

# データ解析と可視化の共通基盤を 求めて -- 電脳Rubyの理念と実践

堀之内武(北大地球環境)

# 電脳davisプロジェクト

- data analysis and visualization
- 地球流体電脳倶楽部のソフト開発集積活動の一。
- この言葉を使いだしたのは90年代末頃だったような。
  - それ以前からDCL: Fortran用グラフィックライブラリ (GKS) by 塩谷雅人他

99年: JST 計算科学技術活用型特定研究開発推進事業  
「地球惑星流体現象を念頭においた多次元数値データの構造化」(林祥介代表)

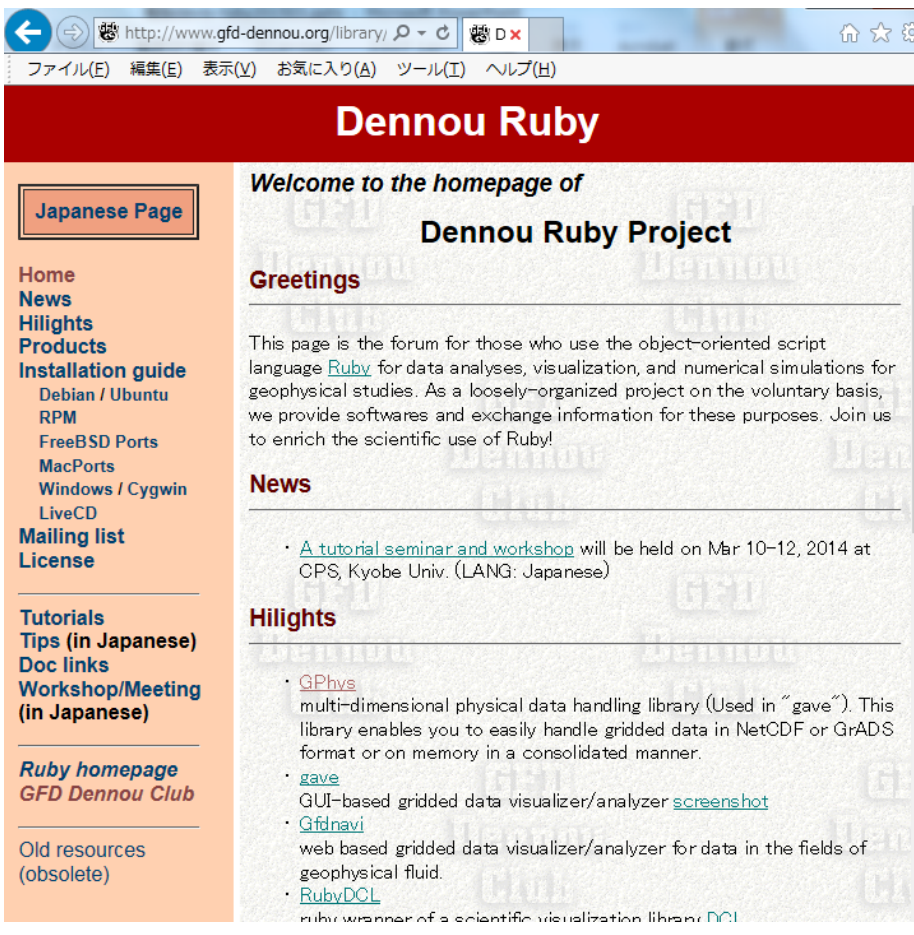
- 現在のdavis活動の基礎構築に貢献
- 新しいGtoolに向けたデータ構造の設計, 試作  
→ Gtool4 NetCDF規約(メタデータ)  
その後の発展: Gtool5 Fortran90/95 IOライブラリ(電脳数値モデル群の足回り)
- オブジェクト指向スクリプト言語(Ruby/Python)の検討と試作 → RubyDCL  
その後の展開: 電脳Rubyソフト群(本日の話題)

