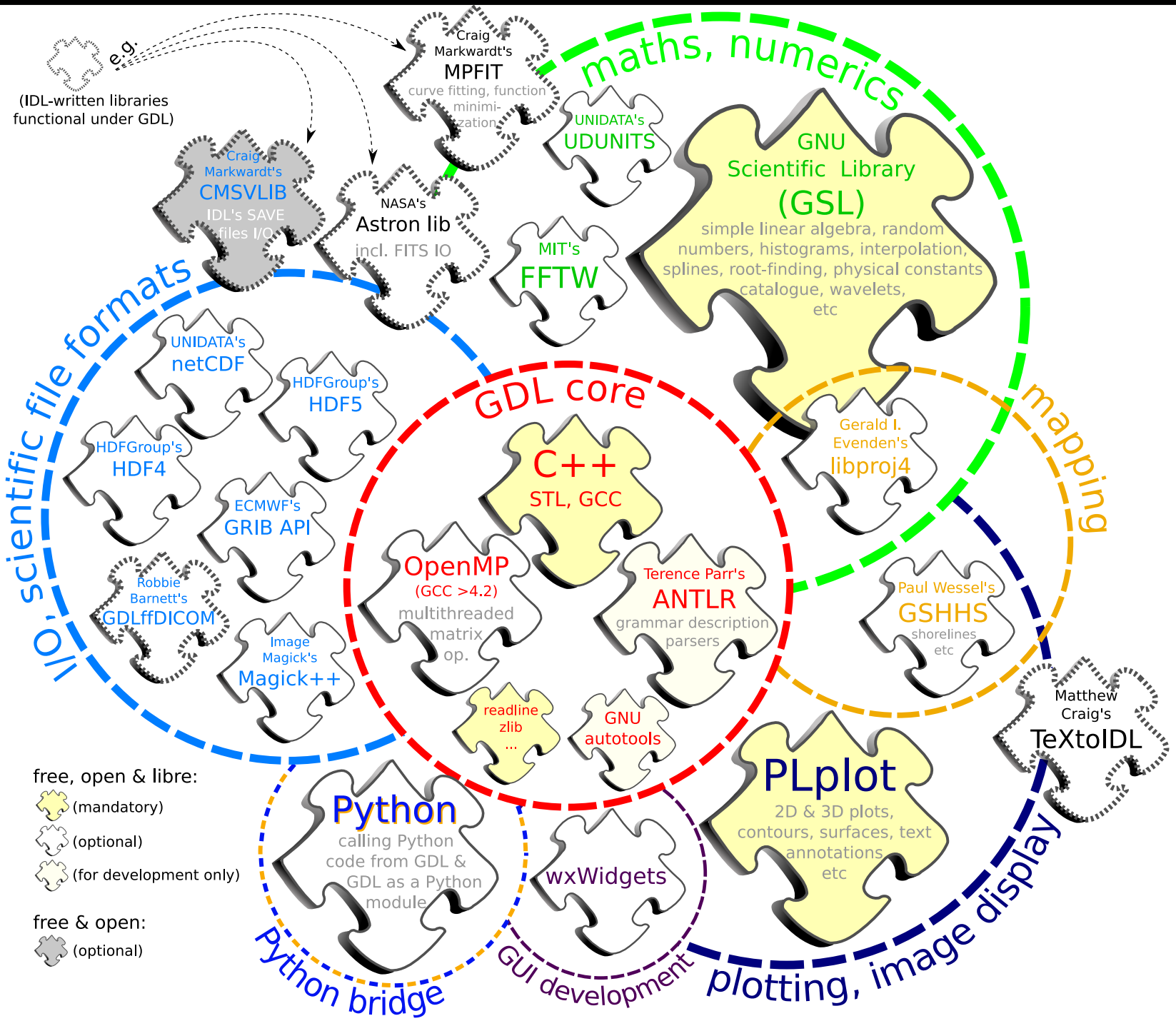
- aim: develop an open-source drop-in replacement for IDL
- project founder and leader: Marc Schellens
- project team:
  - volounteers from around the world including:
    Joel Gales, Pierre Chanial, Peter Messmer, ...
  - Alain's students including:
    N. Galmiche, M. Lenoir, G. Marshall, L. Noreskal, T. Mermet
  - users sending bug-reports, patches, comments, ...
  - packagers including Takeshi Enomoto from JAMSTEC

About GDL: the project and its authors

- aim: develop an open-source drop-in replacement for IDL
- project founder and leader: Marc Schellens
- project team:
  - volounteers from around the world including:
    Joel Gales, Pierre Chanial, Peter Messmer, ...
  - Alain's students including:
    N. Galmiche, M. Lenoir, G. Marshall, L. Noreskal, T. Mermet
  - users sending bug-reports, patches, comments, ...
  - packagers including Takeshi Enomoto from JAMSTEC
- current status:
  - compiler/interpreter: generally complete
  - library routines: partially implemented (ca. 400 routines)
  - documentation: virtually nothing (IDL docs are on the web)
  - development environment: no plans

About GDL: the project and its authors

- aim: develop an open-source drop-in replacement for IDL
- project founder and leader: Marc Schellens
- project team:
  - volounteers from around the world including:
    Joel Gales, Pierre Chanial, Peter Messmer, ...
  - Alain's students including:
    N. Galmiche, M. Lenoir, G. Marshall, L. Noreskal, T. Mermet
  - users sending bug-reports, patches, comments, ...
  - packagers including Takeshi Enomoto from JAMSTEC
- current status:
  - compiler/interpreter: generally complete
  - library routines: partially implemented (ca. 400 routines)
  - documentation: virtually nothing (IDL docs are on the web)
  - development environment: no plans
- more info: http://gnudatalanguage.sourceforge.net/


e.g.
(IDL-written libraries functional under GDL)
Craig Markwardt's MPFIT
curve fitting, function minimi-zation
Craig Markwardt's CMSVLIB
IDL's SAVE files I/O
NASA's Astron lib
incl. FITS IO
maths, numerics
UNIDATA's UDUNITS
GNU Scientific Library (GSL)
simple linear algebra, random numbers, histograms, interpolation, splines, root-finding, physical constants catalogue, wavelets, etc
MIT's FFTW
I/O, scientific file formats
UNIDATA's netCDF
HDFGroup's HDF5
HDFGroup's HDF4
ECMWF's GRIB API
Robbie Barnett's GDLffDICOM
Image Magick's Magick++
GDL core
C++
STL, GCC
OpenMP
(GCC >4.2)
multithreaded matrix op.
Terence Parr's ANTLR
grammar description parsers
readline
zlib
...
GNU autotools
mapping
Gerald I. Evenden's libproj4
Paul Wessel's GSHHS
shorelines etc
Matthew Craig's TeXtoIDL
PLplot
2D & 3D plots, contours, surfaces, text annotations etc
plotting, image display
Python
calling Python code from GDL & GDL as a Python module
Python bridge
wxWidgets
GUI development
free, open & libre:
(mandatory)
(optional)
(for development only)
free & open:
(optional)

About GDL: some pros and cons

- inherited from IDL:
  - one can process data and get customized plots of them with a few (dozens) lines of code
  - opens many data formats: text, binary, netCDF, HDF, FITS, GRIB...

About GDL: some pros and cons

- inherited from IDL:
  - one can process data and get customized plots of them with a few (dozens) lines of code
  - opens many data formats: text, binary, netCDF, HDF, FITS, GRIB...
- GDL specific:
  - it's free! and you can modify and expand it to suit your needs
  - numerics based on reliable open-source libraries as (GSL, FFTW, ...)
  - enhanced CLI thanks to readline, SVG & GRIB support, Python bridge
  - compiles on many platforms (Linux, BSDs, OSX, Solaris, Cygwin, ...)

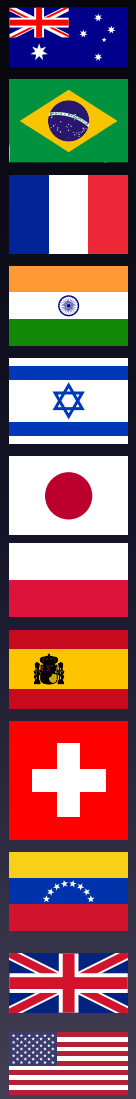
About GDL: some pros and cons

- inherited from IDL:
  - one can process data and get customized plots of them with a few (dozens) lines of code
  - opens many data formats: text, binary, netCDF, HDF, FITS, GRIB...
- GDL specific:
  - it's free! and you can modify and expand it to suit your needs
  - numerics based on reliable open-source libraries as (GSL, FFTW, ...)
  - enhanced CLI thanks to readline, SVG & GRIB support, Python bridge
  - compiles on many platforms (Linux, BSDs, OSX, Solaris, Cygwin, ...)
- inherited from IDL:
  - bit archaic syntax with constructs from numerous languages from Fortran to C++ and inconsistent naming of library routines ... but who cares, it's meant to be more a bash than a Java of scientific computing – it's intended for short scripts and interactive usage

About GDL: some pros and cons

- inherited from IDL:
  - one can process data and get customized plots of them with a few (dozens) lines of code
  - opens many data formats: text, binary, netCDF, HDF, FITS, GRIB...
- GDL specific:
  - it's free! and you can modify and expand it to suit your needs
  - numerics based on reliable open-source libraries as (GSL, FFTW, ...)
  - enhanced CLI thanks to readline, SVG & GRIB support, Python bridge
  - compiles on many platforms (Linux, BSDs, OSX, Solaris, Cygwin, ...)
- inherited from IDL:
  - bit archaic syntax with constructs from numerous languages from Fortran to C++ and inconsistent naming of library routines ... but who cares, it's meant to be more a bash than a Java of scientific computing – it's intended for short scripts and interactive usage
- GDL specific:
  - not all library routines are implemented yet
  - some useful keywords are missing
  - lack of documentation (but IDL docs are on the web)

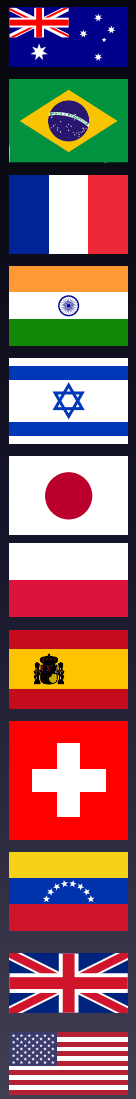
About GDL: known users



- people at the Paris Observatory, Lyon Observatory, Strasbourg Observatory

...?

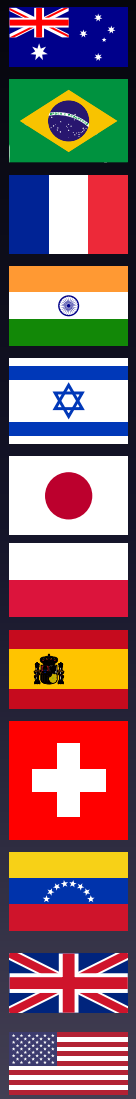
About GDL: known users



- people at the Paris Observatory, Lyon Observatory, Strasbourg Observatory
- people at NASA (GSFC, JPL, . . .) and MIT

...?

