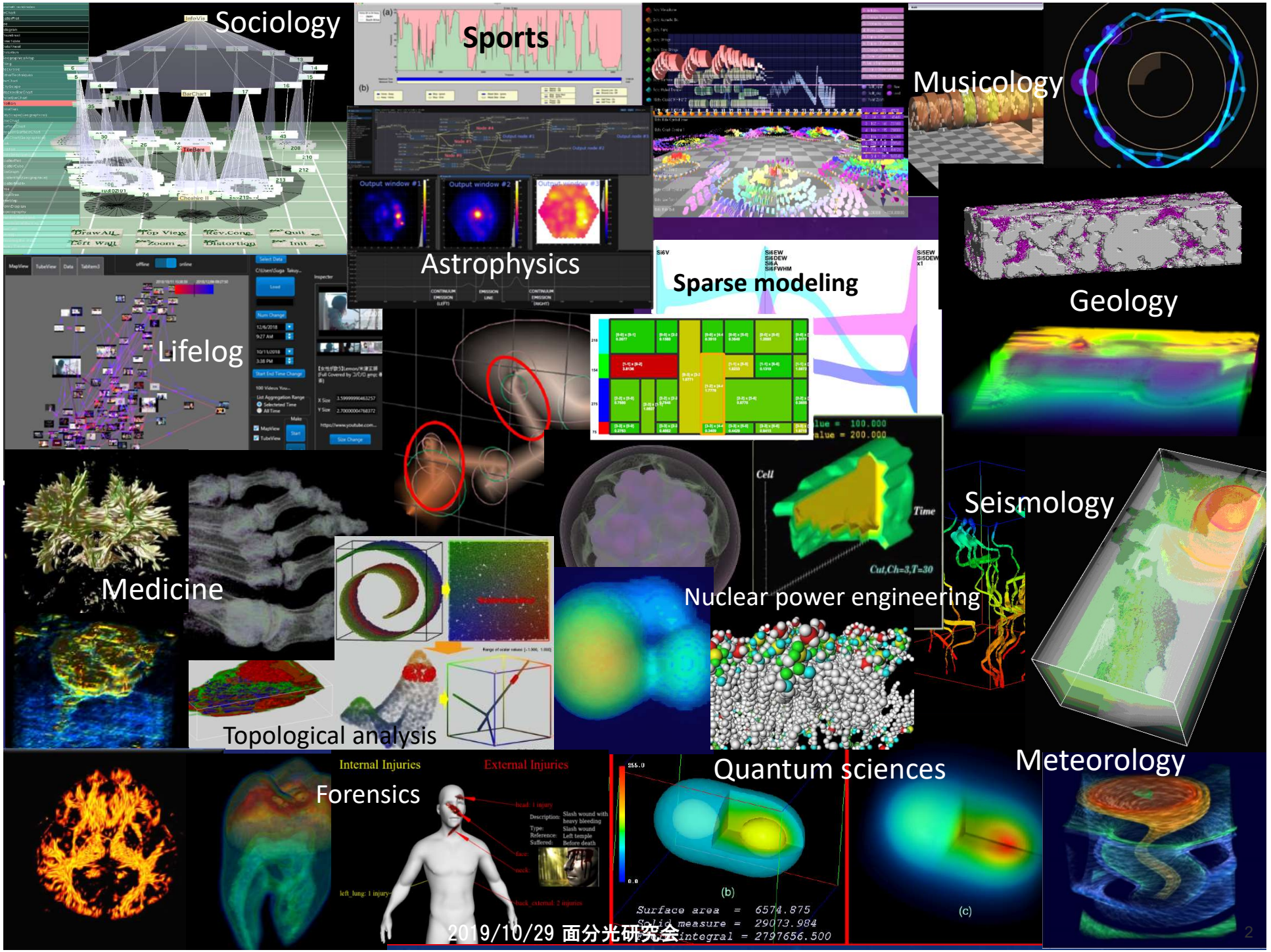
http://aflak.jp

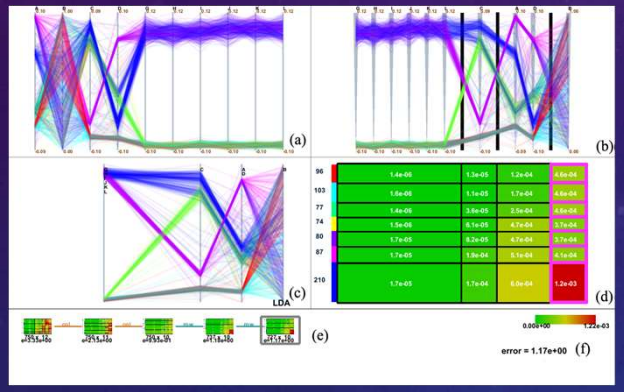
aflak: 面分光データ視覚分析のための ビジュアルプログラミング環境

藤代 一成，Malik Olivier Boussejra, 打木 陸雄
慶應義塾大学理工学部情報工学科

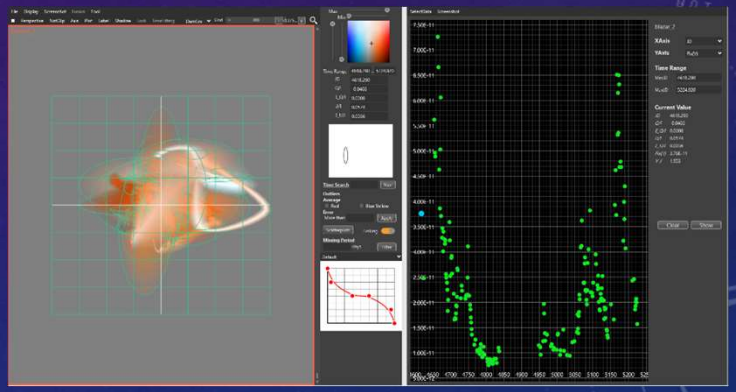

Sociology
Sports
Musicology
Astrophysics
Sparse modeling
Geology
Lifelog
Seismology
Medicine
Nuclear power engineering
Topological analysis
Quantum sciences
Meteorology
Forensics
Internal Injuries
External Injuries
Surface area = 6574.875
Solid measure = 29073.984
integral = 2797656.500

天文学との出会い

- 科研費新学術領域「スパースモデリング」(2013～2018)
  - A02-3:天文学班(代表:本間 希樹 先生)
  - C01-4:可視化班(代表:藤代)
    - 植村　誠 先生(A02-3分担)との共同研究開始



ABC(非対称バイクラスタリング)
Type Ia超新星データの分類



TimeTubes
ブレーザー時系列特徴解析

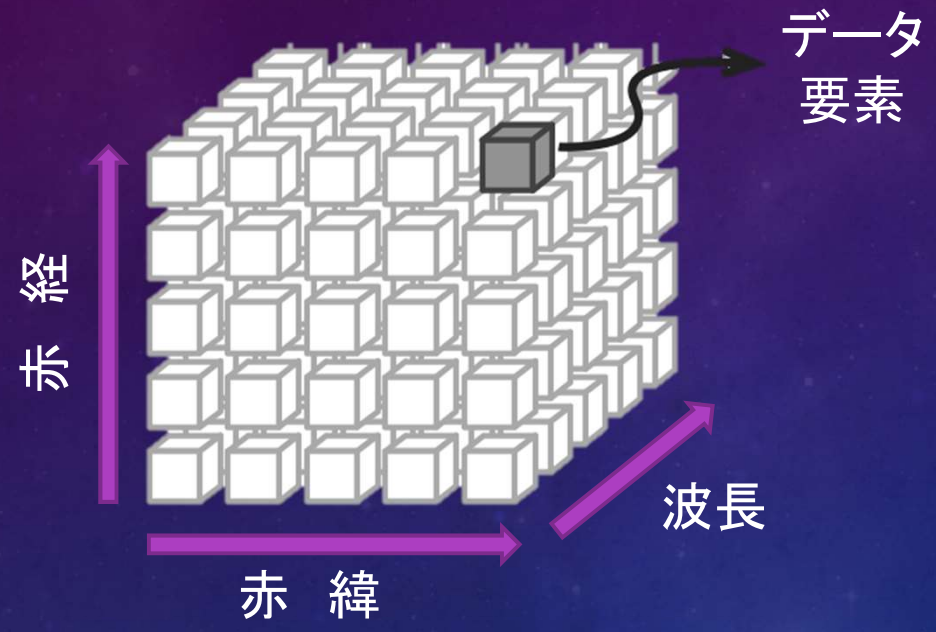
- 松林 和也 先生との共同研究開始(2017)
  - マルチスペクトルデータの視覚解析

発表内容

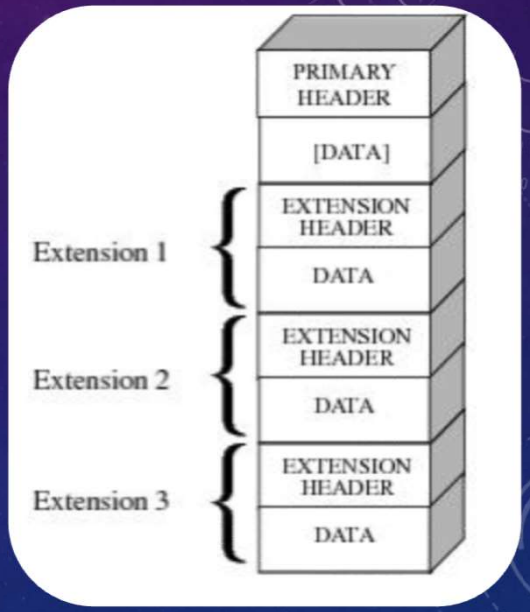
- 藤代：背景と設計指針（15分）
- Malik（訪問研究員）：システム概要（15分）
- 打木（M1）：応用例（20分）
- 藤代：現状と課題（5分）
- 全員：Q&A（5分）

背景と設計指針

天文学マルチスペクトルデータ

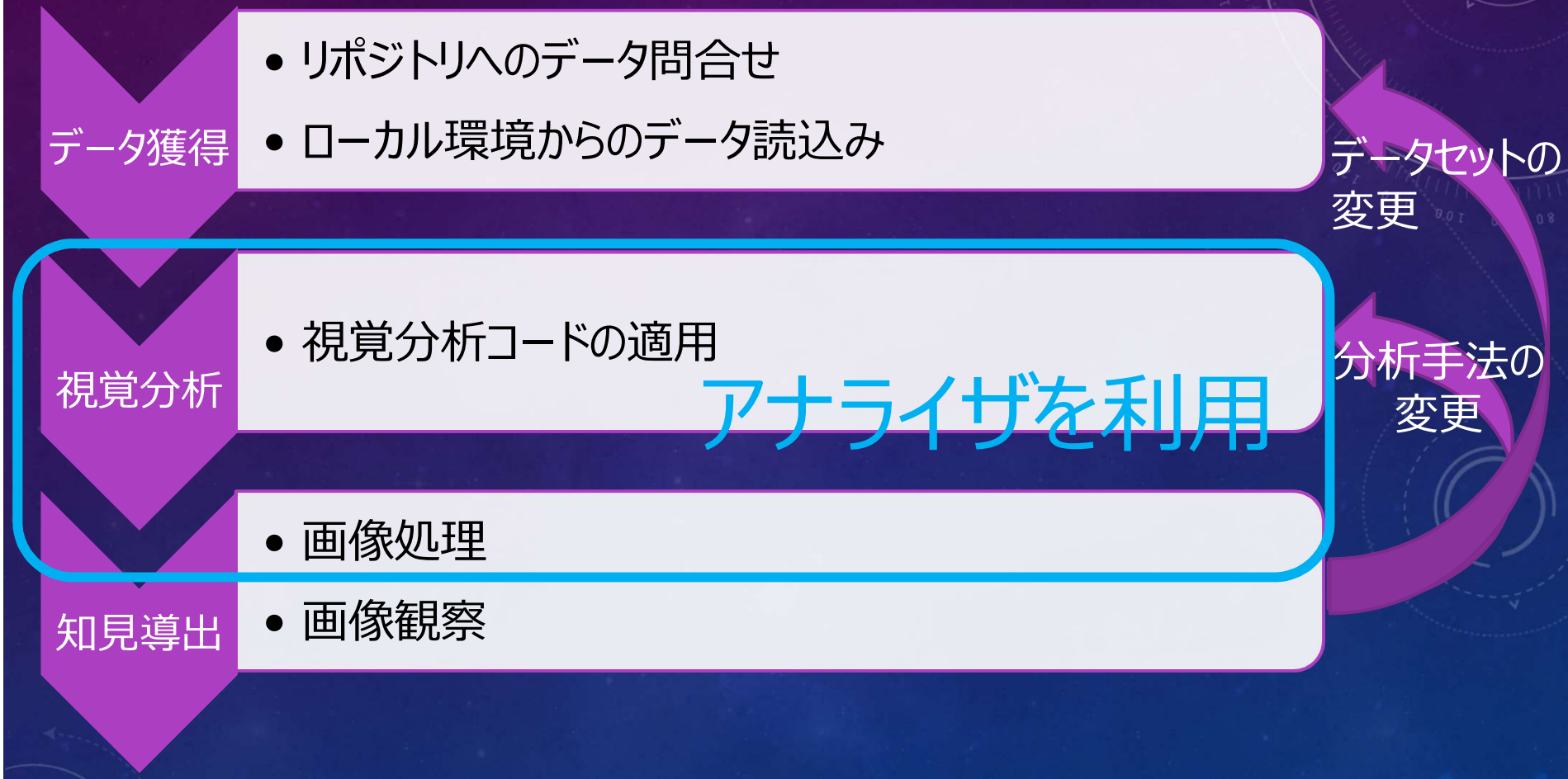


レギュラーボリュームデータ



.fits (Flexible image transport system)