# なぜ Ruby？

- 型なし、スクリプト言語（インタープリター）  
⇒ **素早くプログラムが開発できる**
- 洗練され**使いやすいオブジェクト指向言語** ⇒ 開発・保守効率がよく、汎用なソフトを作り易い ⇒ **コミュニティでツールを共有**
- 対話的に利用可能 ⇒ 試行錯誤に良い
- **拡張性が高い** ⇒ CやFortranのライブラリーの有効利用
- 増え続けるライブラリー（ネットワーク関連 / GUI / データベース等々） ⇒ 高度なサービスを実現しやすい。
- 文字処理が容易（データ解析中に文字処理が必要になることは多い）
- ゴミ集め、例外処理等の近代的支援機能あり

# 電脳Ruby「プロジェクト」

- 地球流体研究のためのRuby用ライブラリを開発
- 成果はオープンソースで公開(BSD 2 clauseライセンス)
  - 電脳サーバーで(電脳davisサーバーに「小物置き場も」)



The screenshot shows the homepage of the Dennou Ruby Project. The browser address bar displays <http://www.gfd-dennou.org/library/>. The page features a red header with the text "Dennou Ruby". Below the header, there is a navigation menu with links for "Home", "News", "Highlights", "Products", "Installation guide", "Mailing list", and "License". The main content area includes a "Welcome to the homepage of Dennou Ruby Project" message, a "Greetings" section with a paragraph about the project's purpose, a "News" section with a link to a tutorial seminar, and a "Highlights" section listing various libraries like GPhys, gave, Gfdnavi, and RubyDCL.



The screenshot shows the GFD Ruby object storage page. The browser address bar displays <https://davis.gfd-dennou.org/rut/>. The page has a blue header with the text "GFD電脳Ruby小物置き場". Below the header, there is a search bar and a "検索" button. The main content area includes a "Rubyでデータ解析や数値計算、可視化などをしていると、自作プログラムについて、「もしかしら他の人の役にたつかも」と思うことも多いでしょう。この小物置き場は、そんなプログラムをアップロードしてもらおう場所です。もちろん大物も歓迎!" message, a "最新の記事" section with a link to "電脳Rubyホームページはこちら", and a "どうやって編集する?" section with a list of instructions for editing.

1. 編集するには **ユーザー登録依頼フォーム** で、個人登録をする必要があります。
2. 登録したユーザーでログインすると、ページ最上段に「新規作成」のボタンが現れるのでそこから新規作成ページに入ります。最初にページ名(タイトル)を決めます。ページ名(タイトル)の先頭を「カテゴリ」という書式にするとこの分類に入ります(シングルクォーテーションは除く)。例:  
`(Library) De launay 三角形メッシュ生成`
3. あとは自由にページ本文を作成してください。書式は `RD` を拡張した `RD+`

# 我々が扱うデータ

- 離散的(自由度有限:計算機上の必然)
- 多次元(時間, 空間, 波数空間...) (incl. 0次元)
  - 座標の存在(一次元以上で)
- 単位や名前があるのが普通
  
- データファイルの形式はいろいろ: NetCDF, GRIB, GrADS, テキスト, etc. etc.
  - 構造いろいろ, 読み方いろいろ。
  - 伝統的アプローチだと, 初めてのデータは読めるようになるだけで一苦勞。

# NetCDF:「自己記述型」データ形式

- 気象(地球流体)業界で広く使われる形式の一。バイナリ。(Ver.3は独自, Ver.4:HDFベース)
- バイナリ構造は隠蔽されていて, APIを通じて名前ベースでアクセス。(Ver.4 APIは後方互換)
- ユーザーズガイドによる規約が広く守られている。
  - 単位や名前を表す属性名
  - 座標を表す変数への辿り方
  - ⇒ 物理量を表す変数名から芋づる式に辿れる:「**自己記述**」
- ユーザーズガイド規約の上に, より詳細なメタデータ規約も: CF, Gtool4,... (多くは互いの親和性大)

# NetCDFファイルの構成例

(テキストダンプツールによる)