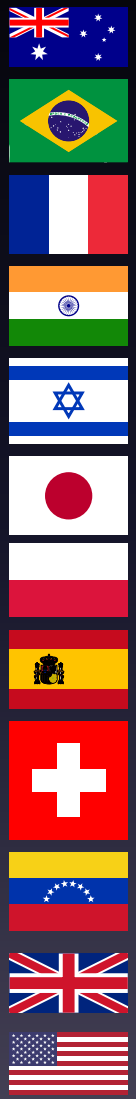
About GDL: known users



- people at the Paris Observatory, Lyon Observatory, Strasbourg Observatory
- people at NASA (GSFC, JPL, . . .) and MIT
- people in Max Planck's labs

...?

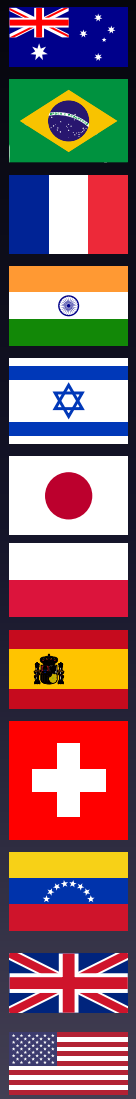
About GDL: known users



- people at the Paris Observatory, Lyon Observatory, Strasbourg Observatory
- people at NASA (GSFC, JPL, . . .) and MIT
- people in Max Planck's labs
- people at the Lockheed Martin Solar and Astrophysics Laboratory and Stanford University (2 conf. papers on GDL)

...?

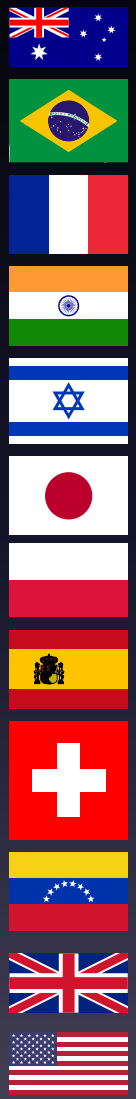
About GDL: known users



- people at the Paris Observatory, Lyon Observatory, Strasbourg Observatory
- people at NASA (GSFC, JPL, . . .) and MIT
- people in Max Planck's labs
- people at the Lockheed Martin Solar and Astrophysics Laboratory and Stanford University (2 conf. papers on GDL)
- contacts in Australia, Brazil, India, Israel, Japan, Spain, Switzerland, UK, USA, Venezuela... (astro- and geophysicists)

...?

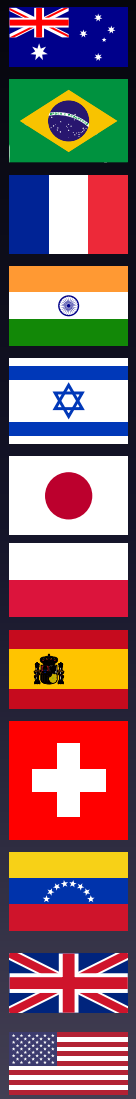
About GDL: known users



- people at the Paris Observatory, Lyon Observatory, Strasbourg Observatory
- people at NASA (GSFC, JPL, . . .) and MIT
- people in Max Planck's labs
- people at the Lockheed Martin Solar and Astrophysics Laboratory and Stanford University (2 conf. papers on GDL)
- contacts in Australia, Brazil, India, Israel, Japan, Spain, Switzerland, UK, USA, Venezuela... (astro- and geophysicists)
- some guys in the medical imaging comunity (the GDLffDICOM package)

...?

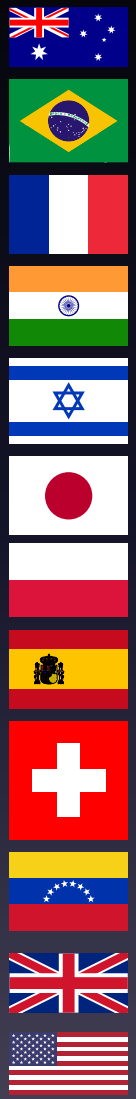
About GDL: known users



- people at the Paris Observatory, Lyon Observatory, Strasbourg Observatory
- people at NASA (GSFC, JPL, . . .) and MIT
- people in Max Planck's labs
- people at the Lockheed Martin Solar and Astrophysics Laboratory and Stanford University (2 conf. papers on GDL)
- contacts in Australia, Brazil, India, Israel, Japan, Spain, Switzerland, UK, USA, Venezuela... (astro- and geophysicists)
- some guys in the medical imaging comunity (the GDLffDICOM package)
- Sylwester's students at the University of Warsaw

...?

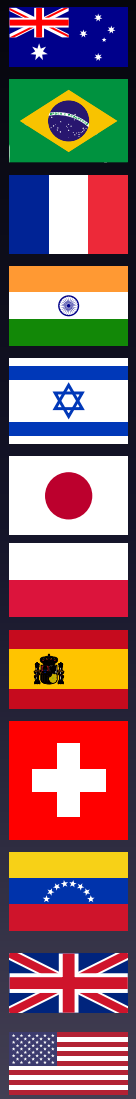
About GDL: known users



- people at the Paris Observatory, Lyon Observatory, Strasbourg Observatory
- people at NASA (GSFC, JPL, . . .) and MIT
- people in Max Planck's labs
- people at the Lockheed Martin Solar and Astrophysics Laboratory and Stanford University (2 conf. papers on GDL)
- contacts in Australia, Brazil, India, Israel, Japan, Spain, Switzerland, UK, USA, Venezuela... (astro- and geophysicists)
- some guys in the medical imaging comunity (the GDLffDICOM package)
- Sylwester's students at the University of Warsaw
- students at the Dept of Physics, Montana State Univ.

...?

About GDL: known users



...?

- people at the Paris Observatory, Lyon Observatory, Strasbourg Observatory
- people at NASA (GSFC, JPL, . . .) and MIT
- people in Max Planck's labs
- people at the Lockheed Martin Solar and Astrophysics Laboratory and Stanford University (2 conf. papers on GDL)
- contacts in Australia, Brazil, India, Israel, Japan, Spain, Switzerland, UK, USA, Venezuela... (astro- and geophysicists)
- some guys in the medical imaging comunity (the GDLffDICOM package)
- Sylwester's students at the University of Warsaw
- students at the Dept of Physics, Montana State Univ.
- and many more... (assuming from the number of downloads, packages, forum posts)

About GDL: how we use it

- @ the Observatoire de Paris:
  - analysing time series (several satelittes)
  - modelization and inversion of time responses of various detectors: IR detectors, bolometers
  - 2D deconvolution methods for Radio Interferometry (Clean, MEM, and alternative)
  - teaching various topics, at different levels: FFT, image processing ...

About GDL: how we use it

- @ the Observatoire de Paris:
  - analysing time series (several satelittes)
  - modelization and inversion of time responses of various detectors: IR detectors, bolometers
  - 2D deconvolution methods for Radio Interferometry (Clean, MEM, and alternative)
  - teaching various topics, at different levels: FFT, image processing ...
- @ the University of Warsaw:
  - aircraft & LIDAR measurements data analysis and visualisation
  - teaching meteorological data processing:
    - geosci. model ouptut data: interpolation, subsetting, statistics
    - signals processing: FFT, wavelets, principal components, filtering
    - satellite image and weather-forecast display/processing
    - plotting, data formats: CVS, netCDF, HDF4, HDF5, GRIB
  - "operational" weather forecast visualisation (GRIB+SVG)

About GDL: how others use it

- At Strasbourg and Lyon:
  doing heavy computations without blocking IDL licences
  (GDL cited in an MNRAS article!)

About GDL: how others use it

- At Strasbourg and Lyon:
  doing heavy computations without blocking IDL licences (GDL cited in an MNRAS article!)
- at LESIA, Paris Obs.:
  substituting IDL by GDL in an automatic pipeline from a given format (PDS used in Planetary mission) to a new one (Virtual Observatory, XML ...). Gaining perenity on very long time scale (space missions: 10 to 30 years).

About GDL: how others use it

- At Strasbourg and Lyon:
  doing heavy computations without blocking IDL licences
  (GDL cited in an MNRAS article!)
- at LESIA, Paris Obs.:
  substituting IDL by GDL in an automatic pipeline from a given format (PDS used in Planetary mission) to a new one (Virtual Observatory, XML ...). Gaining perenity on very long time scale (space missions: 10 to 30 years).
- at Lockheed Martin/Stanford:
  deconvolution of images from solar telescopes using a CUDA-enhanced GDL (poster at NVIDIA conference!)

About GDL: how others use it

- At Strasbourg and Lyon:
  doing heavy computations without blocking IDL licences (GDL cited in an MNRAS article!)
- at LESIA, Paris Obs.:
  substituting IDL by GDL in an automatic pipeline from a given format (PDS used in Planetary mission) to a new one (Virtual Observatory, XML ...). Gaining perenity on very long time scale (space missions: 10 to 30 years).
- at Lockheed Martin/Stanford:
  deconvolution of images from solar telescopes using a CUDA-enhanced GDL (poster at NVIDIA conference!)
- at Montana State University:
  web-based remote access to data analysis environment for students of an IDL course

GDL - GNU Data Language Beta by m_schellens

Summary | Files | Support | Develop | Tracker | Mailing Lists | Forums | Code

GDL - GNU Data Language, a free IDL (Interactive Data Language, see http://ittvis.com/idl/) compatible incremental compiler.

Download Now!
gdl-0.9rc4.tar.gz (1.5 MB)

OR View all files

Project Reviews

That's a very good work.
posted by astroste 313 days ago — Was this review helpful for you? Yes or No