天文学の標準的ワークフロー



天文学データ解析ツール: IRAF/PYRAF/ASTROPY

例：IRAFによるバッチ処理

```
imcomb @file-off1.list off1-average.fits combine=average
imcomb @file-on.list on-average.fits combine=average
imcomb @file-off2.list off2-average.fits combine=average
imcomb off1-average.fits,off2-average.fits off-
average.fits \
    combine=average weight=@scale-off.dat
imarith off-average.fits - on-average.fits off-on-
average.fits
```

天文学データ解析ツール: IRAF/PYRAF/ASTROPY

例：IRAFによるバッチ処理

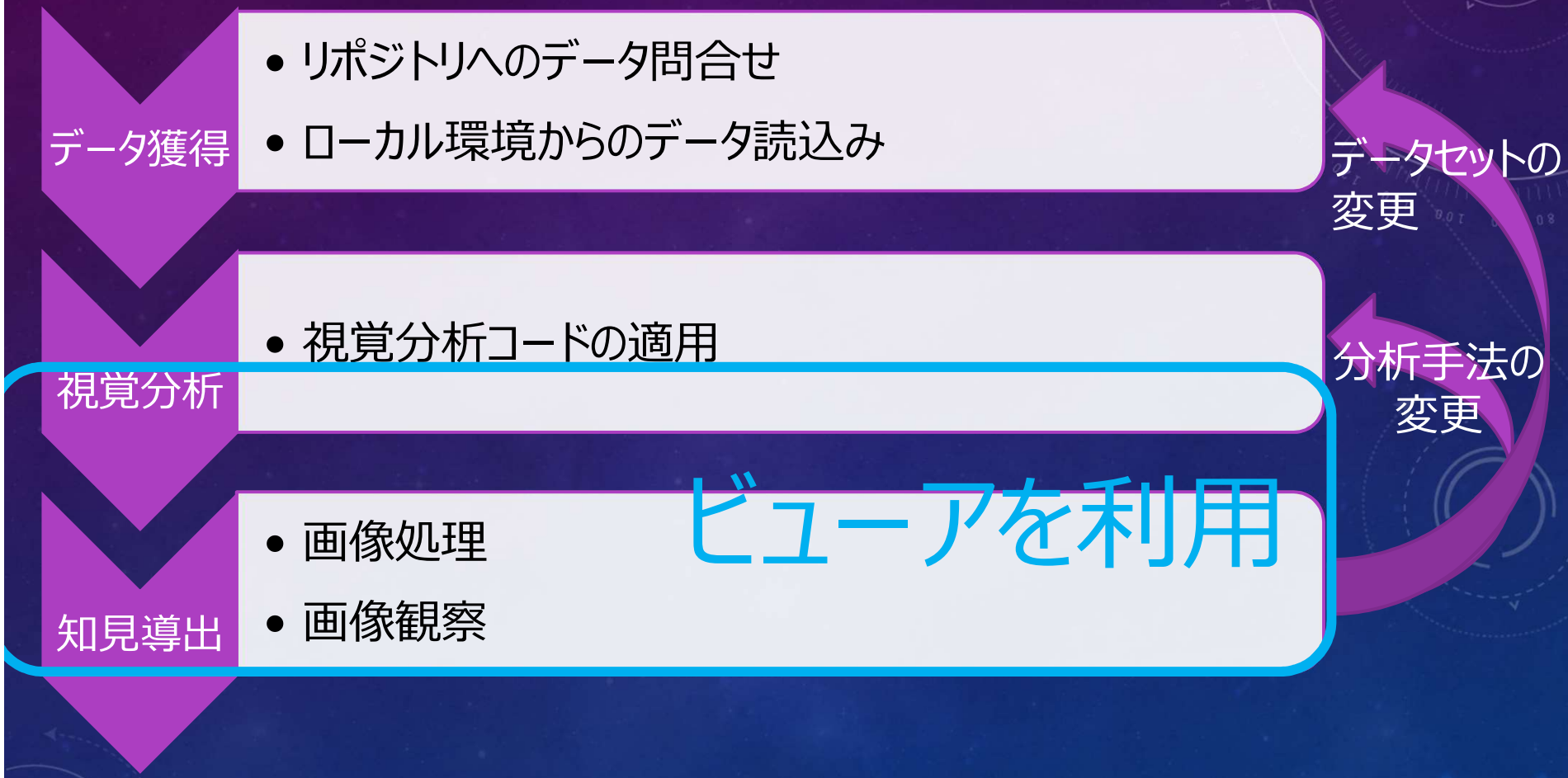
```
# Sub-datacube for each band

awk 'BEGIN{for (i = 3099; i < 3119; i++) {
    printf ("manga-8454-12703-
LINCUBE.fits[1][,,%d]¥n", i)}
  }' > file-off1.list
awk 'BEGIN{for (i = 3134; i < 3154; i++) {
    printf ("manga-8454-12703-
LINCUBE.fits[1][,,%d]¥n", i)}
  }' > file-on.list
awk 'BEGIN{for (i = 3174; i < 3194; i++) {
    printf ("manga-8454-12703-
LINCUBE.fits[1][,,%d]¥n", i)}
  }' > file-off2.list
```

```
# Then compute average on each band

imcomb @file-off1.list off1-average.fits
combine=average
imcomb @file-on.list on-average.fits
combine=average
imcomb @file-off2.list off2-average.fits
combine=average
```

```
# Combine off-band images

echo "0.533333" > scale-off.dat
echo "0.466667" >> scale-off.dat
imcomb off1-average.fits,off2-average.fits
off-average.fits ¥
    combine=average weight=@scale-off.dat
```

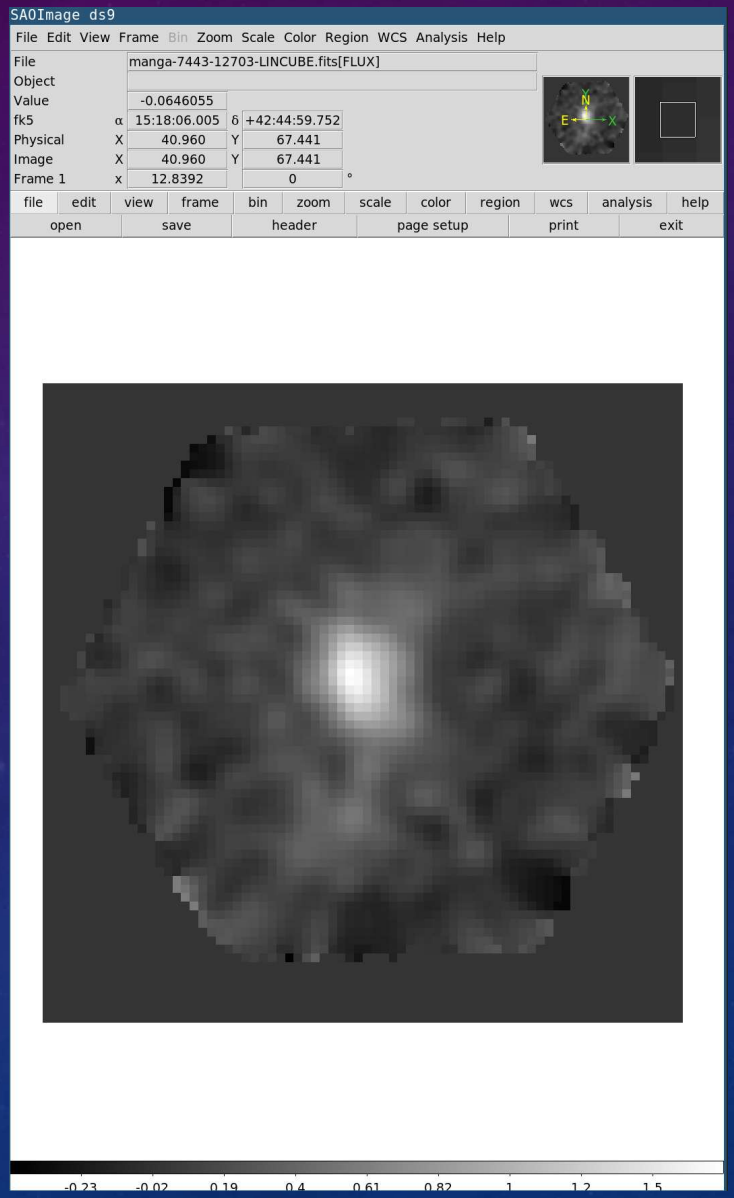
```
# Make equivalent-width map

imarith off-average.fits - on-average.fits off-
on-average.fits
imarith off-on-average.fits * 20 flux.fits
imarith flux.fits / off-average.fits
equivalent-width.fits
```