NCEP再解析(客観解析)の気温データを収めるNetCDFファイルの「ヘッダ」(メタデータ)の内容。

ダンプツール ncdump で表示後、一部省略。

airが気温の4次元データ: lon, lat, level, timeの関数。

バイナリ構造は隠蔽され、名前ベースでアクセス。

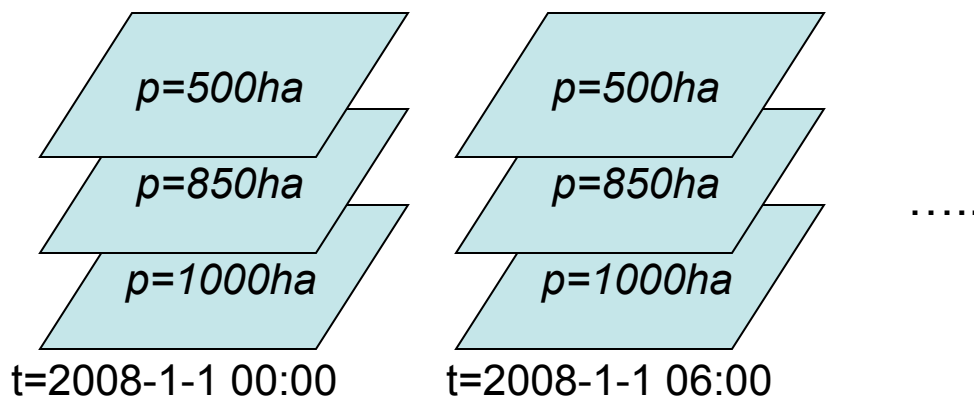
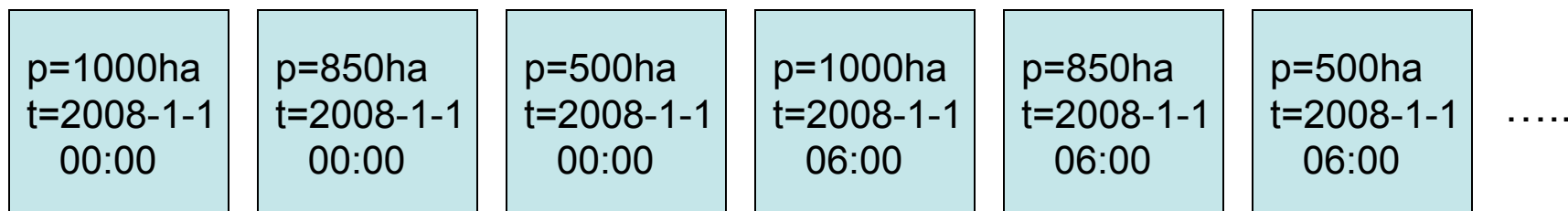
```
$ ncdump -h air.2007.nc
netcdf air.2007 {
dimensions:
    lon = 144 ;
    lat = 73 ;
    level = 17 ;
    time = UNLIMITED ; // (365 currently)
variables:
    float level(level) ;
        level:units = "millibar" ;
        level:long_name = "Level" ;
    float lat(lat) ;
        lat:units = "degrees_north" ;
        lat:long_name = "Latitude" ;
    float lon(lon) ;
        lon:units = "degrees_east" ;
        lon:long_name = "Longitude" ;
    double time(time) ;
        time:units = "hours since 1-1-1 00:00:0.0" ;
        time:long_name = "Time" ;
    short air(time, level, lat, lon) ;
        air:long_name = "mean Daily Air temperature" ;
        air:valid_range = 150.f, 350.f ;
        air:units = "degK" ;
        air:add_offset = 477.66f ;
        air:scale_factor = 0.01f ;
        air:missing_value = 32766s ;
        air:precision = 2s ;

// global attributes:
    :Conventions = "COARDS" ;
    :title = "mean daily NMC reanalysis (2007)" ;
    :history = "created 2007/01/03 by Hoop (netCDF2.3)" ;
    :description = "Data is from NMC initialized reanalysis\n",
        "(4x/day). It consists of most variables interpolated to\n",
        "pressure surfaces from model (sigma) surfaces." ;
```