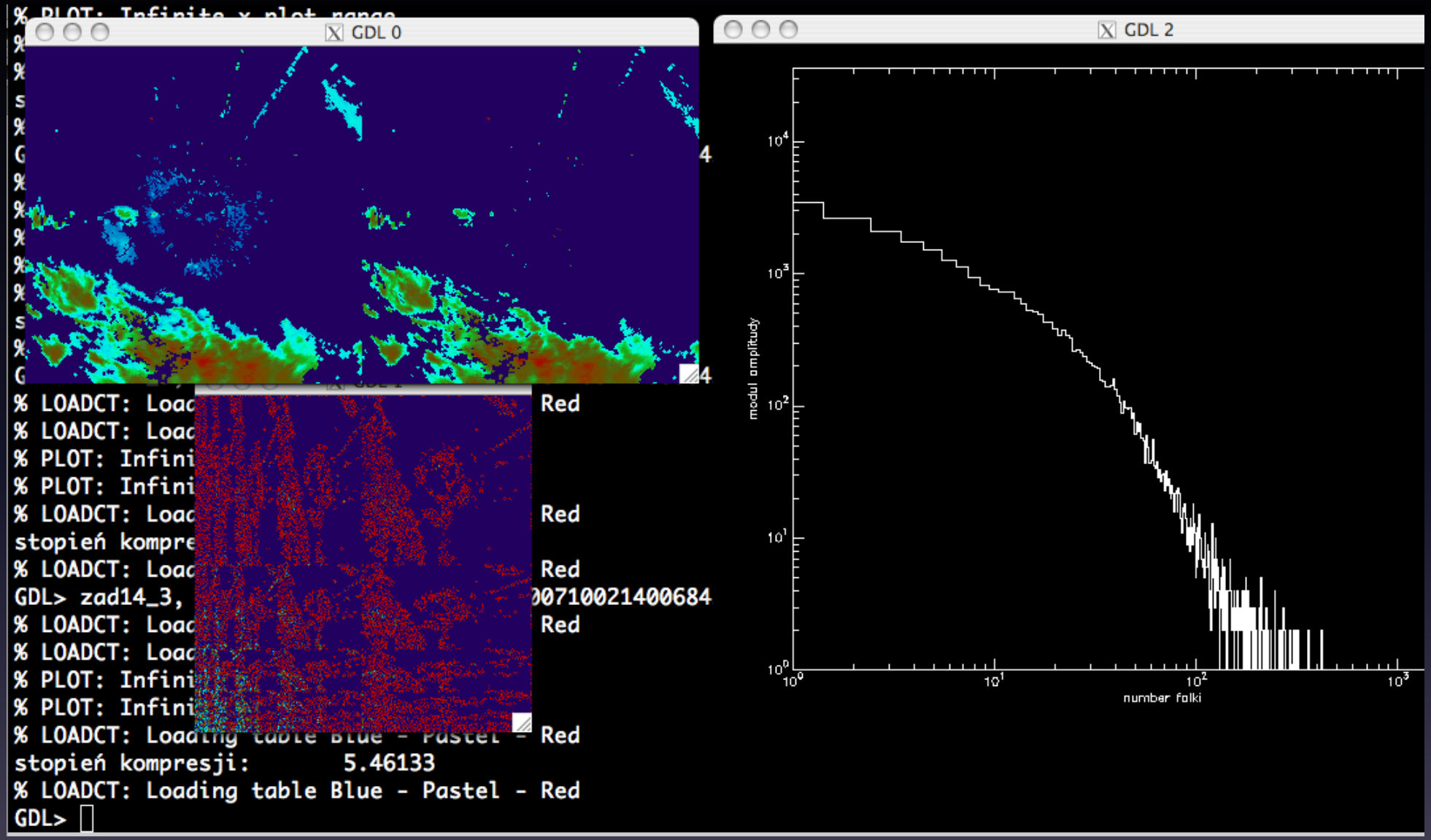
An incomplete but functional and very useful Interactive Data Language interpreter.
posted by anonymous 359 days ago — Was this review helpful for you? Yes or No

Extremely useful!
posted by anonymous 288 days ago — Was this review helpful for you? Yes or No

High funcionality!
posted by anonymous 134 days ago — Was this review helpful for you? Yes or No

GDL is a great chance to introduce IDL to students. Thanks to all developers!
posted by anonymous 197 days ago — Was this review helpful for you? Yes or No

Bit of a pain to compile, but generally works well and makes it quick and easy to do sophisticated data analysis.
posted by anonymous 244 days ago — Was this review helpful for you? Yes or No


GDL 0
GDL 2
% LOADCT: Loac Red
% LOADCT: Loac
% PLOT: Infini
% PLOT: Infini
% LOADCT: Loac Red
stopień kompre
% LOADCT: Loac Red
GDL> zad14_3, 00710021400684
% LOADCT: Loac Red
% LOADCT: Loac
% PLOT: Infini
% PLOT: Infini
% LOADCT: Loading table Blue - Pastel - Red
stopień kompresji: 5.46133
% LOADCT: Loading table Blue - Pastel - Red
GDL>
modul amplitudy
number falki

Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://localhost/babyeulag/panes.php?Grabowski_2008_IGF

↓ click here to run the set-up script and run the model solver

```
;                    Rd   es
; qv ?= RH qvs = RH -- ------
;                    Rv p - es
;
;                  | Lv  / 1    1 \ |
; es(T) = e0 * exp| -- | --- - -- ||
;                  |_Rv  \ T0   T /_|
;
es = ee0 * exp(hlatv / rv * (1 / tt0 - 1 /
  th[0, *, 0] * (p0 / prs[0, *, 0]) ^ (-rg
)))
qvs = rg / rv / (prs[0, *, 0] - es)
qvs *= temporary(es)
; RH increases linearly from 70% at the su
qv = fltarr(X_len, Z_len, 1, /nozero)
for z = 0,        half      do qv[*, z, 0]
for z = half + 1, Z_len - 1 do qv[*, z, 0]
ncdf_varput, nc, 'qv_e', qv[0, *, 0]

; circular bubble (r=.6km + 100m linear tr
; - increased temperature (+.5K)
; - increased water-vapour content (RH=95%
for b = 0, 1 do begin
  ;x0 = X_len * dx / 3.
  x0 = X_len * dx / 3. * (b + 1)
  z0 = Z_len * dz / 4.
  radius = 800
  for z = 0, Z_len - 1 do for x = 0, X_len
    dstnc = sqrt((x * dx - x0)^2 + (z * dx
    if dstnc ge radius + 100 then del = 0.
    else if dstnc ge radius then del = 1 -
```

Position: Ln 1, Ch 1 Total: Ln 149, Ch 6045

```
*** itime, time: 496 24.75
*** itime, time: 497 24.8
*** itime, time: 498 24.85
*** itime, time: 499 24.9
*** itime, time: 500 24.95
 surf max qr (g/kg): 5.56057e-05
 surf max prec rate (mm/hr): 0.000267464
GDL> exit
```

↓ click here to run the analysis script and produce plots

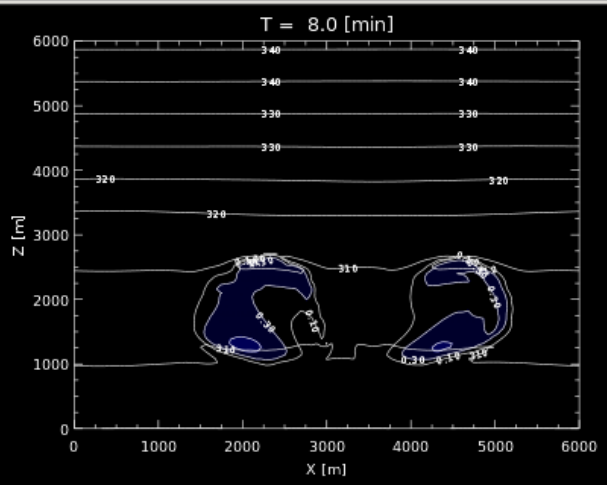
```
c = [Z_len, X_len, 1]
o = [0, 0, 0]

; animation loop
for i = 0, T_len - 1 do begin

  file = string(i, f='(I5.5)')
  device, filename = file + '.svg'
  o[2] = i

  ; getting a slice of data
  ncdf_varget, nc, ncdf_varid(nc, "T"),  T
  ncdf_varget, nc, ncdf_varid(nc, "qc"), q
  ncdf_varget, nc, ncdf_varid(nc, "qr"), q
  ncdf_varget, nc, ncdf_varid(nc, "th"), t

  ; plotting cloud water
  contour, qc, X, Z, /fill, lev=lev_qc/100
    xtitle = "X [m]", ytitle = "Z [m]", $
     title = "T = " + string(T, f='(F4.1)'
  contour, 1000 * qc, X, Z, /foll, /overpl

  ; plotting rain water
  wh = where(qr gt 10e-5, cnt)
  if cnt gt 0 then oplot, psym=3, $
    (rebin(X, X_len, Z_len, /s))[wh], $
    (rotate(rebin(Z, Z_len, X_len, /s), 1)

  ; plotting potential temperature
  contour, th, X, Z, /foll, /overplot, lev

  device, /close

endfor
```

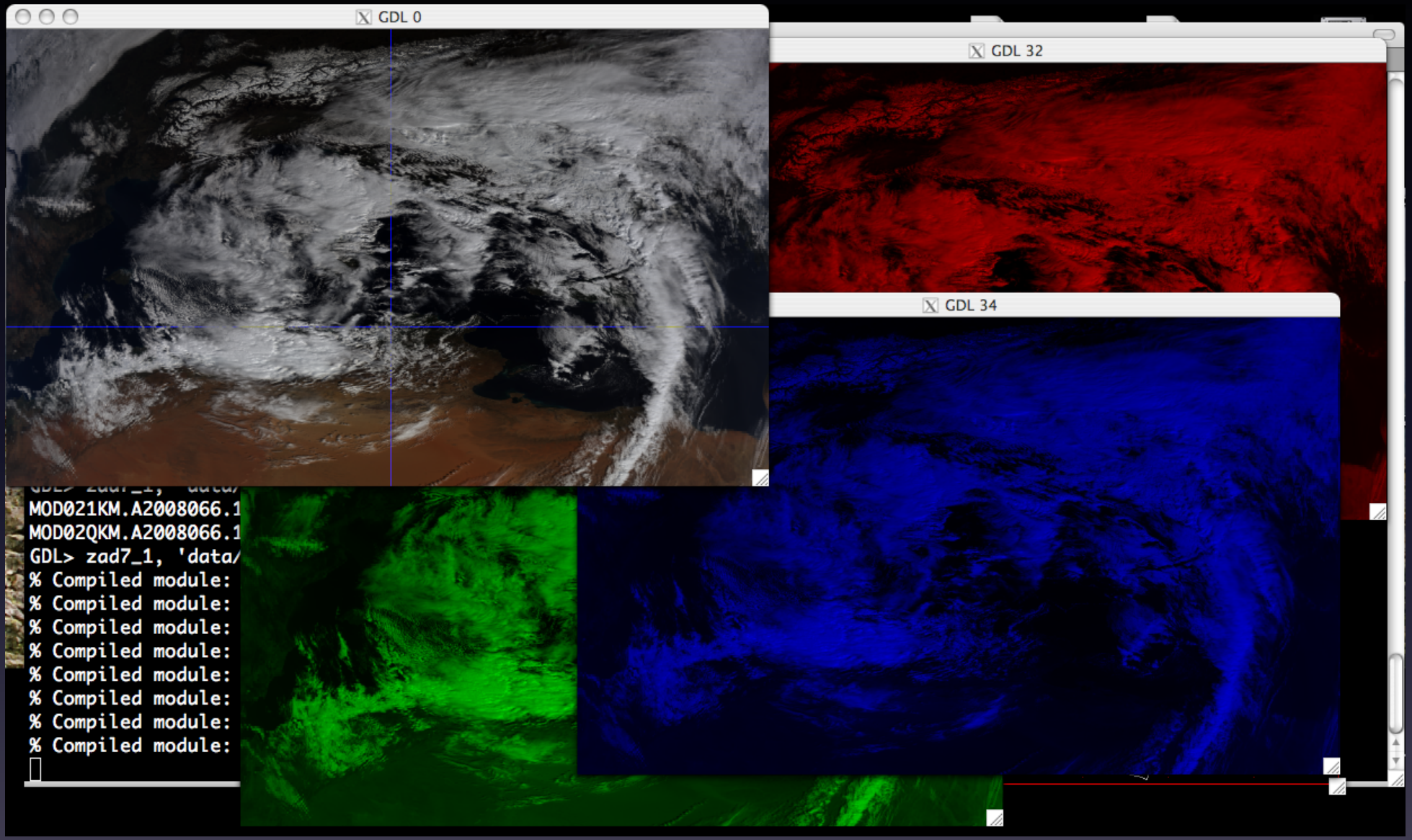
Position: Ln 1, Ch 1 Total: Ln 51, Ch 1379