天文学の標準的ワークフロー

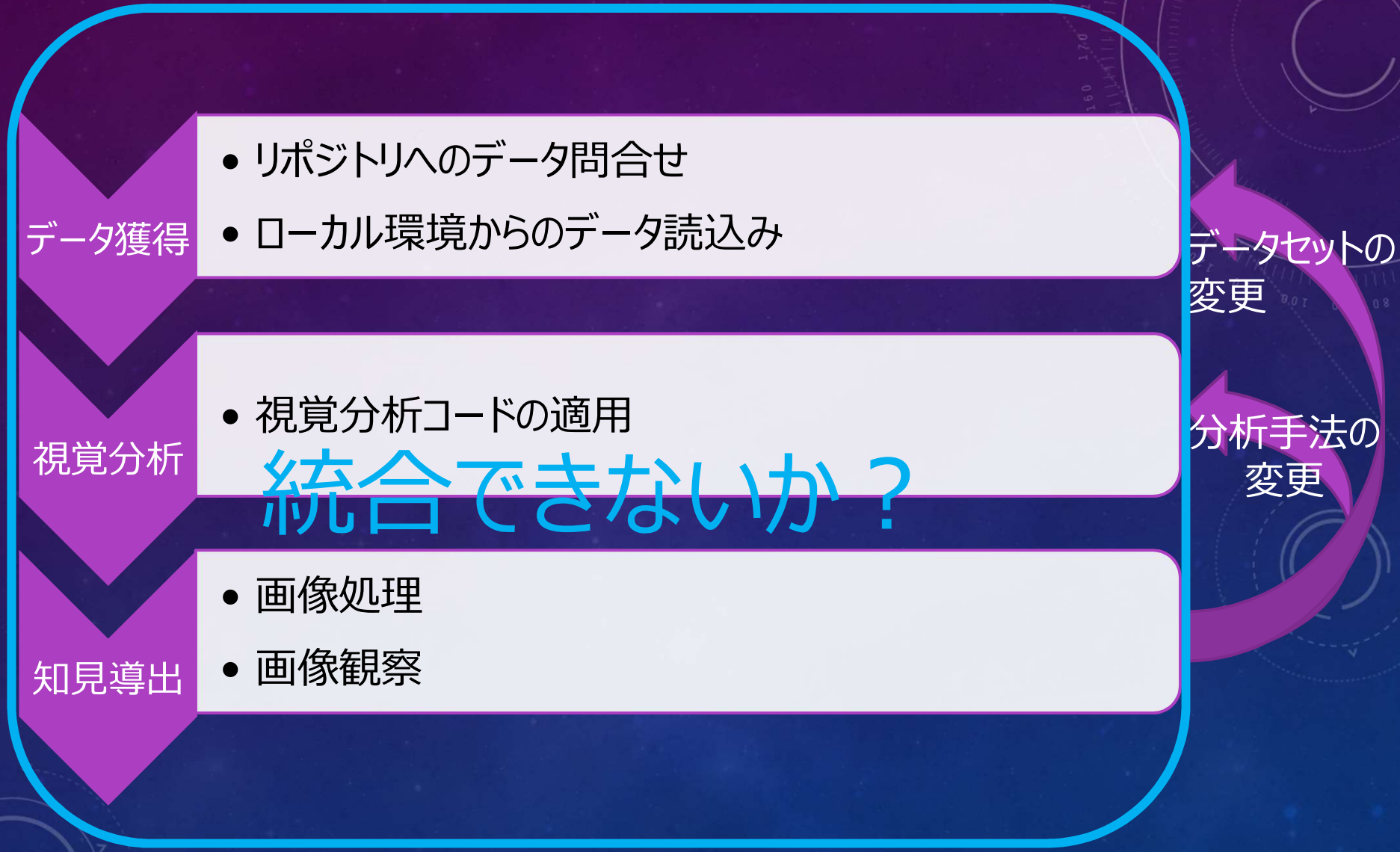


天文学データビューアの例: DS9/QFITSVIEW/NIGHTLIGHT/MATPLOTLIB

DS9

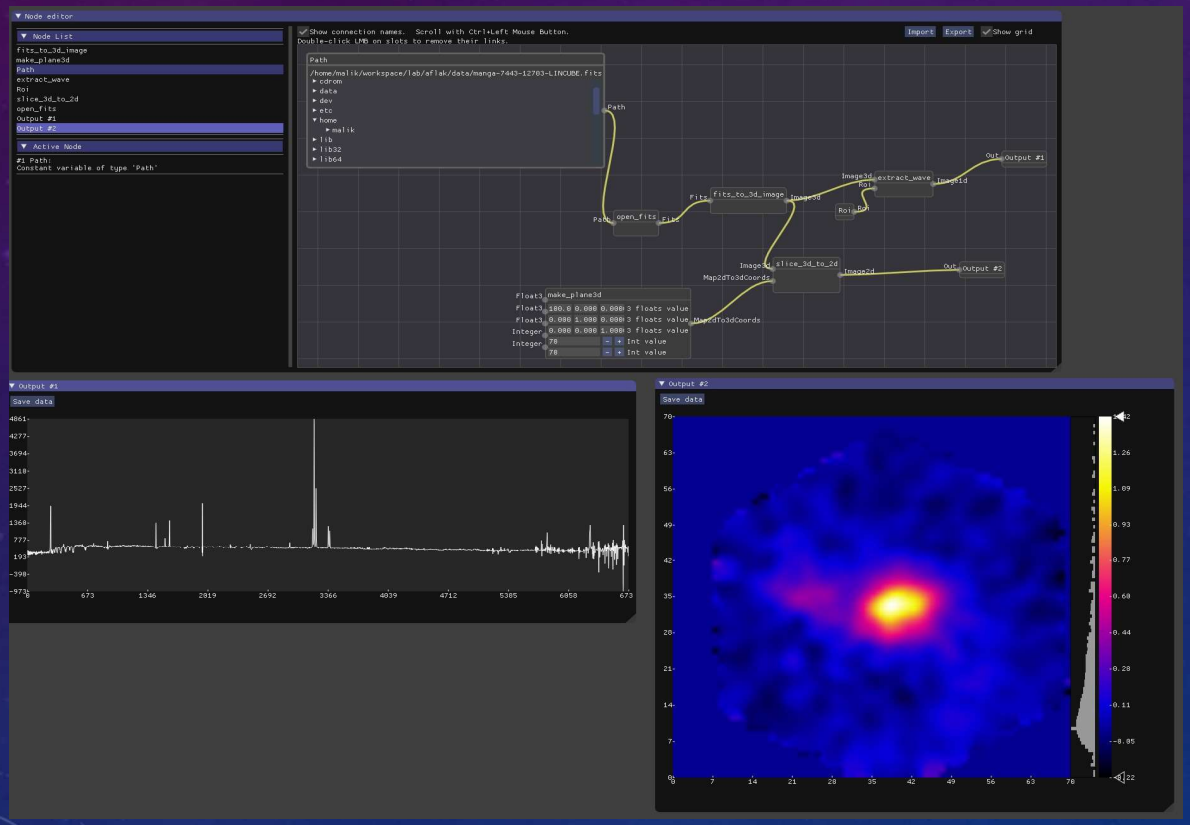


天文学の標準的ワークフロー



AFLAK—أفلاك

Advanced Framework for Learning Astrophysical Knowledge
("stars" in Arabic)



AFLAKの三大設計指針

- ビジュアルプログラミングの採用
- 可視化発見プロセスのサポート
- 出自管理の実現

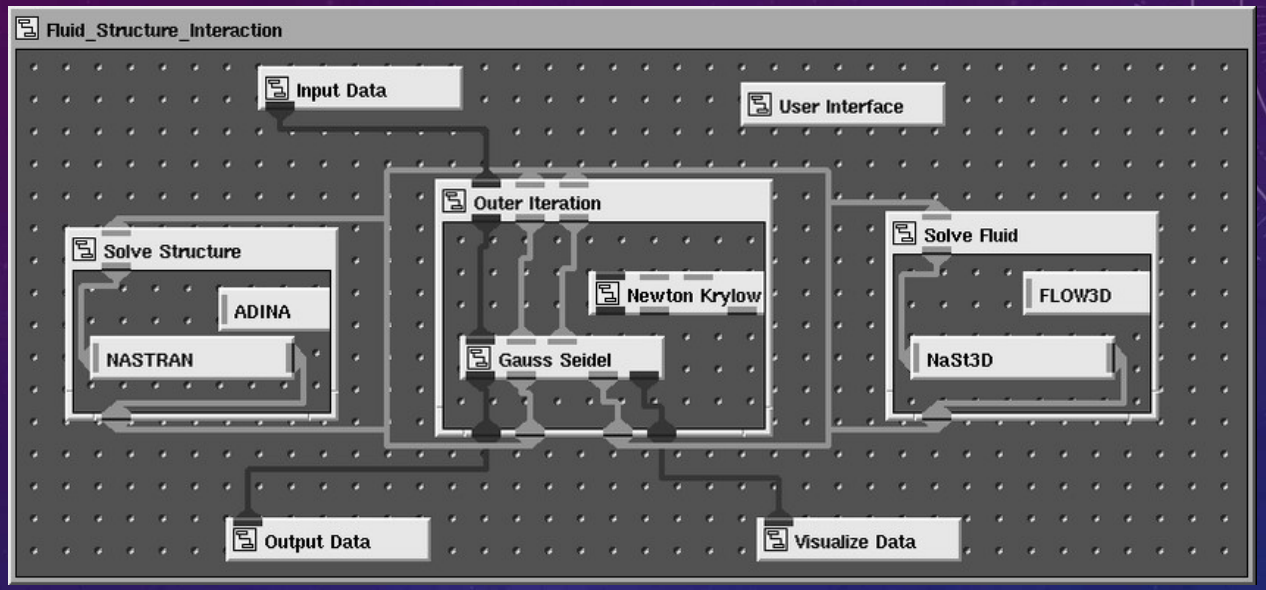
データフローパラダイム
[HABER&MCNABB:90]

可視化イディオム

- あらゆる可視化プログラムは左図の4フェーズに分解可能

ViSC Report [SIGGRAPH&NSF, *ACM Comput. Graph.*, **21**(6), 1987]

モジュール型可視化環境（MVE）

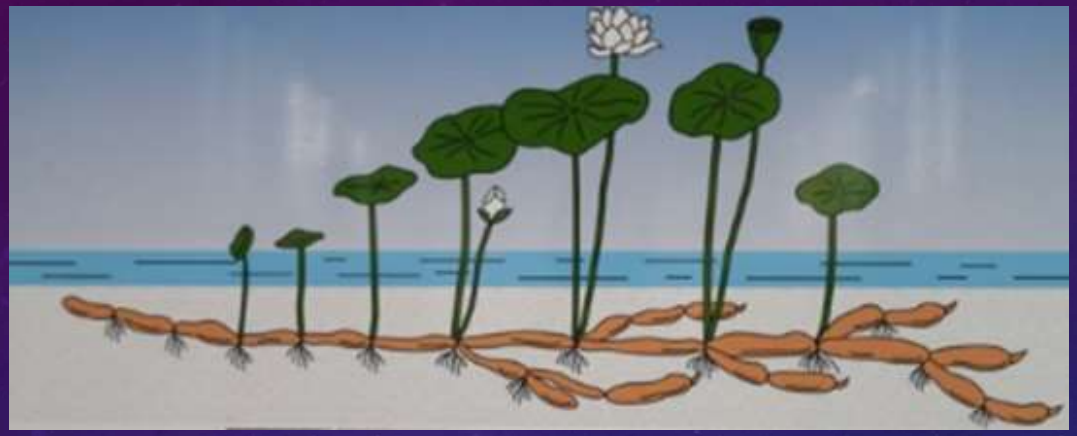


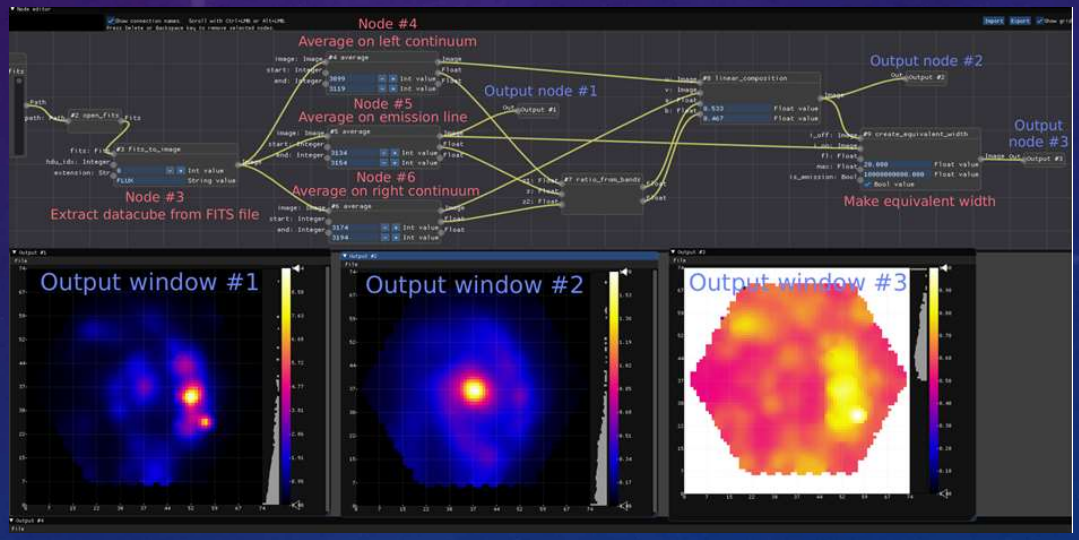
AVS/Express screenshot, from H.-J. Bungartz, et al.,
“Design and implementation of a modular environment for coupled systems.”