# 参考：GRIB形式

- 気象予報機関の世界標準
- 水平2次元スライスに関する独立したバイナリデータの集合体。
- ヘッダはバイト(&ビット)単位で各種符号が規定されている。
  - 物理量の種類や時刻や高度はヘッダーに書かれている
  - 時系列や高度方向の次元の認識は解釈系に任されている
- NuSDAS、「国内二進」などの気象庁の形式も同様な構成



# IO(特にI)のアプローチの転換へ

- ドキュメントを読みこなしてデータやメタデータの読み込みを逐一プログラミング



- 自己記述性を活かして、芋づる自動処理
  - 自己記述でない場合も機械処理可能な形でメタデータを補えるはず。
  - ファイル形式が違ってても、中身の論理構造が一緒なら、同じように扱えるはず。
  - さらに、オブジェクト指向なら「同じように」を「同じに」にできるはず：JST課題の基本発想
    - 堀之内は、それまでオブジェクト指向発想でFortran90やIDLを使って試みてたが、OO言語でないが故に「同じように」で止まる壁に当たっていた。(汎用にする手間が大きすぎ)



# BJECT-ORIENTED

SOFTWARE CONSTRUCTION

**SECOND EDITION**

Meyer, 1997  
Object-Oriented  
Software Construction  
(2<sup>nd</sup> Ed)

1<sup>st</sup> Edは1988



CD-ROM INCLUDED



*The Most Comprehensive, Definitive O-O Reference Ever Published*



*An O-O Tour de Force by a Pioneer in the Field*



CD-ROM Includes Complete Hypertext Version of Book AND Object-Oriented Development Environment