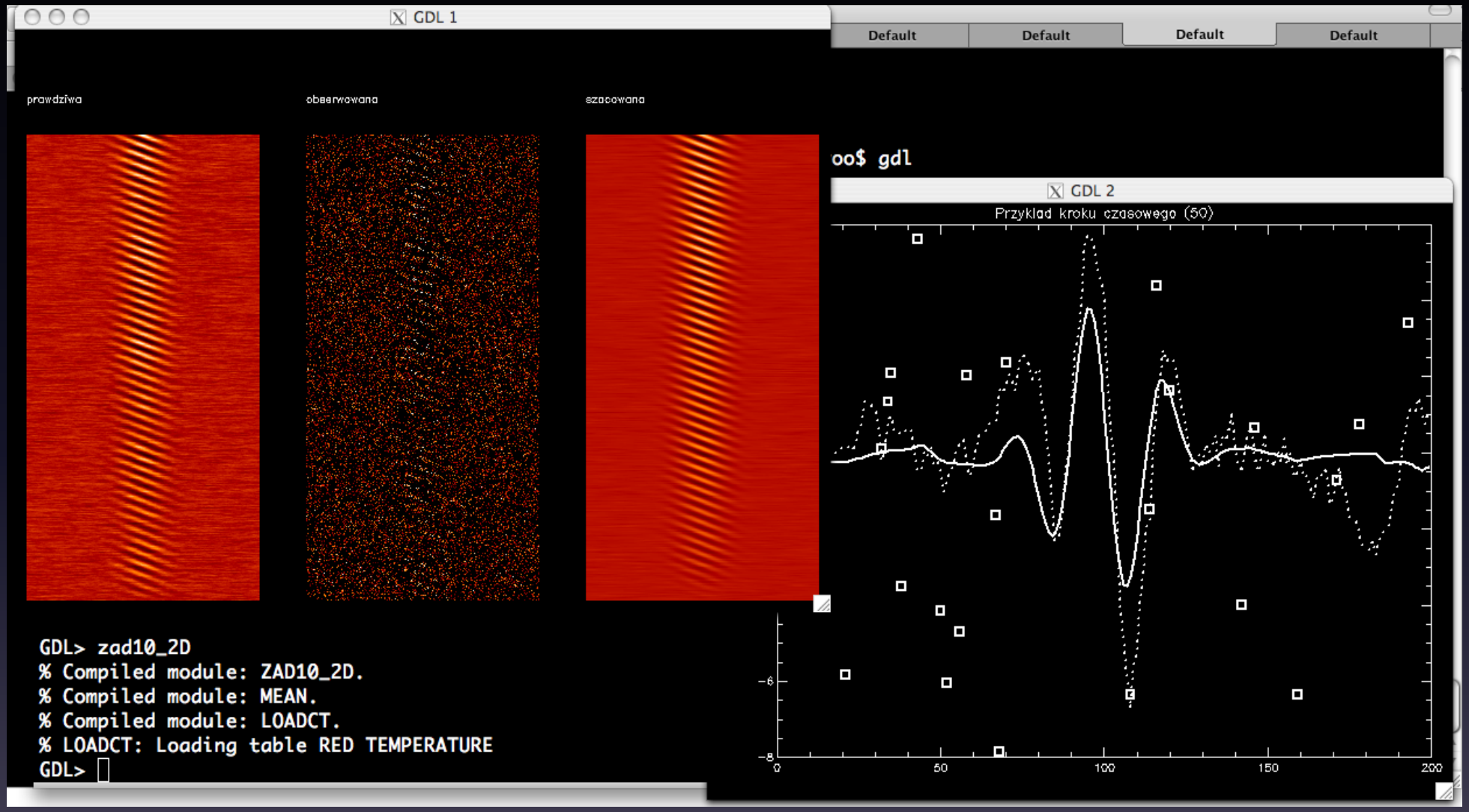
```
% Compiled module: ANALYSIS.
% Compiled module: LOADCT.
% LOADCT: Loading table BLUE/WHITE
GDL> exit
```

↓ choose a plot from the list below exit ↓

< 00016.svg > play rate: 1/0.8 sec

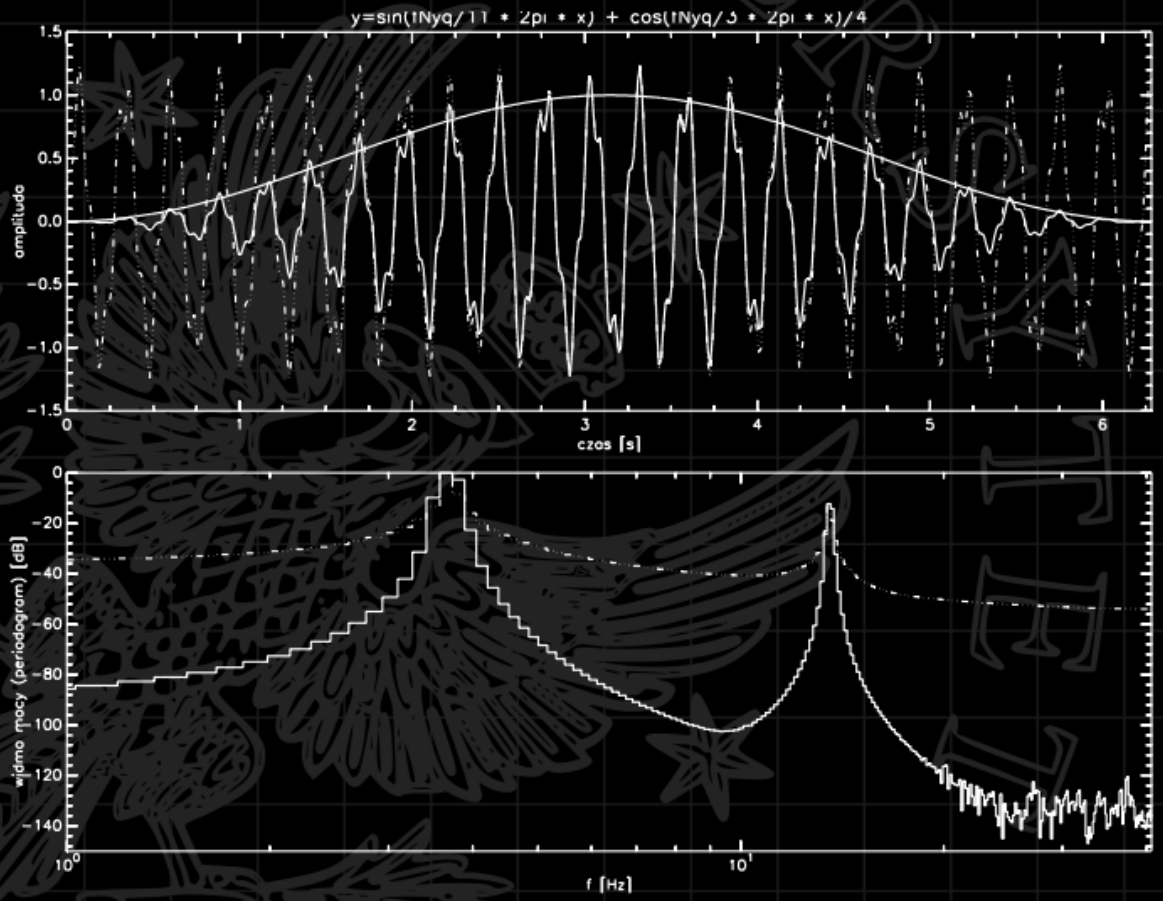



GDL 0
GDL 32
GDL 34
MOD021KM.A2008066.1
MOD02QKM.A2008066.1
GDL> zad7_1, 'data/
% Compiled module:
% Compiled module:
% Compiled module:
% Compiled module:
% Compiled module:
% Compiled module:
% Compiled module:
% Compiled module:

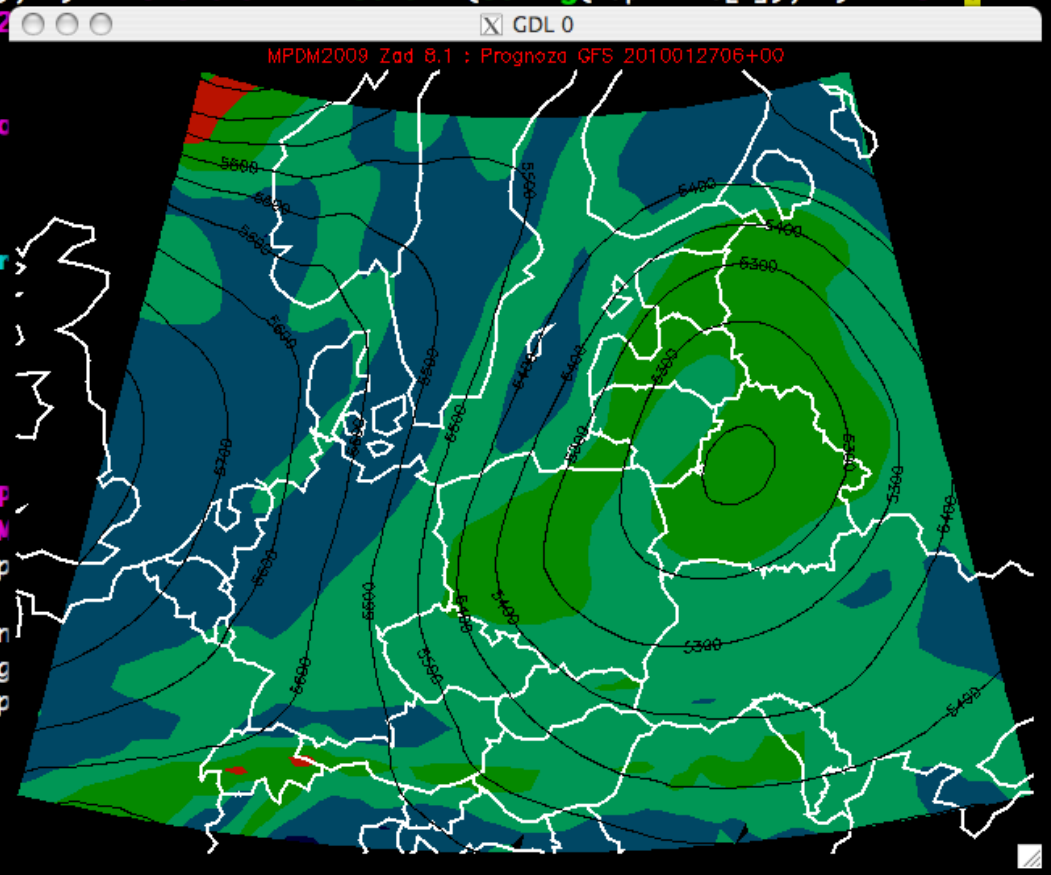

GDL 1
prawdziwa
obserwowana
szacowana
GDL> zad10_2D
% Compiled module: ZAD10_2D.
% Compiled module: MEAN.
% Compiled module: LOADCT.
% LOADCT: Loading table RED TEMPERATURE
GDL>
Default
Default
Default
Default
oo$ gdl
GDL 2
Przyklad kroku czasowego (50)
-6
-8
0
50
100
150
200