モジュール型ビジュアルプログラミング

- ✓ 非手続き性/**プラガビリティー**（再利用性/拡張性）
  - ＞分野専用ソフトウェア
- × 実効性能
  - ＜プログラム性能チューニング

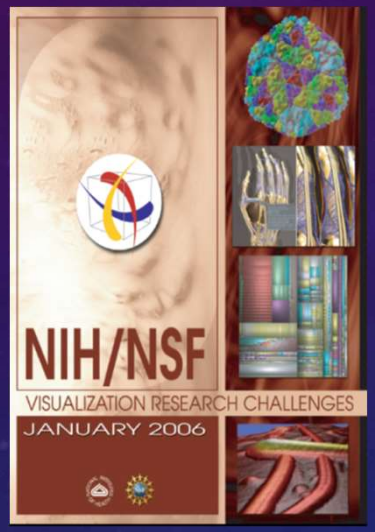
LOTUS ROOT : AFLAKプログラミング メタファ

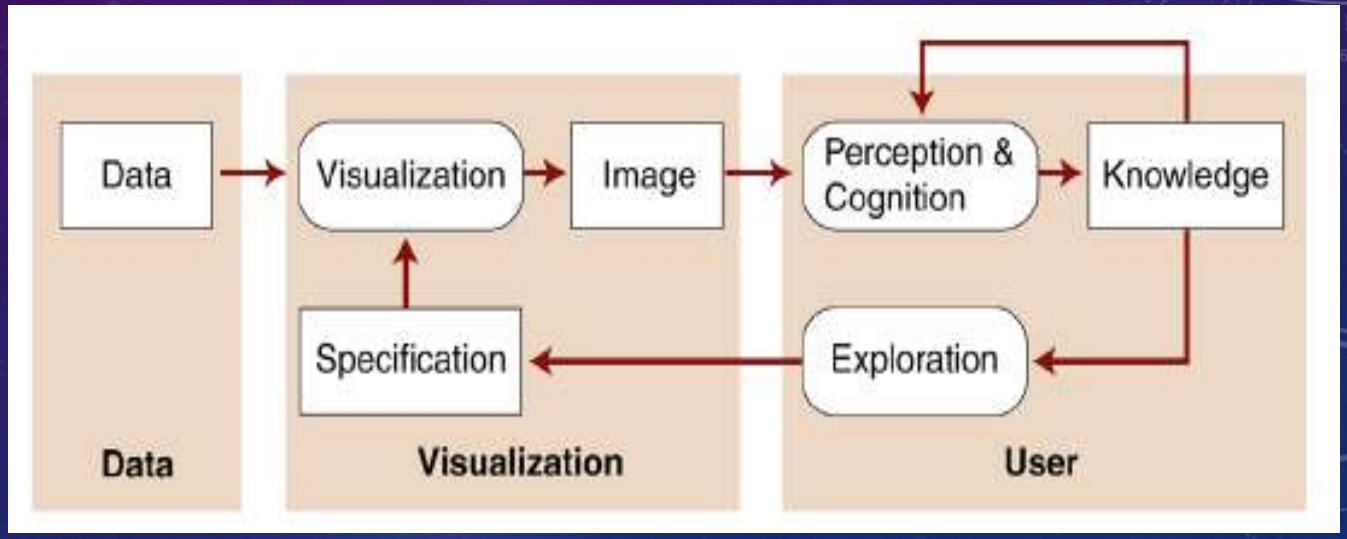




可視化発見プロセスのサポート

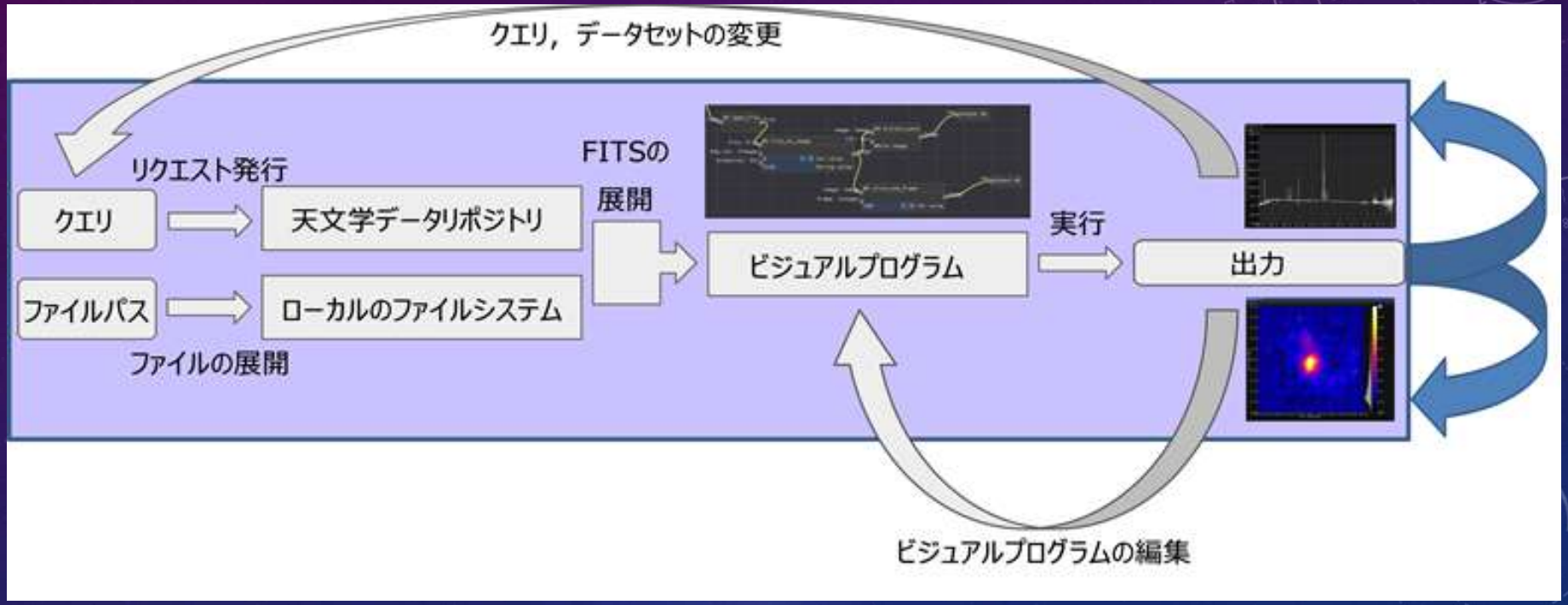
“Computational science now encompasses modeling and simulation using data from many sources, requiring data management, mining and interrogation.”





- Human（astronomer）-in-the-loopの実現
- 出自（provenance）の管理：
  視覚分析プロセスの記録，追跡，再利用→集合知

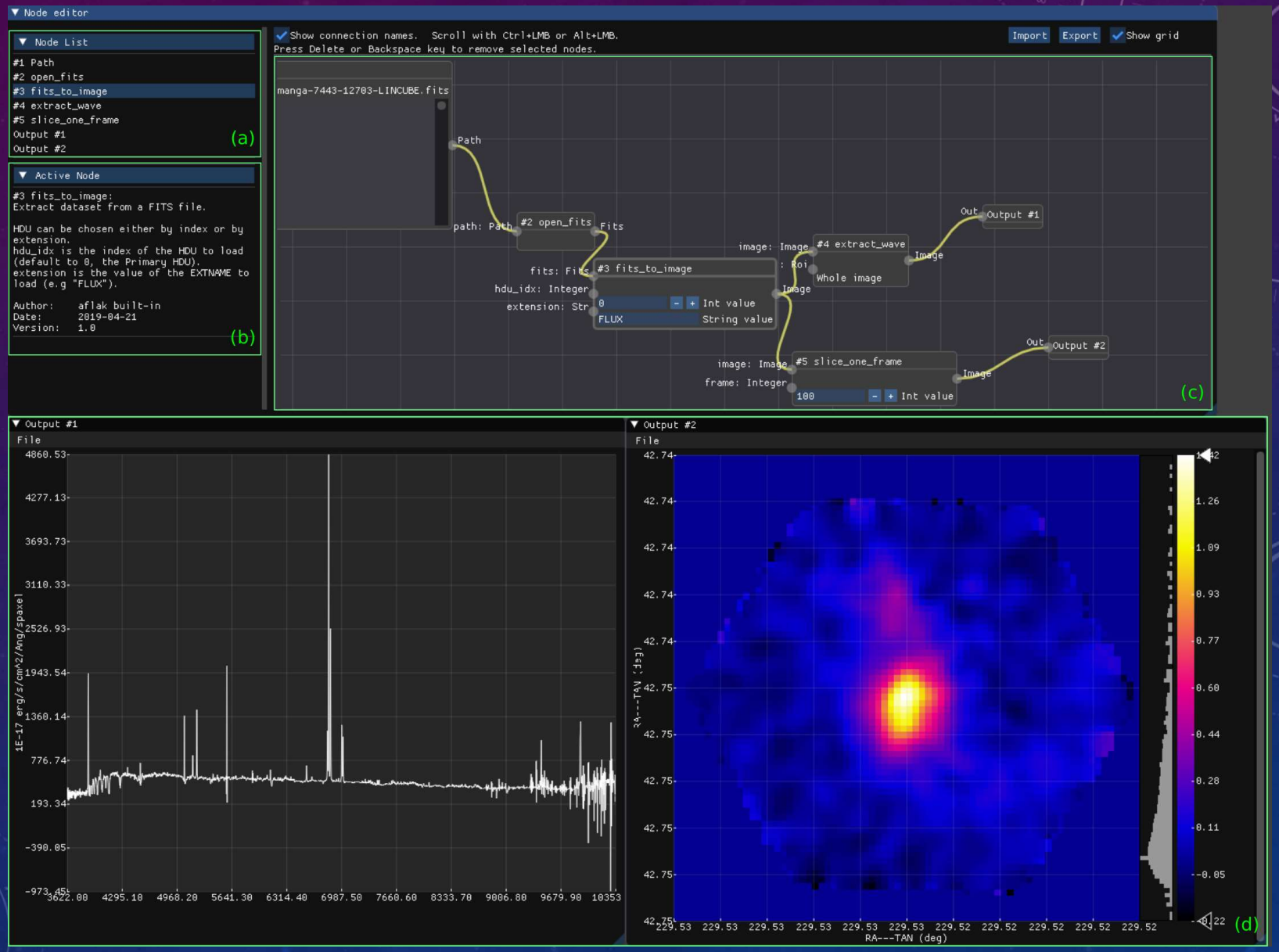
AFLAKのフレームワーク



システム概要

概要

http://aflak.jp



設計における目標

- 使いやすさと応答性
- 再利用可能性と拡張可能性
- 協同開発の支援とコードの共有

RUST

Documentation Install Community Contribute

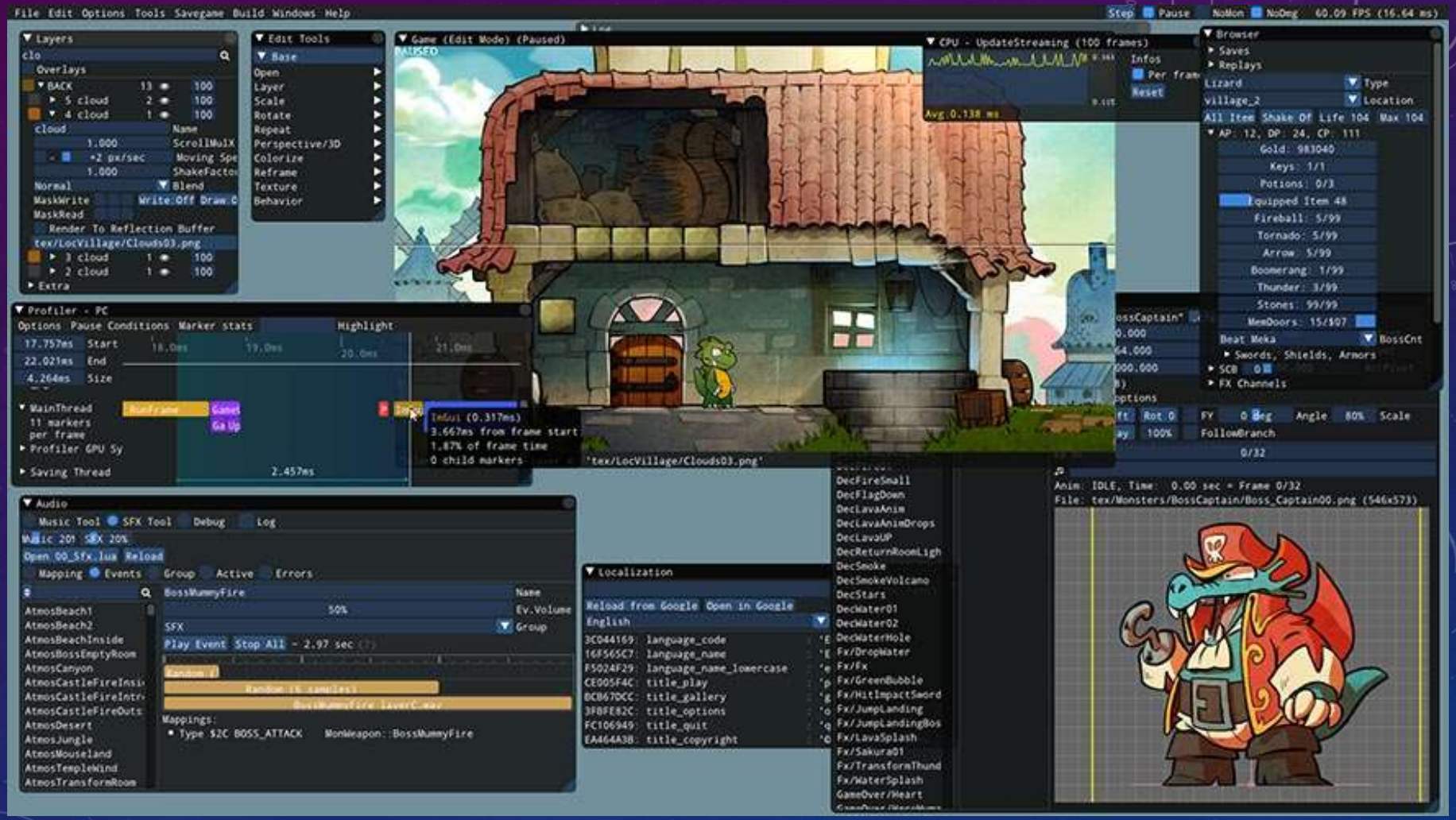
Rust is a systems programming language that runs blazingly fast, prevents segfaults, and guarantees thread safety.