Winner  
SOFTWARE  
Development  
PRODUCT  
EXCELLENCE  
AWARD  
1997

**BERTRAND MEYER**

# Rubyによるデータ解析可視化 基盤ライブラリ: GPhys

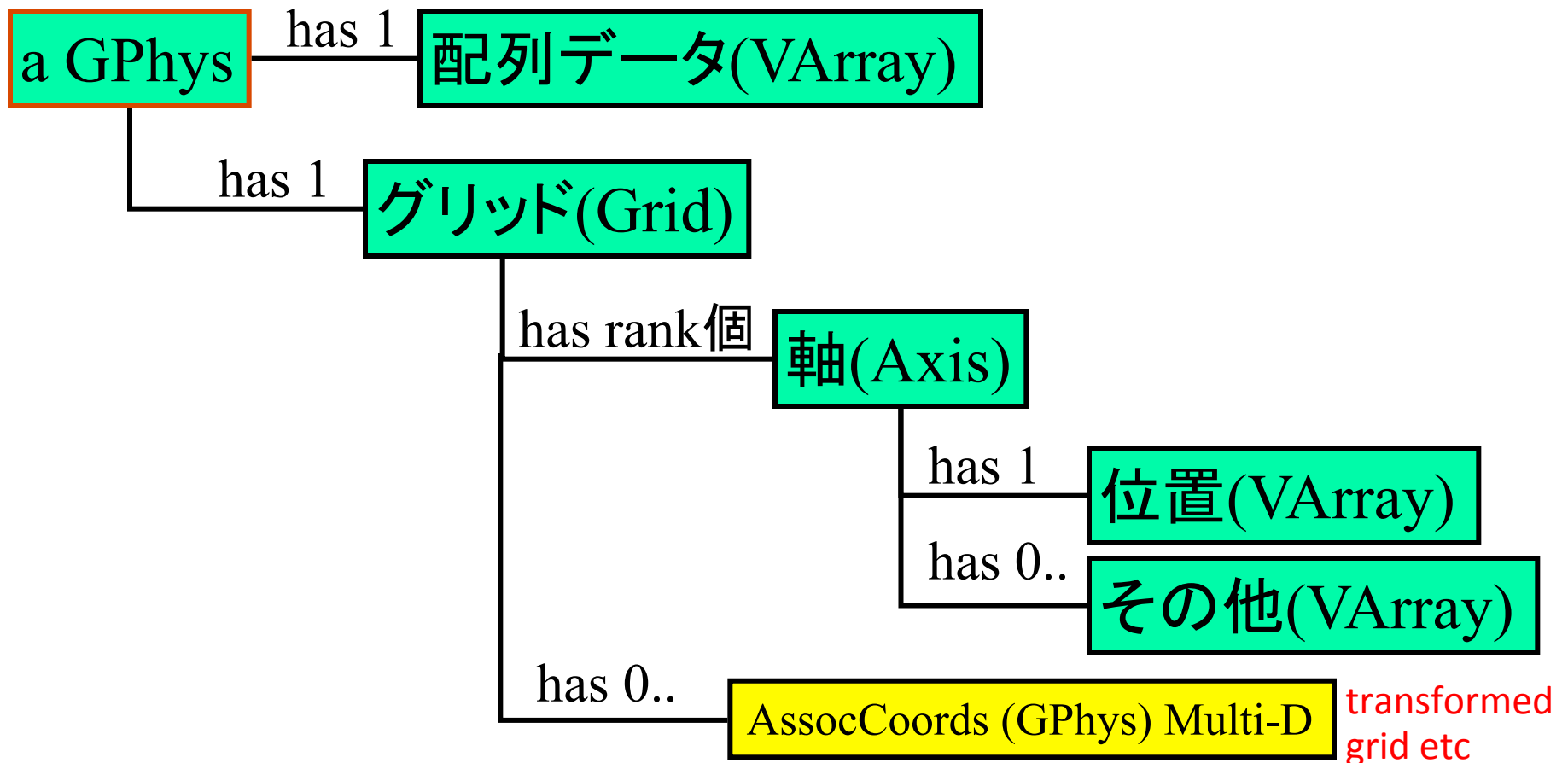
- **GPhys** = **G**ridded **P**hysical quantity
- 任意次元(0,1,...)の座標系における物理量をあらわす「クラス」(型)であり、また、GPhysクラスを頂点とするライブラリ。
- データの物理的実体を隠蔽して(下位の諸ライブラリにそれぞれよろしくやってもらって)、ファイルの形式や次元性によらず、統一的なAPIで操作できる。

## 応用分野

- 流体等、連続空間における物理量に関するデータ解析(incl.可視化)と数値シミュレーション

# GPhysオブジェクトの構成

- グリッド(座標データ)と配列データからなる。
- 数学・算術演算や、積分等の座標に関する演算が行なえる。



# GPhysの重要な構成要素: VArray

- Virtual Arrayの略
- 配列のように振舞うが、データ実体は、Ruby用多次元配列 (NArray)やファイル中の多次元データなど多様な場合を統一的にサポート。(サポート形式: NetCDF, GRIB, GrADS, NuSDAS, HDF5-EOS)
- 他のVArrayのサブセットだったり、複数のVArray合成の場合も。
- NetCDF同様「属性」を持てる。

## パターン1

a VArray

has 1

多次元配列的なデータ(配列またはファイル中の多次元データへのポインタ)

## パターン2

a VArray

has 1..\* (複数)

VArray (別のVArrayのサブセットへのマッピングにもなれる)

# 利用例 (irbによる対話セッション)

```
% irb -r ggraph_startup.rb
```

```
*** MESSAGE (SWDOPN) *** GRPH1 : STARTED / IWS = 1.
```

```
irb(main):001:0> temp = gopen('air.mon.ltm.nc/air')
```

```
=> <GPhys grid=<4D grid <axis pos=<'lon' in 'air.mon.ltm.nc' sfloat[144]>>
```

```
<axis pos=<'lat' in 'air.mon.ltm.nc' sfloat[73]>>
```

```
<axis pos=<'level' in 'air.mon.ltm.nc' sfloat[17]>>
```

```
<axis pos=<'time' in 'air.mon.ltm.nc' float[12]>>>
```

```
data=<'air' in 'air.mon.ltm.nc' sfloat[144, 73, 17, 12]>>
```

```
irb(main):002:0> contour temp.cut('level'=>925)
```

```
*** WARNING (STSWTR) *** WORKST
```

```
=> nil
```

```
irb(main):003:0>
```