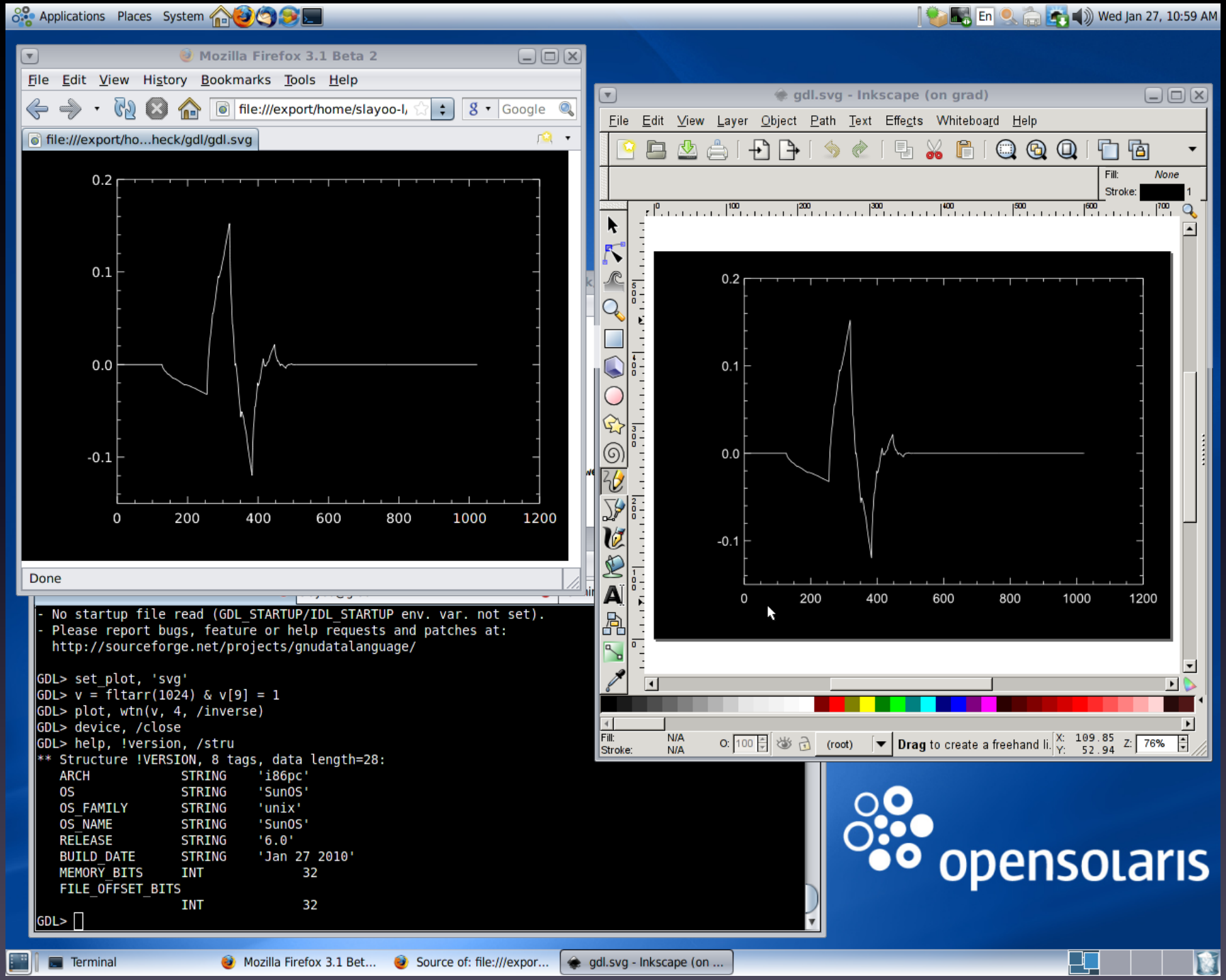
hann_demo.pro

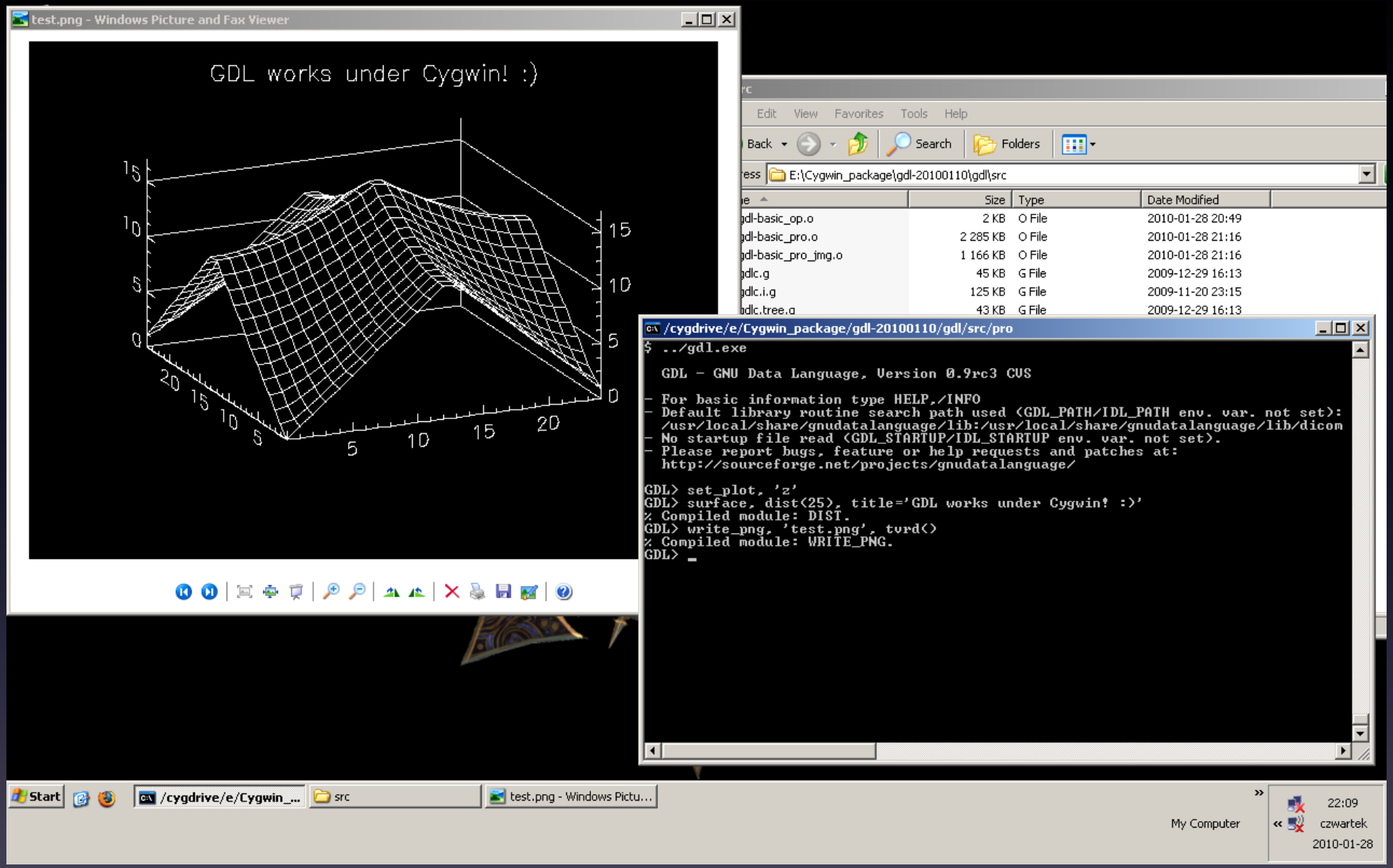
```
1 pro periodogram, czas, sgnl, _extra = ex
2   n = n_elements(czas)
3   dt = czas[1]-czas[0]
4   f = findgen(n/2+1) / n / dt
5   amp2 = abs((fft(sgnl))[0:n/2])^2
6   amp2[1:n/2-1] *= 2 ; uwzgl. częstości ujemnych
7   moc_dB = 10*alog10(amp2/max(amp2))
8   ; norm. do max więc bez norm. do sumy wag w oknie
9   oplot, f, moc_dB, psym=10, _extra = ex
10 end
11
12 pro hann_demo
13   n = 512 & dt = 1./(n-1) * 2 * !PI
14   nyquist = .5 / dt
15
16   czas = findgen(n) * dt
17   sgnl = sin(nyquist/11 * 2*!PI * czas) + $
18          cos(nyquist/3  * 2*!PI * czas) / 4
19   hann = hanning(n)
20
21   !P.MULTI = [0,1,2] & !X.STYLE = 1
22
23   plot,  linestyle=4, czas, sgnl, $
24     xtitle="czas [s]", ytitle="amplituda", $
25     title="y=sin(fNyq/11 * 2pi * x) +" + $
26     " cos(fNyq/3 * 2pi * x)/4"
27   oplot, linestyle=0, czas, sgnl*hann
28   oplot, czas, hann
29
30   plot, [0, nyquist], [0, -150], /nodata, $
31     xtitle="f [Hz]", /xlog, $
32     ytitle="widmo mocy (periodogram) [dB]"
33   periodogram, linestyle=4, czas, sgnl
34   periodogram, linestyle=0, czas, sgnl * hann
35 end
```