Install Rust **1.30.1**

November 8, 2018

https://www.rust-lang.org/

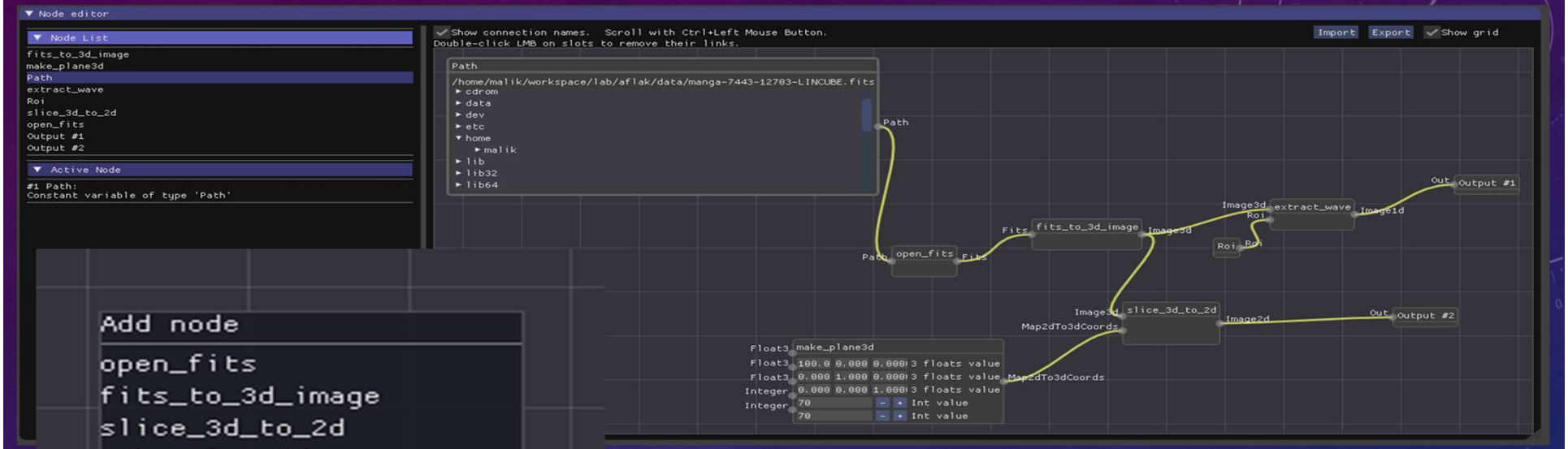
DEAR IMGUI（C++）

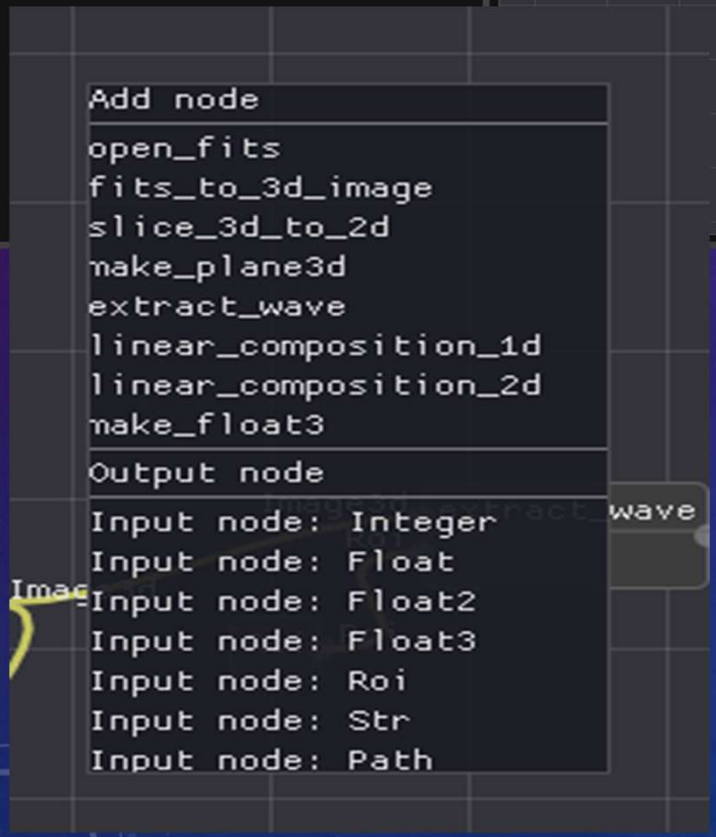


https://github.com/ocornut/imgui

最新版へアップデート済！

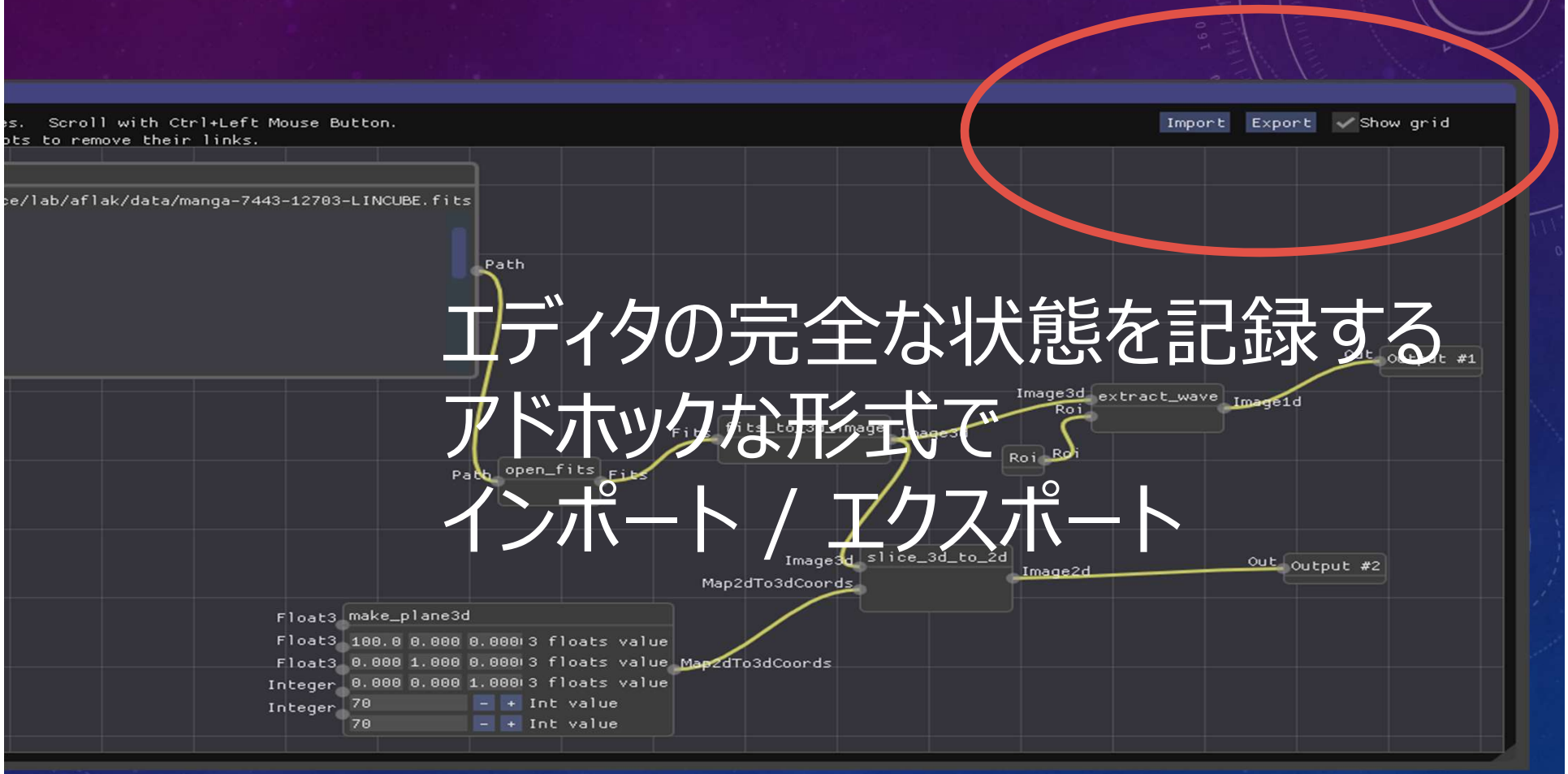
ノードエディタ





- リストからノードを追加
- 各々のノードを配線
- 出力ノードが作成・配線されると対応する出力窓が表示

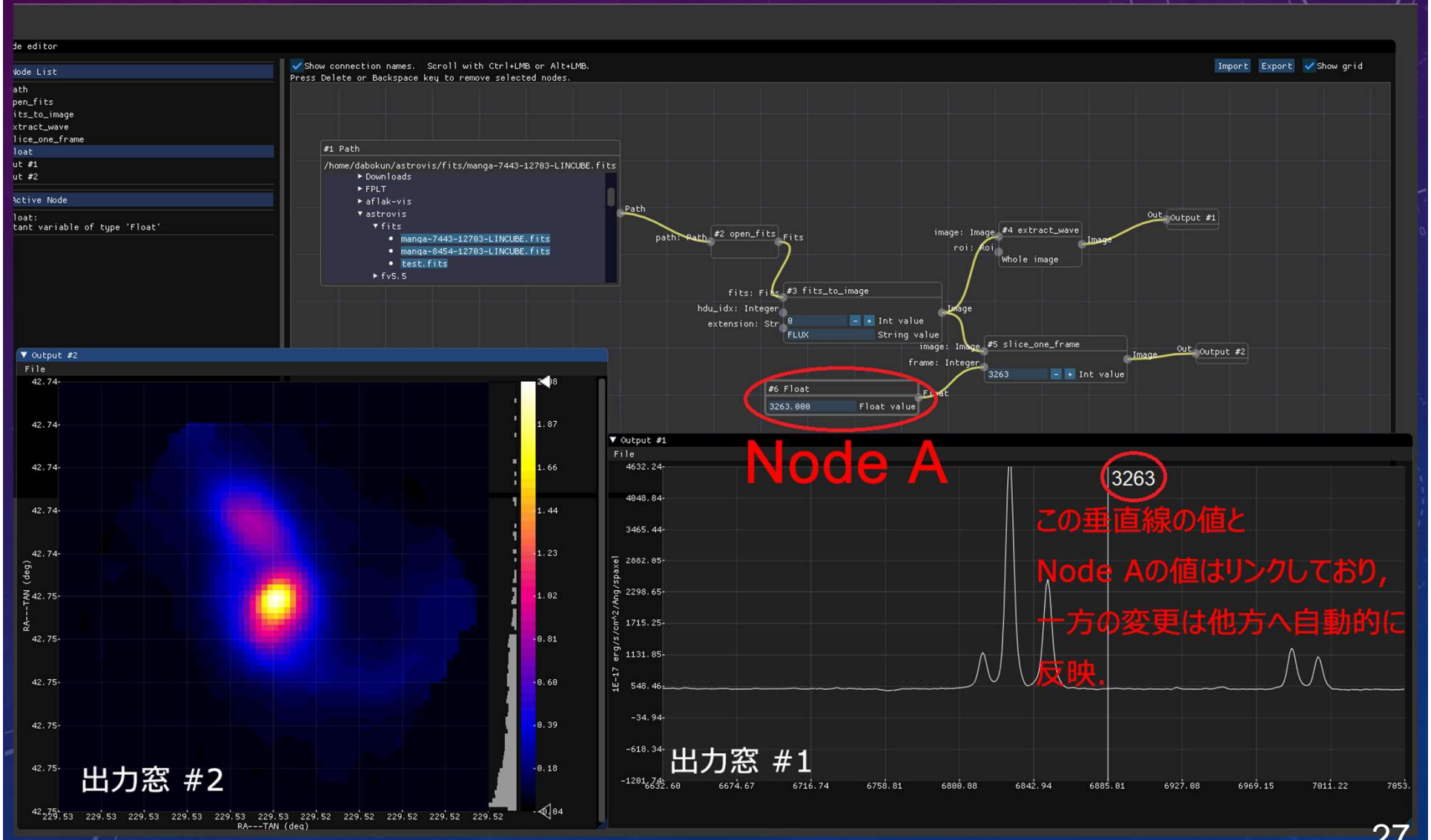
ノードプログラムのインポート / エクスポート