演算例:

```
teddy = temp - temp.mean("lon")
```

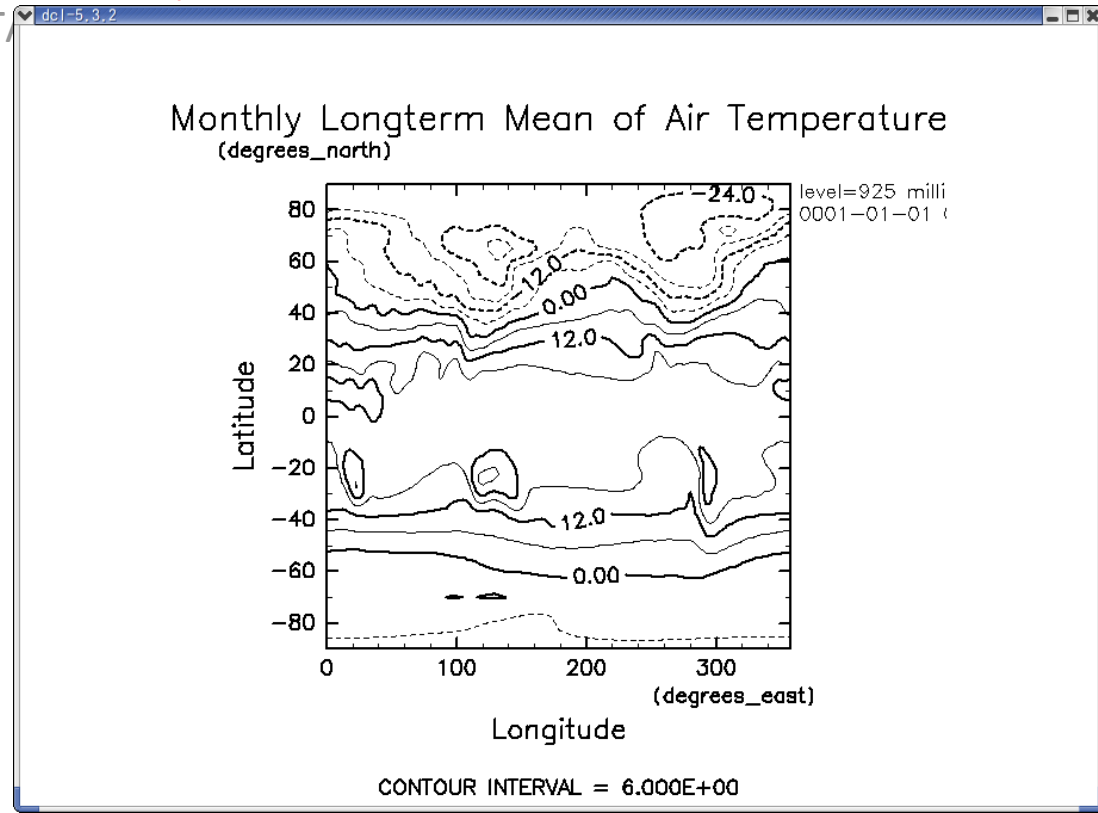
(経度平均を引く。次元の対応は自動判断)

```
tfc = temp.fft(nil, 0, 2)
```

(1次元目(0)および3次元目(2)に関するフーリエ変換. nil→forward. 座標軸は波数軸に。単位も変換。)