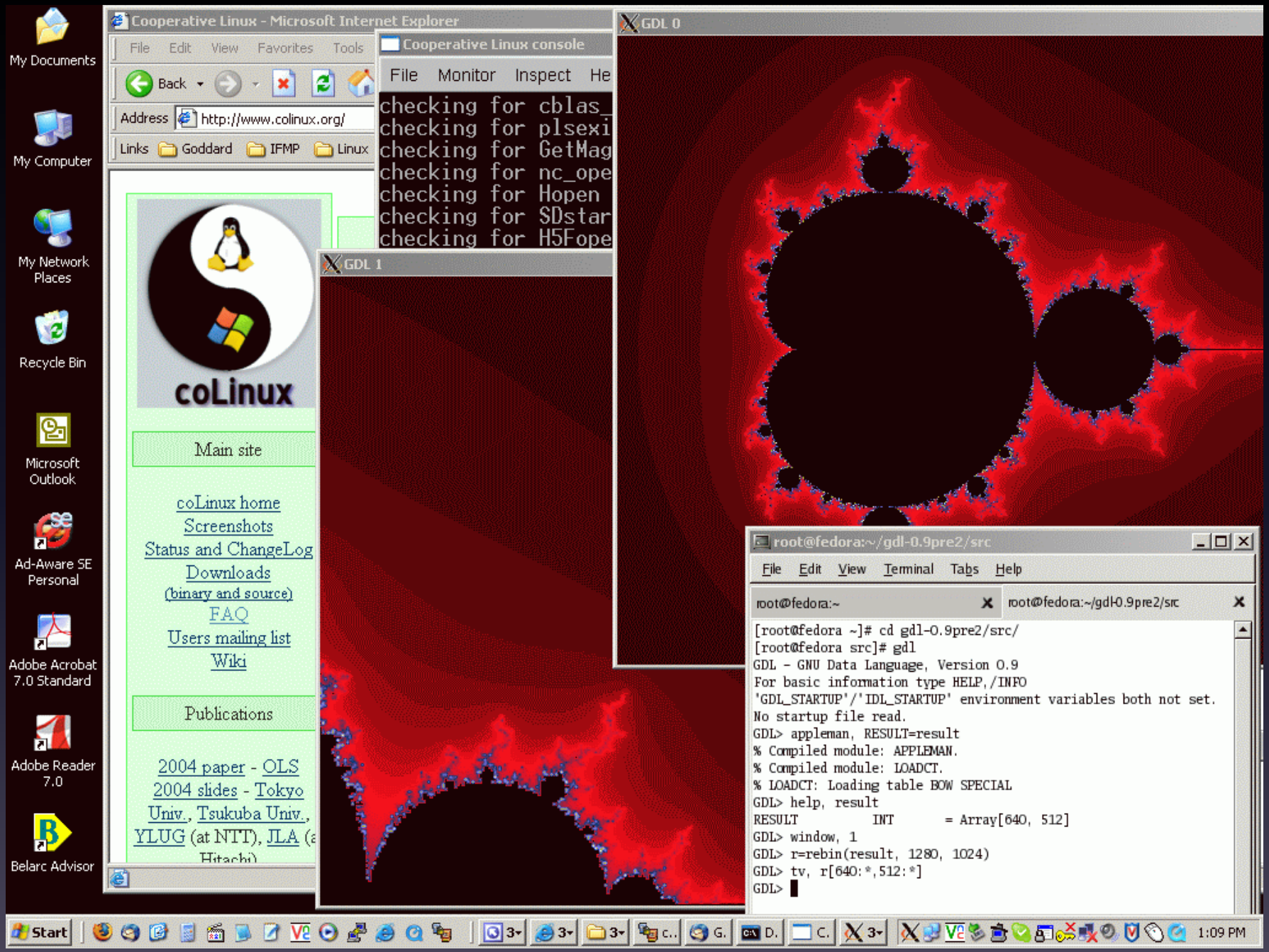


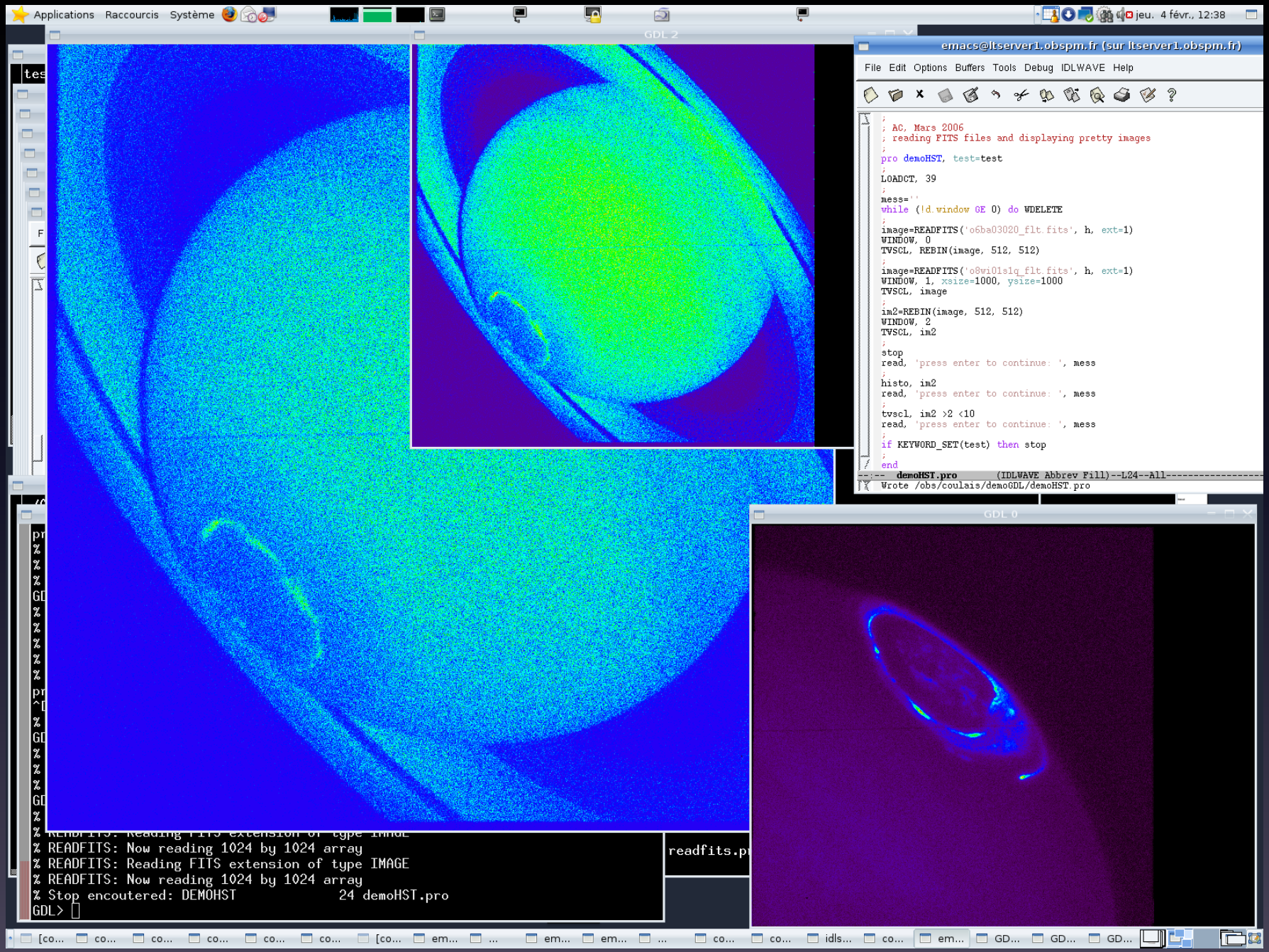
```
22    cmd = "wget --continue --output-document=" + plik $
23      + " 'http://nomads.ncep.noaa.gov/cgi-bin/filter_gfs_hd.pl?file=gfs.t06z.mastergrb2f" + hh_str $
24      + "&lev_500_mb=on&var_HGT=on&var_UGRD=on&var_VGRD=on&var_ABSV=on&subregion=&dir=%2Fgfs." + data + "%2Fmaster"
25      + "&bottomlat=" + strtrim(string(maplimit[0]), 2) + "&leftlon=" + strtrim(string(maplimit[1]), 2) + "&" $
26      + "&toplat=" + strtrim(string(maplimit[2]), 2
27    spawn, cmd, output, exit_status=status
28    if status ne 0 then begin
29      message, /continue, 'pobranie pliku nie powio
30      continue
31    endif
32
33    ; pobranie danych z plików GRIB (w pierwszym kr
34    grib_f = gribapi_open_file(plik)
35    n_msgs = gribapi_count_in_file(grib_f)
36    for m = 0, n_msgs - 1 do begin
37      grib_m = gribapi_new_from_file(grib_f)
38      if h eq h_prwsz and m eq 0 then begin
39        gribapi_get_data, grib_m, lats, lons, tmp
40        gribapi_get, grib_m, 'numberOfPointsAlongAP
41        gribapi_get, grib_m, 'numberOfPointsAlongAN
42        n_stps = 1 + (h_osttn - h_prwsz) / h_perstp
43        lons = (temporary(lons))[indgen(n_lons)]
44        lats = (temporary(lats))[indgen(n_lats) * n
45        vals = fltarr(n_lons, n_lats, n_stps, n_msg
46      endif else gribapi_get, grib_m, 'values', tmp
47      vals[*, *, h/h_perstp, m] = temporary(tmp)
48      gribapi_release, grib_m
49    endfor ; m
50    gribapi_close_file, grib_f
```




Applications Places System
Wed Jan 27, 10:59 AM
Mozilla Firefox 3.1 Beta 2
File Edit View History Bookmarks Tools Help
file:///export/home/slayoo-l
file:///export/ho...heck/gdl/gdl.svg
Done
gdl.svg - Inkscape (on grad)
File Edit View Layer Object Path Text Effects Whiteboard Help
Fill: None
Stroke:
Drag to create a freehand li.
- No startup file read (GDL_STARTUP/IDL_STARTUP env. var. not set).
- Please report bugs, feature or help requests and patches at:
http://sourceforge.net/projects/gnudatalanguage/
GDL> set_plot, 'svg'
GDL> v = fltarr(1024) & v[9] = 1
GDL> plot, wtn(v, 4, /inverse)
GDL> device, /close
GDL> help, !version, /stru
** Structure !VERSION, 8 tags, data length=28:
ARCH STRING 'i86pc'
OS STRING 'SunOS'
OS_FAMILY STRING 'unix'
OS_NAME STRING 'SunOS'
RELEASE STRING '6.0'
BUILD_DATE STRING 'Jan 27 2010'
MEMORY_BITS INT 32
FILE_OFFSET_BITS
INT 32
GDL>
Terminal
Mozilla Firefox 3.1 Bet...
Source of: file:///expor...
gdl.svg - Inkscape (on ...
opensolaris


test.png - Windows Picture and Fax Viewer
GDL works under Cygwin! :)
E:\Cygwin_package\gdl-20100110\gdl\src
/cygdrive/e/Cygwin_package/gdl-20100110/gdl/src/pro
$ ../gdl.exe
GDL - GNU Data Language, Version 0.9rc3 CVS
- For basic information type HELP,/INFO
- Default library routine search path used (GDL_PATH/IDL_PATH env. var. not set):
/usr/local/share/gnudatalanguage/lib:/usr/local/share/gnudatalanguage/lib/dicom
- No startup file read (GDL_STARTUP/IDL_STARTUP env. var. not set).
- Please report bugs, feature or help requests and patches at:
http://sourceforge.net/projects/gnudatalanguage/
GDL> set_plot, 'z'
GDL> surface, dist(25), title='GDL works under Cygwin! :)'
% Compiled module: DIST.
GDL> write_png, 'test.png', tvrd()
% Compiled module: WRITE_PNG.
GDL>
Start
My Computer
22:09
czwartek
2010-01-28