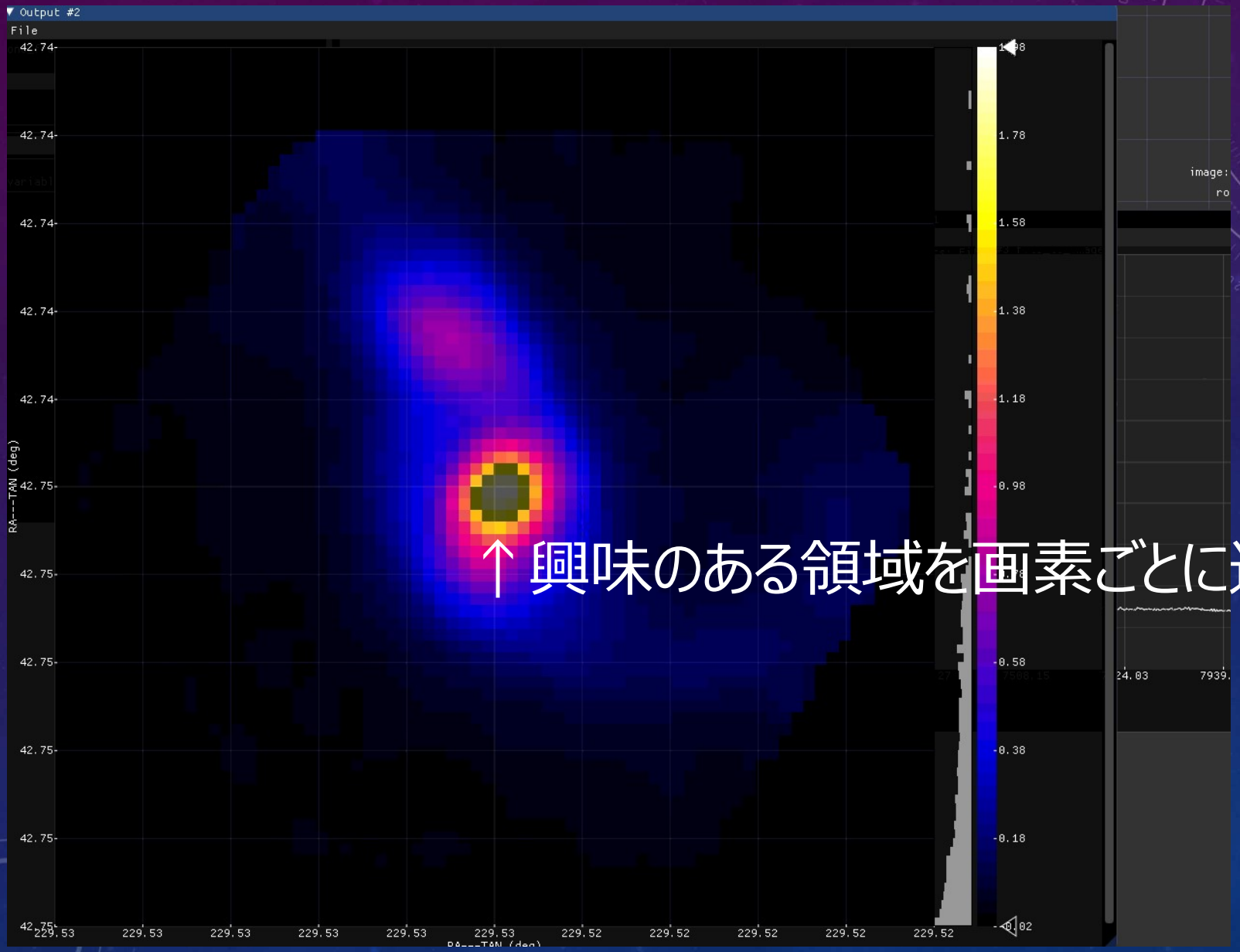
エディタの完全な状態を記録する
アドホックな形式で
インポート / エクスポート

http://aflak.jp

ノードエディタの変数と可視化結果の間の双方向バインディング

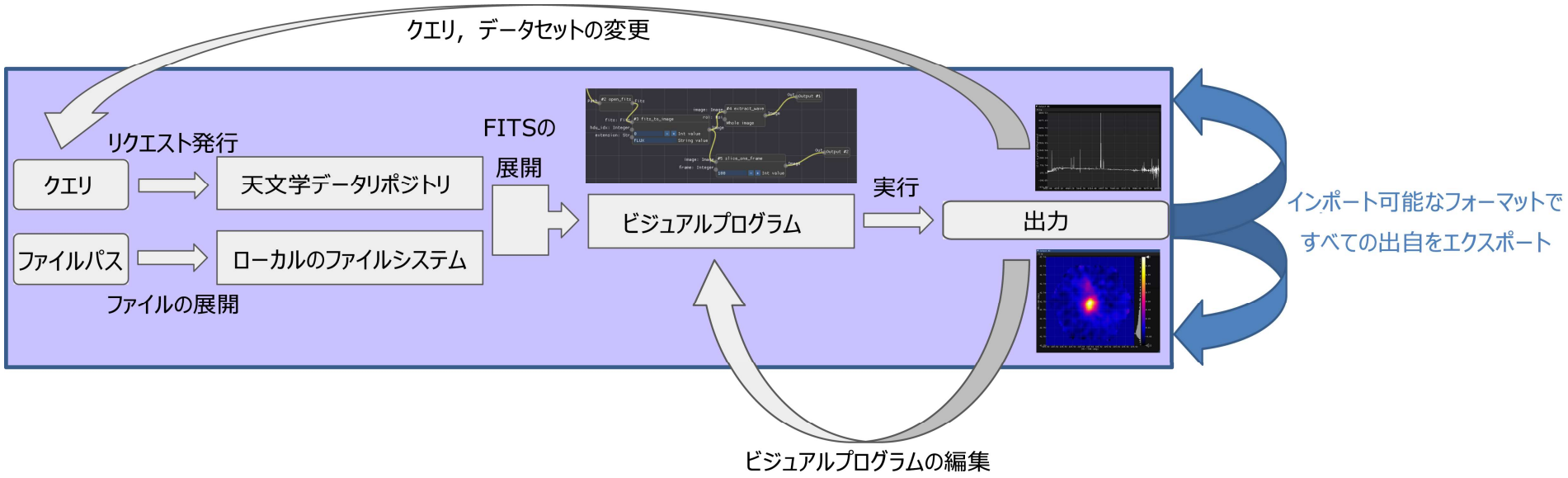


興味のある領域の選択

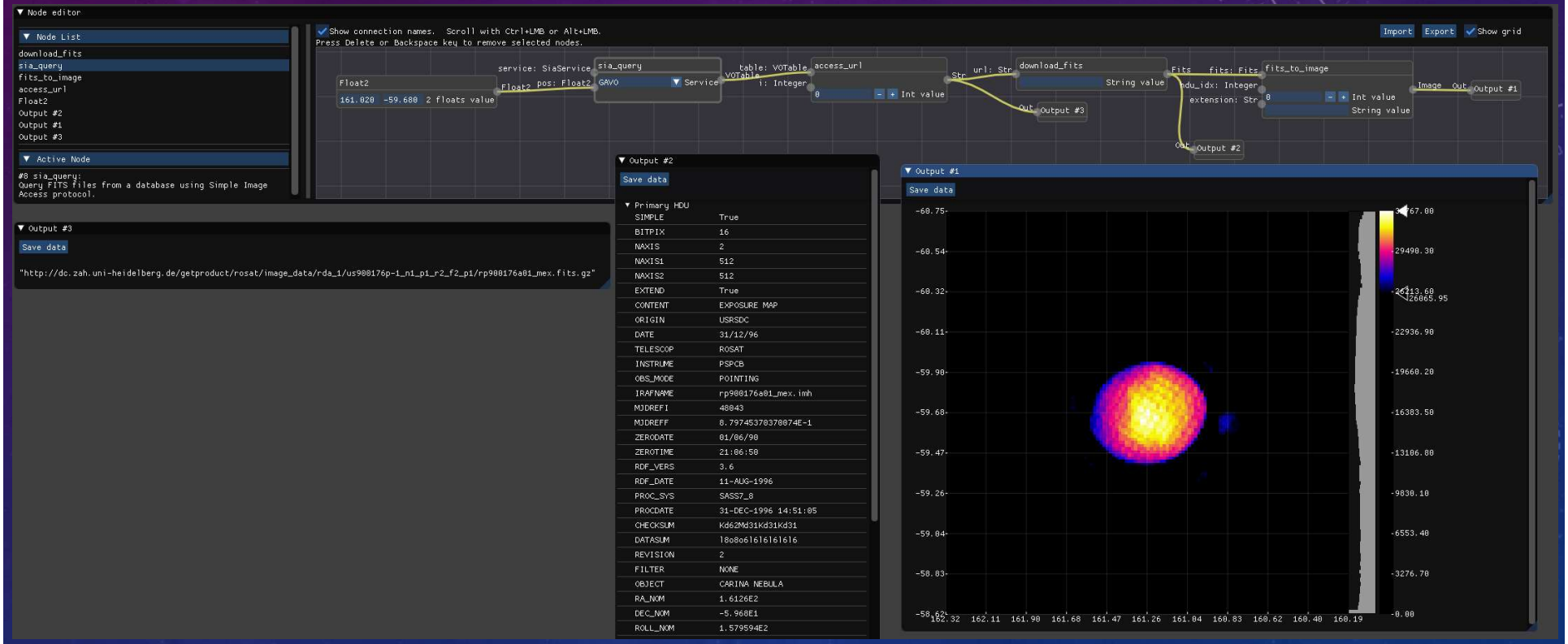


↑興味のある領域を画素ごとに選択

ワークフロー

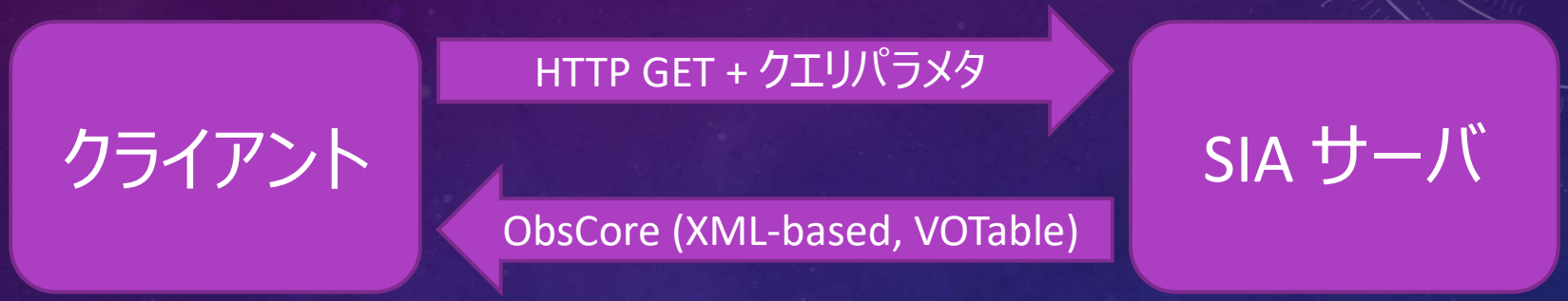

クエリ，データセットの変更
リクエスト発行
クエリ
天文学データリポジトリ
ファイルパス
ローカルのファイルシステム
ファイルの展開
FITSの
展開
ビジュアルプログラム
実行
出力
インポート可能なフォーマットで
すべての出自をエクスポート
ビジュアルプログラムの編集