スタートアップ用おまじないファイル

コマンドライン入力



# さらに演算例

スタートアップファイルを用いた別表記だと

```
u = gpopen("u.nc/U")
```

```
u = GPhys::IO.open("u.nc", "U")           ← in NetCDF [m/s]
v = GPhys::IO.open("v.ct1", "V")          ← in GrADS [m/s]
uv = u * v                                 ← result on memory [m2s-2]
outfl = NetCDF.create("out.nc")           ← 出力ファイル
GPhys::IO.write(outfl, uv)                ← 出力(座標も一緒に)
```

経度,緯度, 高度, 時刻の4次元データより経度,時間に関するスペクトルを求める。(前処理後処理いろいろ. メソッドチェーンで)

```
sp = u.detrend(3).cos_taper(3).
    fft(false, 0, 3).abs**2
pw = sp.rawspect2powerspect(0, 3).
    spect_zero_centering(0).
    spect_one_sided(3)
```



# GPhysのその他の特徴

- 遅延評価
  - 必要になるまでデータは読まない／コピーしない. 例:  
サブセット切り出しは対応写像だけをバーチャルに
- 大きな実データを無理なく扱う仕組み
  - 処理を自動分割するイテレータのサポートなど
- 演算時に単位を自動更新
- 付属ライブラリ
  - すばやく可視化できる描画ライブラリGGraph
    - クイックルックから論文用の凝った図まで(手数は凝り方に応じて。凝り出すと急に難しくなるギャップなしに)
  - データ解析ライブラリGAnalysis(気象学用などいろいろ)
  - 簡単なデータ解析や描画用の実行コマンド群

# 科研費特定領域 情報爆発IT基盤 2005年発足。翌年からの研究公募開始

info-plosion 情報爆発 - Microsoft Internet Explorer

ファイル(E) 戻る 検索 お気に入り Google G 設定

アドレス(D) http://www.infoplosion.nii.ac.jp/info-plosion/index.php 移動

**info-plosion** H17~H22年度  
情報爆発時代に向けた新しいIT基盤技術の研究  
文部科学省科学研究費補助金「特定領域研究」

English

HOME

プロジェクト概要  
研究組織一覧

- 総括班
- 研究項目 A01
- 研究項目 A02
- 研究項目 A03
- 研究項目 B01
- 支援班

研究共通基盤

InTriggerプラットフォーム利用者登録

公募情報

**情報爆発時代に向けた研究を推進します。**  
**2006-2010**  
**info-plosion**  
情報爆発

2002 3.4~5.4 ExaByte  
2000 2.1~3.2 ExaByte  
World Wide Web (1993)  
Internet, DARPA(1970)  
computing (1950)  
transistor (1947)  
electricity, telephone (1800)  
printing press (1475)  
writing (5000 BCE)  
printing (40,000 BCE)

■ NEWS ■

2008.01.07 本プロジェクトが共催する [International Workshop on Interaction Dynamics, Embodiment, and Implicit Primordial Knowledge Model](#) のサイトが開設されました。参加申込は [京都大学グローバルCOEより「GCOE知識循環社会・国際会議+ワークショップ\(2008.1.15-17\)申込ページ」](#) をクリックしてお申し込みください。

2007.12.05 本プロジェクトのメンバーは、東京工業大学に設置されている、日本一の性能のスーパーコンピュータTSUBAMEを利用するこ

情報爆発時代に向けた新しいIT基盤技術の研究

領域代表者:喜連川 優  
(東京大学・生産技術研究所・教授)