Cooperative Linux - Microsoft Internet Explorer
File Edit View Favorites Tools
Back
Address http://www.colinux.org/
Links Goddard IFMP Linux
coLinux
Main site
coLinux home
Screenshots
Status and ChangeLog
Downloads
(binary and source)
FAQ
Users mailing list
Wiki
Publications
2004 paper - OLS
2004 slides - Tokyo Univ., Tsukuba Univ., YLUG (at NTT), JLA (at Hitachi)
Cooperative Linux console
File Monitor Inspect He
checking for cblas_
checking for plsexi
checking for GetMag
checking for nc_ope
checking for Hopen
checking for SDstar
checking for H5Fope
GDL 0
GDL 1
root@fedora:~/gdl-0.9pre2/src
File Edit View Terminal Tabs Help
root@fedora:~
root@fedora:~/gdl-0.9pre2/src
[root@fedora ~]# cd gdl-0.9pre2/src/
[root@fedora src]# gdl
GDL - GNU Data Language, Version 0.9
For basic information type HELP,/INFO
'GDL_STARTUP'/'IDL_STARTUP' environment variables both not set.
No startup file read.
GDL> appleman, RESULT=result
% Compiled module: APPLEMAN.
% Compiled module: LOADCT.
% LOADCT: Loading table BOW SPECIAL
GDL> help, result
RESULT INT = Array[640, 512]
GDL> window, 1
GDL> r=rebin(result, 1280, 1024)
GDL> tv, r[640:*,512:*]
GDL>
My Documents
My Computer
My Network Places
Recycle Bin
Microsoft Outlook
Ad-Aware SE Personal
Adobe Acrobat 7.0 Standard
Adobe Reader 7.0
Belarc Advisor
Start
1:09 PM


Applications Raccourcis Système
jeu. 4 févr., 12:38
GDL 2
emacs@ltserver1.obspm.fr (sur ltserver1.obspm.fr)
File Edit Options Buffers Tools Debug IDLWAVE Help
; AC, Mars 2006
; reading FITS files and displaying pretty images
pro demoHST, test=test
LOADCT, 39
mess=''
while (!d.window GE 0) do WDELETE
image=READFITS('o6ba03020_flt.fits', h, ext=1)
WINDOW, 0
TVSCL, REBIN(image, 512, 512)
image=READFITS('o8wi01s1q_flt.fits', h, ext=1)
WINDOW, 1, xsize=1000, ysize=1000
TVSCL, image
im2=REBIN(image, 512, 512)
WINDOW, 2
TVSCL, im2
stop
read, 'press enter to continue: ', mess
histo, im2
read, 'press enter to continue: ', mess
tvscl, im2 >2 <10
read, 'press enter to continue: ', mess
if KEYWORD_SET(test) then stop
end
demoHST.pro (IDLWAVE Abbrev Fill)--L24--All
Wrote /obs/coulais/demoGDL/demoHST.pro
% READFITS: Now reading 1024 by 1024 array
% READFITS: Reading FITS extension of type IMAGE
% READFITS: Now reading 1024 by 1024 array
% Stop encoutered: DEMOHST 24 demoHST.pro
GDL>
readfits.pr
GDL 0

Plik Edycja Widok Terminal Karty Pomoc

File Edit Options Buffers Tools Debug IDLWAVE Help

```
 ncdf_varget, nc, 'range', r
 ncdf_varget, nc, 'average_time', avt
 ncdf_varget, nc, 'beta_raw', snl
 ncdf_varget, nc, 'temp_int', temp_int
 ncdf_varget, nc, 'temp_ext', temp_ext
 ncdf_varget, nc, 'temp_det', temp_det
 ncdf_varget, nc2, 'time', t2
 ncdf_varget, nc2, 'range', r2
 ncdf_varget, nc2, 'average_time', avt2
 ncdf_varget, nc2, 'beta_raw', snl2
 ncdf_varget, nc2, 'temp_int', temp_int2
 ncdf_varget, nc2, 'temp_ext', temp_ext2
 ncdf_varget, nc2, 'temp_det', temp_det2

 ; Przygotowanie danych
                                ; lacze dane z dwoch sasiednich
                                ; plikow zadajac warunek, zeby
                                ; pierwszy bin byl o godzin 00 UTC a
                                ; ostatni 00 UTC  24 godziny pozniej
 t_0 = day1904(mm,dd,yyyy,0,0,0) ; godzina zapisana w formacie sekund d
 t_1 = day1904(mm2,dd2,yyyy2,0,0,0)
 indx_t_0 = where(t gt t_0 and t lt t_0+20)
 indx_t_1 = where(t2 lt t_1 and t2 gt t_1-20)
 ; print, indx_t_0, indx_t_1
 ; help, snl, t, temp_int, snl2, t2, temp_int2
 snl = [[snl(*,indx_t_0(0):n_elements(t)-1)],[snl2(*,0:indx_t_1(n_eleme

 ; Tworze tablice odpowiadajace regionowi powyzej wyznaczonej granicy
 r_brdr = 5000
 snl_up = snl(r_brdr/15:n_elements(r)-1,*)
 r_up = r(r_brdr/15:n_elements(r)-1)

 ; snl(*,0:500) = 1000
 ; for i=0,n_elements(r_up)-1 do begin
 ;         snl(i,*) = (15*i)^2*snl(i,*)
 ; end
```

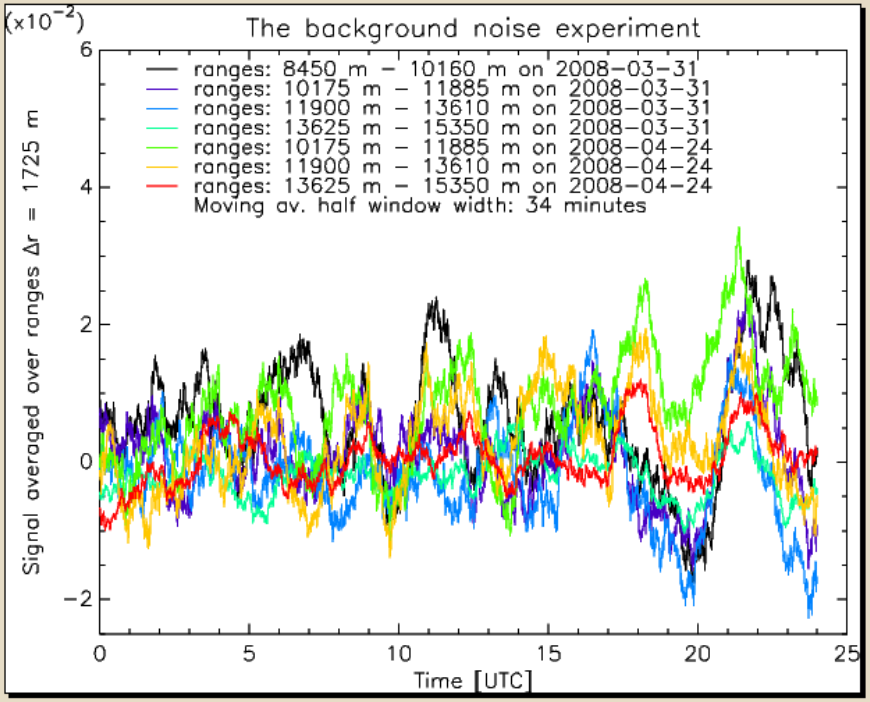
--2-:---F1 **normalis.pro** (IDLWAVE Abbrev Fill)--L137--C0--31%---------

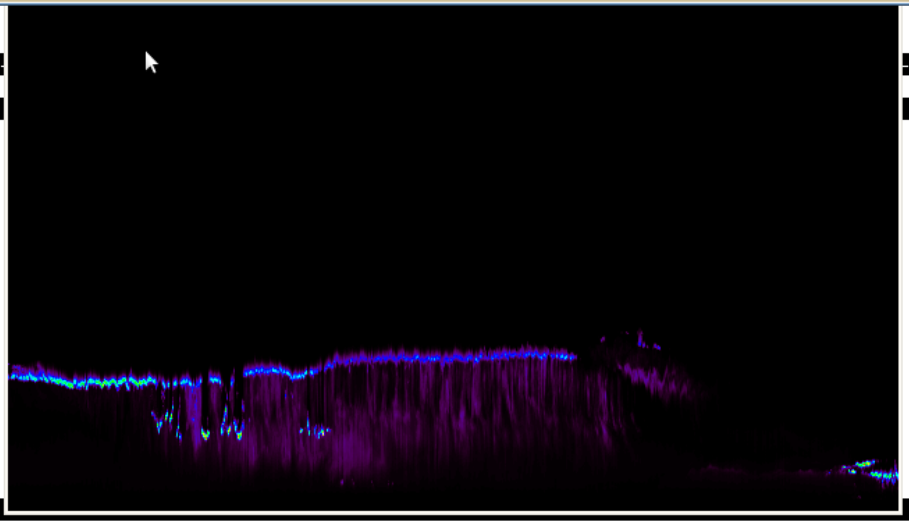
```
  0 bash
% Compiled module: CMSV_WREC.
% Compiled module: CMSV_RREC.
% Compiled module: CMSV_WRAW.
% Compiled module: CMSV_WVTYPE.
% Compiled module: CMSV_WDATA.
SNL             FLOAT     = Array[1024, 8520]
GDL> normalis, 20080424
      153.313
% LOADCT: Loading table Rainbow + white
  1.57355e-06
SNL             FLOAT     = Array[1024, 8516]
GDL> normalis, 20100304
      25270.4
% LOADCT: Loading table Rainbow + white
  5.29369e-06
SNL             FLOAT     = Array[1024, 2871]
GDL> normalis, 20080424
  2 bash
```

PLplot Graph

Plik Edycja Widok Przejdź Pomoc

Poprzednia Następna 1 z 1 100%





Komputer | sirocco | GDL 13 | GDL 2 | PLplot Graph | czw 15 kwi, 15:23

About GDL: how to get GDL

- pre-compiled packages (no trouble, fast, no options, releases only):
  - Fedora: 32/64 bit, up-to-date, even experimental features available
  - Debian: ca. 15 architectures!, bit outdated (2009-09)
  - ArchLinux: up-to-date, small feature set
  - Ubuntu: obsolete (2008-04)
  - unofficial up-to-date packages for Ubuntu & Debian by Lea Noreskal
  - HMUG (OSX): up-to-date
  - hpc.sourceforge.net: obsolete (2008-04)

About GDL: how to get GDL

- pre-compiled packages (no trouble, fast, no options, releases only):
  - Fedora: 32/64 bit, up-to-date, even experimental features available
  - Debian: ca. 15 architectures!, bit outdated (2009-09)
  - ArchLinux: up-to-date, small feature set
  - Ubuntu: obsolete (2008-04)
  - unofficial up-to-date packages for Ubuntu & Debian by Lea Noreskal
  - HMUG (OSX): up-to-date
  - hpc.sourceforge.net: obsolete (2008-04)
- pre-configured packages (no trouble, some options, releases only):
  - Gentoo: up-to-date, rich feature set
  - Macports: up-to-date, rich feature set (the only one with CMSVLIB!)
  - Fink: up-to-date
  - FreeBSD: bit outdated (2009-09)