VO標準を利用したリポジトリへのデータ問合せ



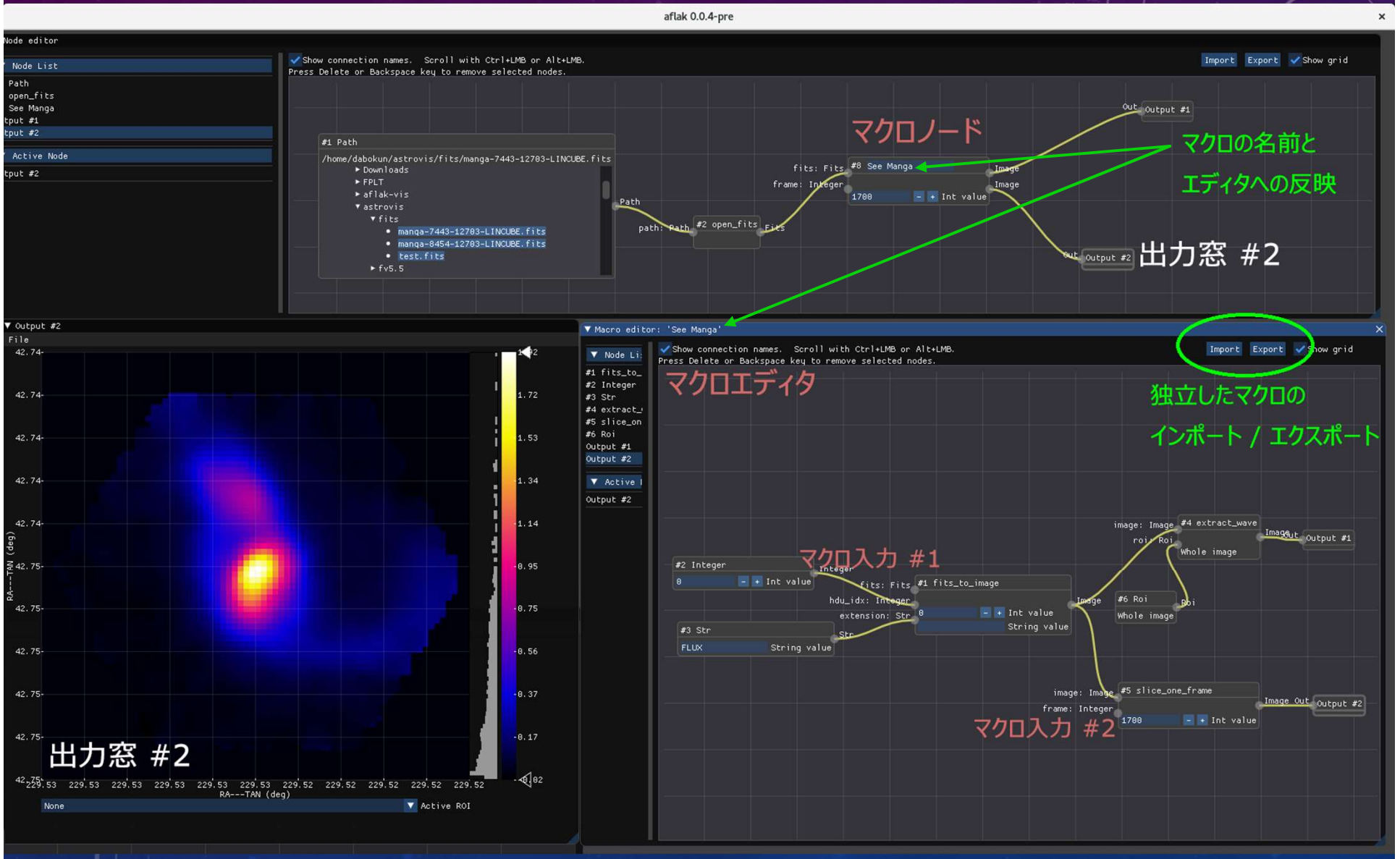
SIA 実装 (SIMPLE IMAGE ACCESS)

入力: SIA サーバURL，天空座標， 周波数，等.

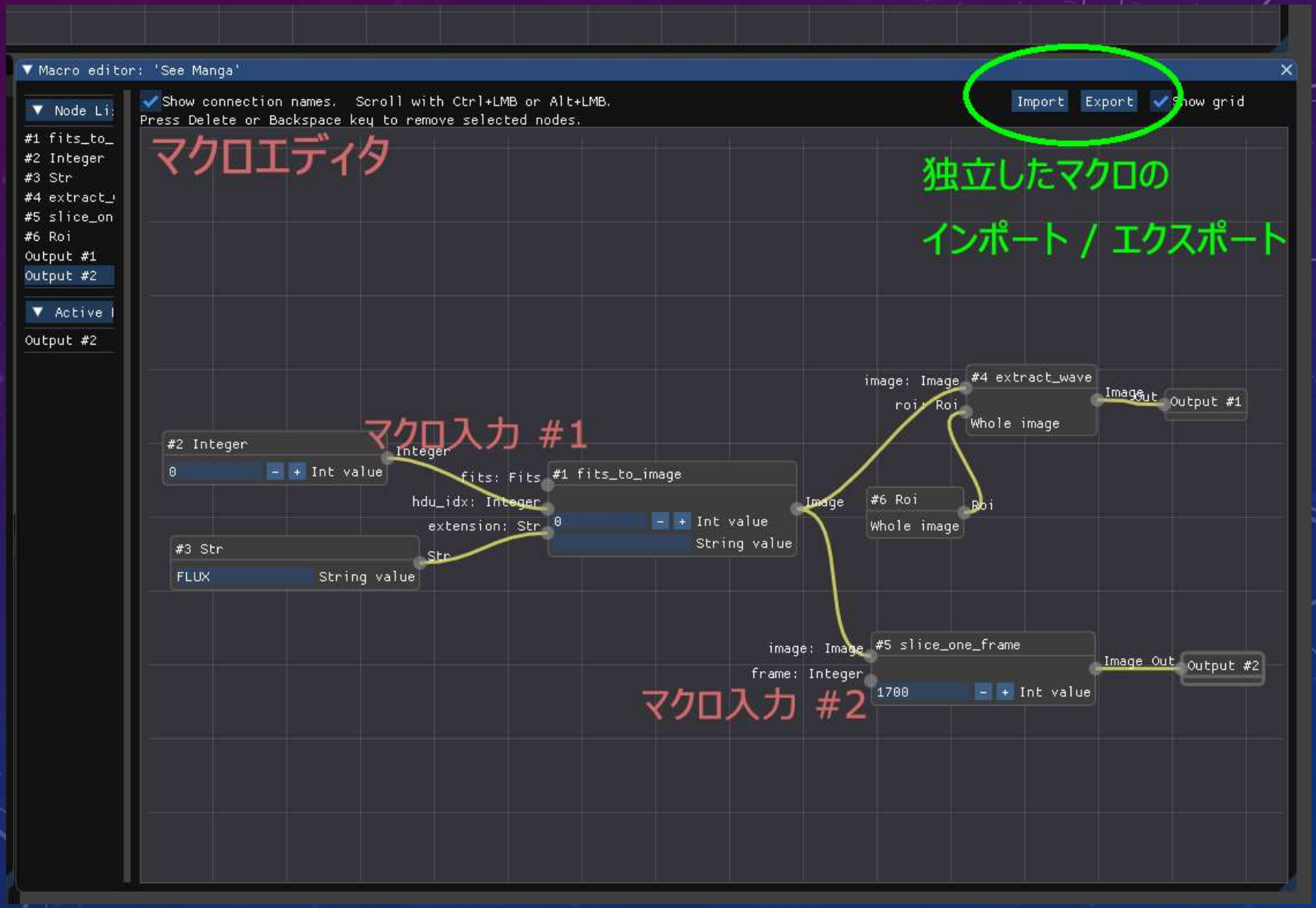


- 出力
  - `obj_publisher_id`: オブジェクトのユニークID
  - `access_url`: オブジェクトのダウンロードリンク
  - `access_format`: ダウンロードされるファイルのMIMEタイプ
- ファイルをダウンロードし，解凍

高速なマクロ機能


aflak 0.0.4-pre
Node editor
Node List
Active Node
Show connection names. Scroll with Ctrl+LMB or Alt+LMB.
Press Delete or Backspace key to remove selected nodes.
Import
Export
Show grid
#1 Path
/home/dabokun/astrovis/fits/manga-7443-12703-LINCUBE.fits
Downloads
FPLT
aflak-vis
astrovis
fits
manga-7443-12703-LINCUBE.fits
manga-8454-12703-LINCUBE.fits
test.fits
fv5.5
マクロノード
マクロの名前と
エディタへの反映
#2 open_fits
#8 See Manga
1700
Int value
Output #1
Output #2
出力窓 #2
Output #2
File
出力窓 #2
RA---TAN (deg)
Active ROI
Macro editor: 'See Manga'
マクロエディタ
独立したマクロの
インポート / エクスポート
マクロ入力 #1
マクロ入力 #2
#1 fits_to_image
#2 Integer
#3 Str
#4 extract_wave
#5 slice_one_frame
#6 Roi
FLUX
String value
Whole image

協同的なコードの共有と再利用性



応用例

ケーススタディ

三次元面分光データの応用三例

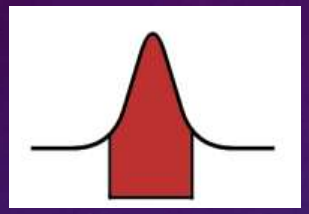
- 等価幅マップの生成
- 速度場マップの生成
- 任意スライシング

ケーススタディ

三次元面分光データの応用三例

- 等価幅マップの生成
- 速度場マップの生成
- 任意スライシング

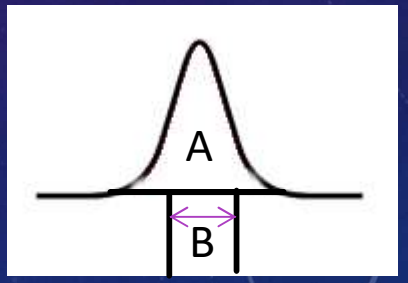
スペクトルから等価幅を計算



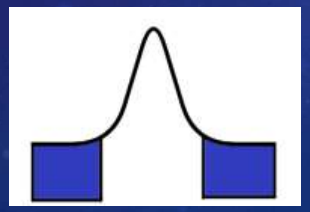
on-band 平均輝度

データ
キューブ

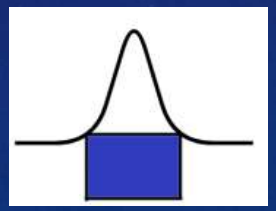
等価幅の計算



off-band 平均輝度



連続光の推定



スペクトルから等価幅を計算（cont'd）

①on-band と off-band （二箇所）における平均フラックスを
　計算（average ノード）
②on-bandにおける連続光の推定
　（ratio_from_bands ノード, linear_composition ノード）
③等価幅の計算（create_equivalent_width ノード）

その他
・定数（Constant ノード）
・データ中の最大値，最小値の取得（image_min_max ノード）
・対数スケーリング（convert_to_logscale ノード）