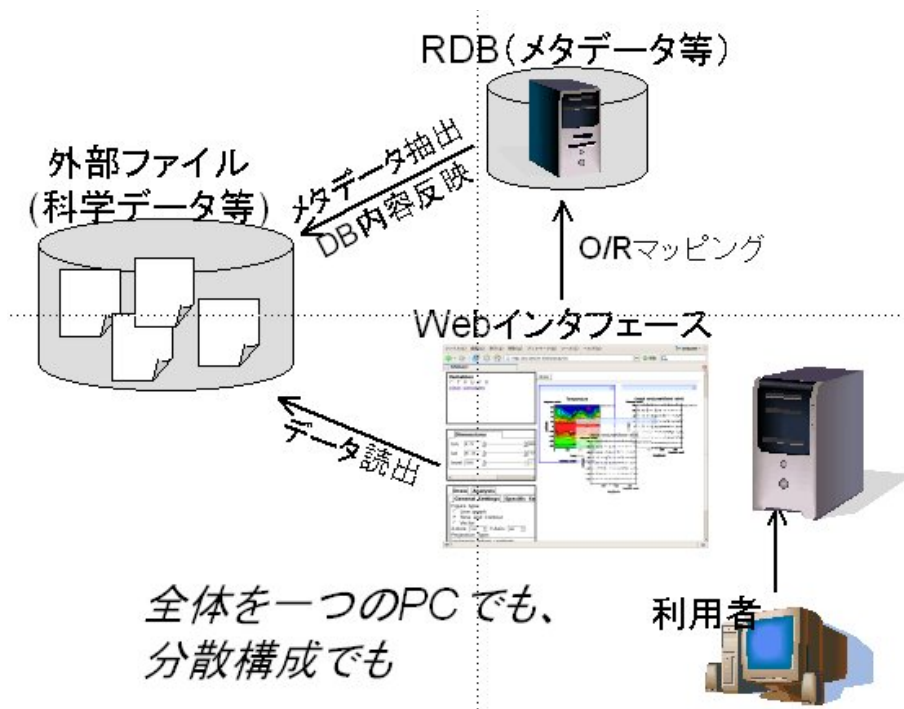
人類によって創出される情報量は2000年に10倍

# GPhysの応用：Webベースのデータサーバ構築ツールGfdnavi（2007-）



- Ruby on Railsベース
- データベース（データ&メタデータ）
- 多彩な検索
- 解析・可視化
- 得られた知見を文書化して登録できる（根拠となる解析可視化の再現スクリプト登録可）
- 現在開発中断中：Ruby on Railsバージョン対応問題

# Gfdnaviの機能



- 指定されたフォルダの下を全スキャン
- 見つかった数値データ、画像をフォルダ構造ごとデータベース化
- 利用者はWebブラウザでアクセス
- 個人のPCで手軽に使えるデータ解析ツールであり、同じものがデータ公開サイト構築にも使える

## 昨年度の飛躍:

# Gfdnavi利用で得られる知見の 文書化 & DB化サポート

- Gfdnaviで行った可視化等をもとに文書を作成  
→ 可視化再現スクリプト、元データへのリンクとともにDB – **応用性大**

**利用例: 共同研究プラットフォーム(共同作業や意思疎通の補助, 文書アーカイブ)、データ公開サイトにおける情報発信(PR)、研究ノートなど多様  
解析内容再現 & 拡張機能 – 検証性の実現**

# まとめ

- オブジェクト指向言語を使うことでファイル形式に依らない統一的な入出力と、データハンドリングを実現
- Rubyを使うことでオープンソースの様々な枠組みが利用可能
  - Webベースのデータサーバー(DB, 解析可視化, 知見文書蓄積)構築ツール Gfdnavi
  - ドキュメンテーション(RDoc), テストフレームワーク, etc.
  - Cでポータブルな拡張→伝統的資源の拡張

# 展望

- GPhys (解析可視化ツール)
  - 着実な発展・増強が望まれる(inclドキュメント類)
- Gfdnavi (データ&知見サーバ)
  - Gfdnavi: 先進的・包括的な実験であった
  - 電脳倶楽部の野望の実現に, 様々なGfdnaviのようなものが役立つのでは?: 構想から知的挑戦
    - 実験整理, 検証できる知見アーカイブ
    - 共通基盤ライブラリプラットフォーム (テスト(incl解析)の実行環境, アーカイブ(DB))
    - インテラクティブな, 探究できる教科書
    - ...
- 余談: *check out.. IRuby*