About GDL: how to get GDL

- pre-compiled packages (no trouble, fast, no options, releases only):
  - Fedora: 32/64 bit, up-to-date, even experimental features available
  - Debian: ca. 15 architectures!, bit outdated (2009-09)
  - ArchLinux: up-to-date, small feature set
  - Ubuntu: obsolete (2008-04)
  - unofficial up-to-date packages for Ubuntu & Debian by Lea Noreskal
  - HMUG (OSX): up-to-date
  - hpc.sourceforge.net: obsolete (2008-04)
- pre-configured packages (no trouble, some options, releases only):
  - Gentoo: up-to-date, rich feature set
  - Macports: up-to-date, rich feature set (the only one with CMSVLIB!)
  - Fink: up-to-date
  - FreeBSD: bit outdated (2009-09)
- source distribution (all options up to you, troubles happen, releases only)

About GDL: how to get GDL

- pre-compiled packages (no trouble, fast, no options, releases only):
  - Fedora: 32/64 bit, up-to-date, even experimental features available
  - Debian: ca. 15 architectures!, bit outdated (2009-09)
  - ArchLinux: up-to-date, small feature set
  - Ubuntu: obsolete (2008-04)
  - unofficial up-to-date packages for Ubuntu & Debian by Lea Noreskal
  - HMUG (OSX): up-to-date
  - hpc.sourceforge.net: obsolete (2008-04)
- pre-configured packages (no trouble, some options, releases only):
  - Gentoo: up-to-date, rich feature set
  - Macports: up-to-date, rich feature set (the only one with CMSVLIB!)
  - Fink: up-to-date
  - FreeBSD: bit outdated (2009-09)
- source distribution (all options up to you, troubles happen, releases only)
- CVS (all options up to you, newest features/fixes, expect troubles)

About GDL: the Macports GDL package

```
eyrie:~ slayoo$ port info gnudatalanguage
gnudatalanguage @0.9rc4, Revision 4 (math, science)
Variants:             mpich, openmp, python, universal, wxWidgets

Description:          A free IDL (Interactive Data Language) compatible
                      incremental compiler (ie. runs IDL programs).
Homepage:             http://gnudatalanguage.sourceforge.net/

Build Dependencies:   autoconf, automake, libtool
Library Dependencies: zlib, gsl, ncurses, readline, plplot, netcdf, hdf4,
                      hdf5-18, grib_api, libproj4, ImageMagick, xorg-libX11,
                      udunits2, fftw-3, fftw-3-single, cmsvlib
Platforms:            darwin
License:              GPLv2
Maintainers:          takeshi@macports.org, slayoo@igf.fuw.edu.pl
```

http://www.macports.org/

About GDL: how to help and contribute

- Try it! Let us know what you use GDL for. Or if you don't, why not.
- Report feature/improvement requests and bugs (with example codes).
- Send patches (GPL code) with new features, bugfixes or test routines.
- Port or package GDL for new platforms.
- Report problems to packagers.
- Let others know about GDL!

We need help!

About GDL: how to help and contribute

- Try it! Let us know what you use GDL for. Or if you don't, why not.
- Report feature/improvement requests and bugs (with example codes).
- Send patches (GPL code) with new features, bugfixes or test routines.
- Port or package GDL for new platforms.
- Report problems to packagers.
- Let others know about GDL!

We need help!

- What is really important now is that GDL is User-wish driven project.
- We cannot check all the cases: please report!
- We don't know all keywords: please report if missing or misbehaving.
- We don't know or haven't checked all (sometime crazy) interactions between keywords: please report problems (IDL Documentation is itself not always without ambiguities).

Hands-on: let's start

```
$ gdl
GDL> print, 'Hello world!'
Hello world!
GDL> $ echo "Hello world!"
Hello world!
GDL> hello
% Compiled module: HELLO.
Hello world!
GDL> hello
Hello world!
GDL> exit
$ gdl -quiet -e hello
% Compiled module: HELLO.
Hello world!
```

hello.pro:

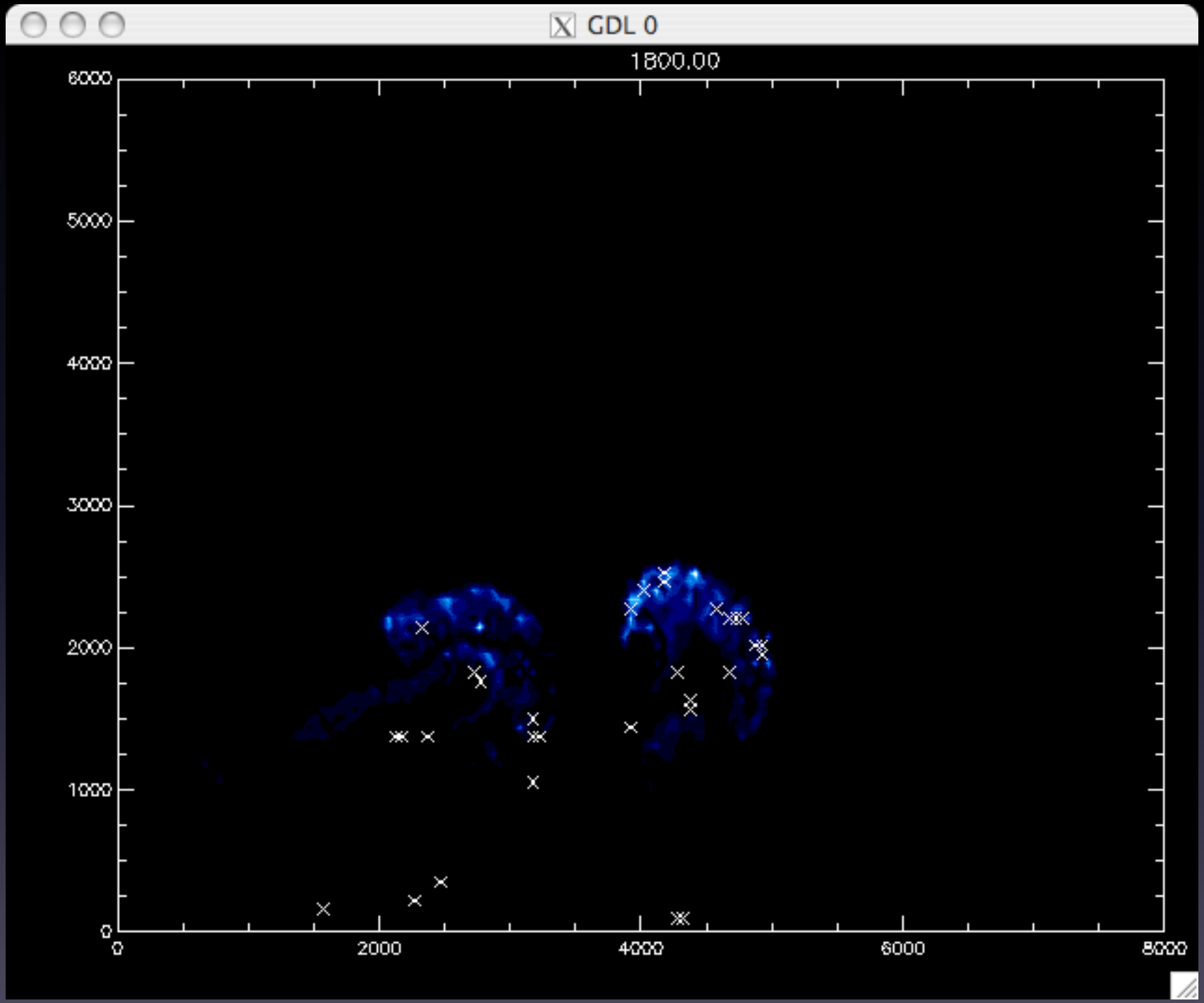
```
1 pro hello
2   print, 'Hello world!'
3 end
```

Hands-on: syntax basics

- assignement & function call with one positional argument:
  GDL> a = findgen(11)
- procedure call & variable size inquiry
  GDL> help, a
- keyword argument & some Fortran heritage...
  GDL> print, a, format='(4G)'
- ... and some relief for C users
  GDL> print, a, format='(%"%4.3g")'
- simplest plot (x-axis: array indices) & math ops. applied to arrays
  GDL> plot, sin(a*!PI/10)^2
- array adressing
  GDL> print, a[0]
  GDL> print, a[4:5]
  GDL> print, a[[9,10]]
  GDL> print, a[a]
  GDL> print, a[indgen(3)]

Hands-on: CReSS-SDM example

```
1 pro cressanim, ncfile
2   nc = ncdf_open(ncfile)                                          ; opening the netCDF file
3   ncdf_varget, nc, 'X', x & nx = n_elements(x)                    ; \
4   ncdf_varget, nc, 'Z', z & nz = n_elements(z)                    ;  | grid-box->x/z ; time-step->time
5   ncdf_varget, nc, 'T', t & nt = n_elements(t)                    ; /
6
7   loadct, 1                                                       ; BLUE/WHITE colour palette
8   qclev = findgen(11) / 1000.                                     ; [0, .001, ... .01] kg/kg
9
10  for i = 0, nt - 1 do begin
11    ncdf_varget, nc, 'qc', qc, offset=[0,0,0,i], count=[nx,1,nz,1]
12    contour, reform(qc), x, z, /fill, levels=qclev, title=t[i] ; filled-contour plots of cloud water
13
14    ncdf_varget, nc, 'qr', qr, o=[0,0,0,i], c=[nx,1,nz,1]       ; any unambiguous keyword name is OK
15    wh = where(qr gt 0, cnt)                                    ; returns "1d indices"
16    if cnt gt 0 then begin
17      ai = array_indices(qr, wh)                                ; converting indices from 1d to 2d
18      plots, x[ai[0,*]], z[ai[2,*]], psym=7                     ; rain water plotted with "X" symbols
19    endif
20
21    wait, 1                                                     ; wating one seconds before next frame
22  endfor
23 end
```


GDL 0
1800.00
6000
5000
4000
3000
2000
1000
0
0
2000
4000
6000
8000

NCDF_OPEN & NCDF_VARGET

NCDF_OPEN() opens a netCDF file

returns: a "handle" to the file (long integer)
arguments: filename
keywords: /nowrite, /write

NCDF_VARGET retrieves data from an open netCDF file

arguments: handle, variable id or name, output variable
keywords: count=array, offset=array, stride=array

LOADCT, CONTOUR & PLOTS

LOADCT loads a color palette

argument: palette id (0...40)

CONTOUR plots data with contours

arguments: values array, [x array, y array]
keywords: /fill, levels=array, title=string, ...

PLOTS plots data with symbols or lines

arguments: x array, y array
keywords: psym=integer, ...

WHERE, ARRAY_INDICES & REFORM

WHERE() returns the (1d) indices of array elems that evaluate to true

arguments: array, output var with count of "true" elems
keywords: ...

ARRAY_INDICES() converts 1-d indices to n-d indices

returns: array with n-d indices for given 1-d indices
arguments: n-d array, 1-d indices
keywords: ...

REFORM() changes the dimensions of an array

returns: the modified array (copy or not)
arguments: array, [new dimenions]
keywords: /overwrite

.COMPILE, STOP & .CONTINUE

.COMPILE triggers recompilation of a file/routine (interpreter directive)

STOP stops execution allowing interactive debuging (to be inserted in the code)

.CONTINUE continues execution after a STOP (interpreter directive)

ご清聴ありがとうございました。