①平均フラックスの計算

average ノード

入力: 三次元画像 im

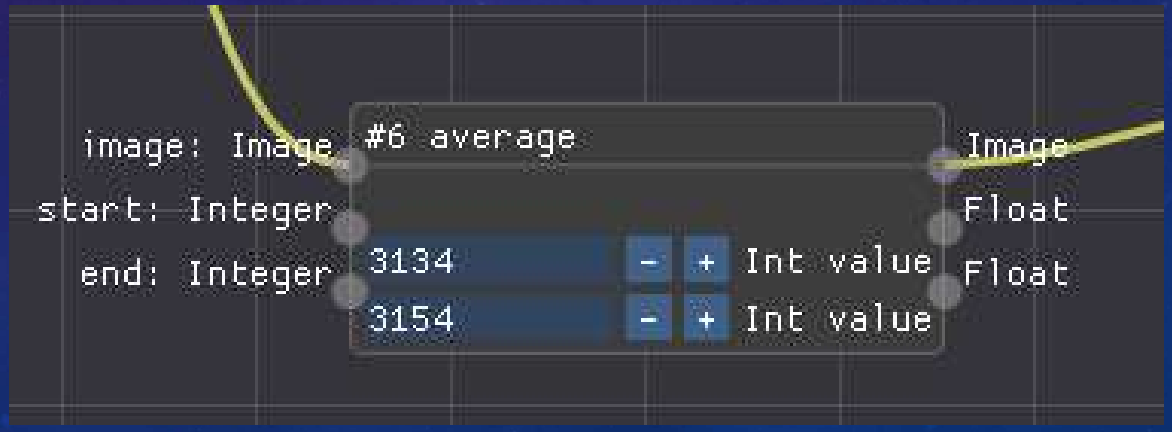
開始波長 $start$

終了波長 end $(start < end)$

出力: 積分結果 $\sum_{start}^{end} \frac{im_k}{end-start}$ for all pixels

中心波長 $\frac{start+end}{2}$

波長幅 $end - start$



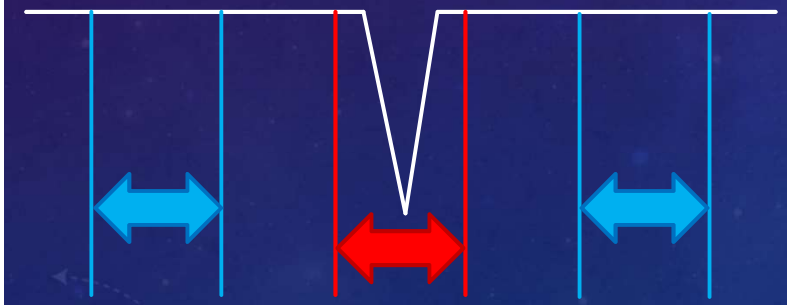
②on-band まわりの連続光推定

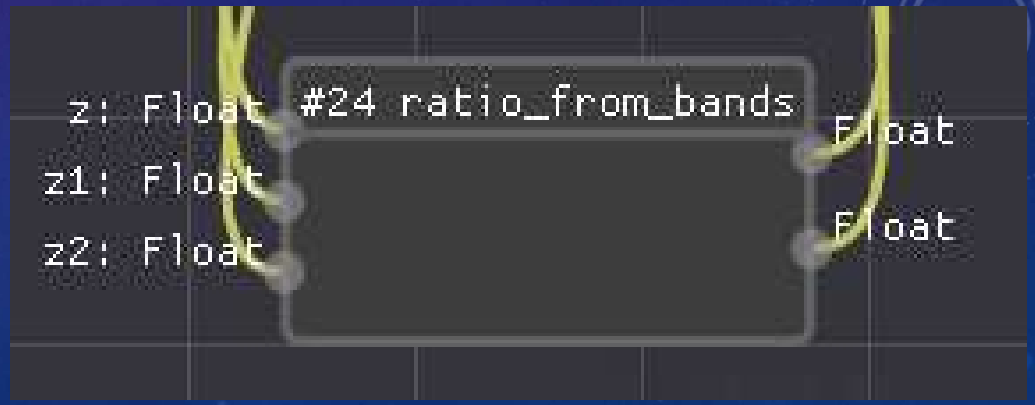
ratio_from_bands ノード

入力: on-band の中心波長 z
2つの off-band の中心波長 z_1, z_2 $(z_1 < z < z_2)$

出力: 2つの off-band の on-band に対する寄与

$$ratio_1 = \frac{z_2 - z}{z_2 - z_1},\ ratio_2 = \frac{z - z_1}{z_2 - z_1} = 1 - ratio_1$$



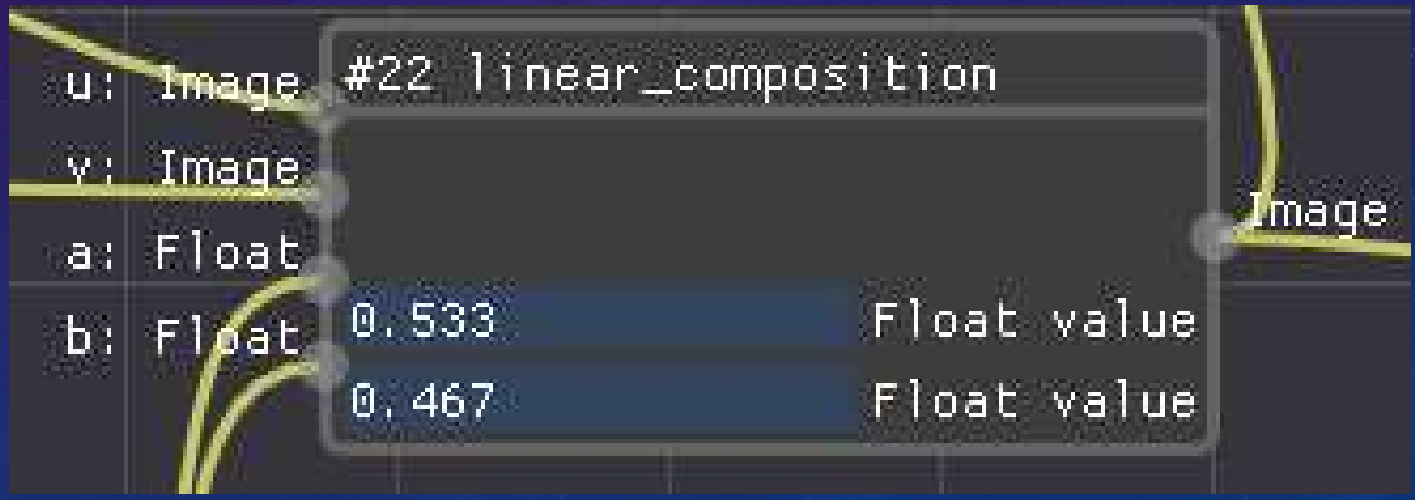


②on-band まわりの連続光推定（cont'd）

linear_composition ノード

入力: n次元イメージ u, v
定数 a, b

出力: $au + bv$



③等価幅の計算

create_equivalent_width ノード

入力: off-band の平均輝度 $flux_{off\_lc}$

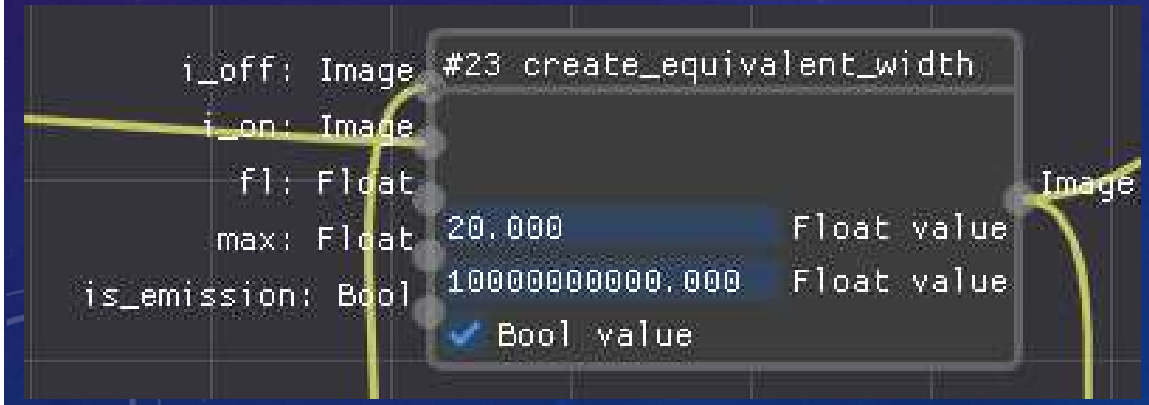
on-band の平均輝度 $flux_{on}$

on-band の波長幅 $range_{on}$

輝度値の最大値

輝線か否かを判断する真理値

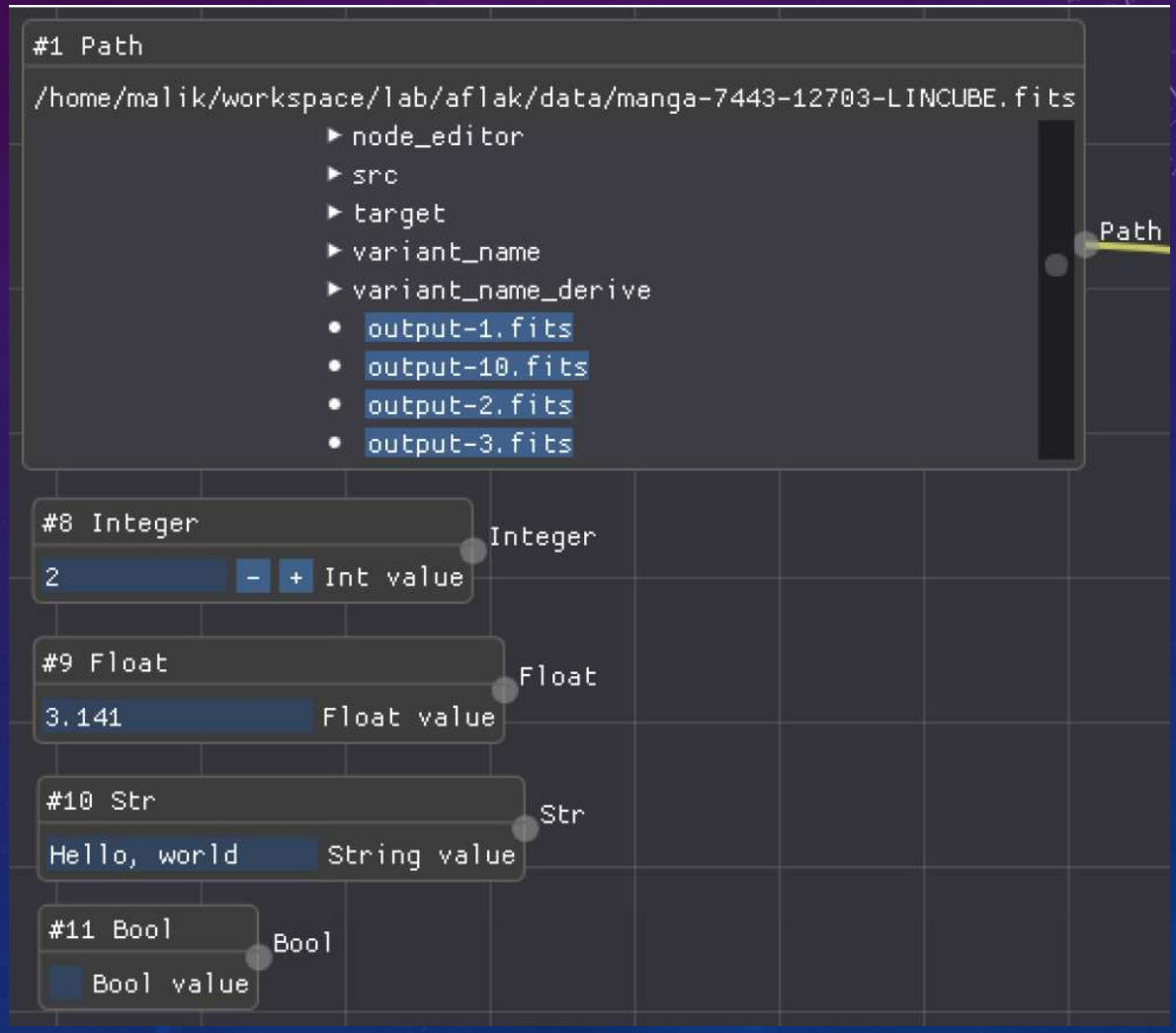
出力: 等価幅マップ（二次元イメージ）



$$(\text{等価幅}) = \frac{flux_{off\_lc} - flux_{on}}{flux_{off\_lc}} \times range_{on}$$

定数

整数，浮動小数点数，文字列，パス，真理値などの定数を表すノード



画像中の最大値/最小値

image_min_max ノード

入力: n次元イメージ（各画素に値vをもつ）

出力: 画素の値の最大値v_{min}, 最小値v_{max}

対数スケールへの変換

convert_to_logscale ノード

入力: n次元イメージ（各画素に輝度値vをもつ）

各変数（a(定数), v_{min}, v_{max}）

v_{min}, v_{max}はイメージ中の値の最大値と最小値

出力: n次元イメージ（対数スケール）

$x = \frac{v - v_{min}}{v_{max} - v_{min}}$として，$y = \frac{\ln(ax+1)}{\ln(a)}$を計算

参考:How DS9 renders an image
http://ds9.si.edu/doc/ref/how.html（2019年10月27日アクセス）

IRAFによる等価幅の計算例

```
# Sub-datacube for each band

awk 'BEGIN{for (i = 3099; i < 3119; i++) {
    printf ("manga-8454-12703-LINCUBE.fits[1][,,%d]¥n", i)}
  }' > file-off1.list
awk 'BEGIN{for (i = 3134; i < 3154; i++) {
    printf ("manga-8454-12703-LINCUBE.fits[1][,,%d]¥n", i)}
  }' > file-on.list
awk 'BEGIN{for (i = 3174; i < 3194; i++) {
    printf ("manga-8454-12703-LINCUBE.fits[1][,,%d]¥n", i)}
  }' > file-off2.list
```

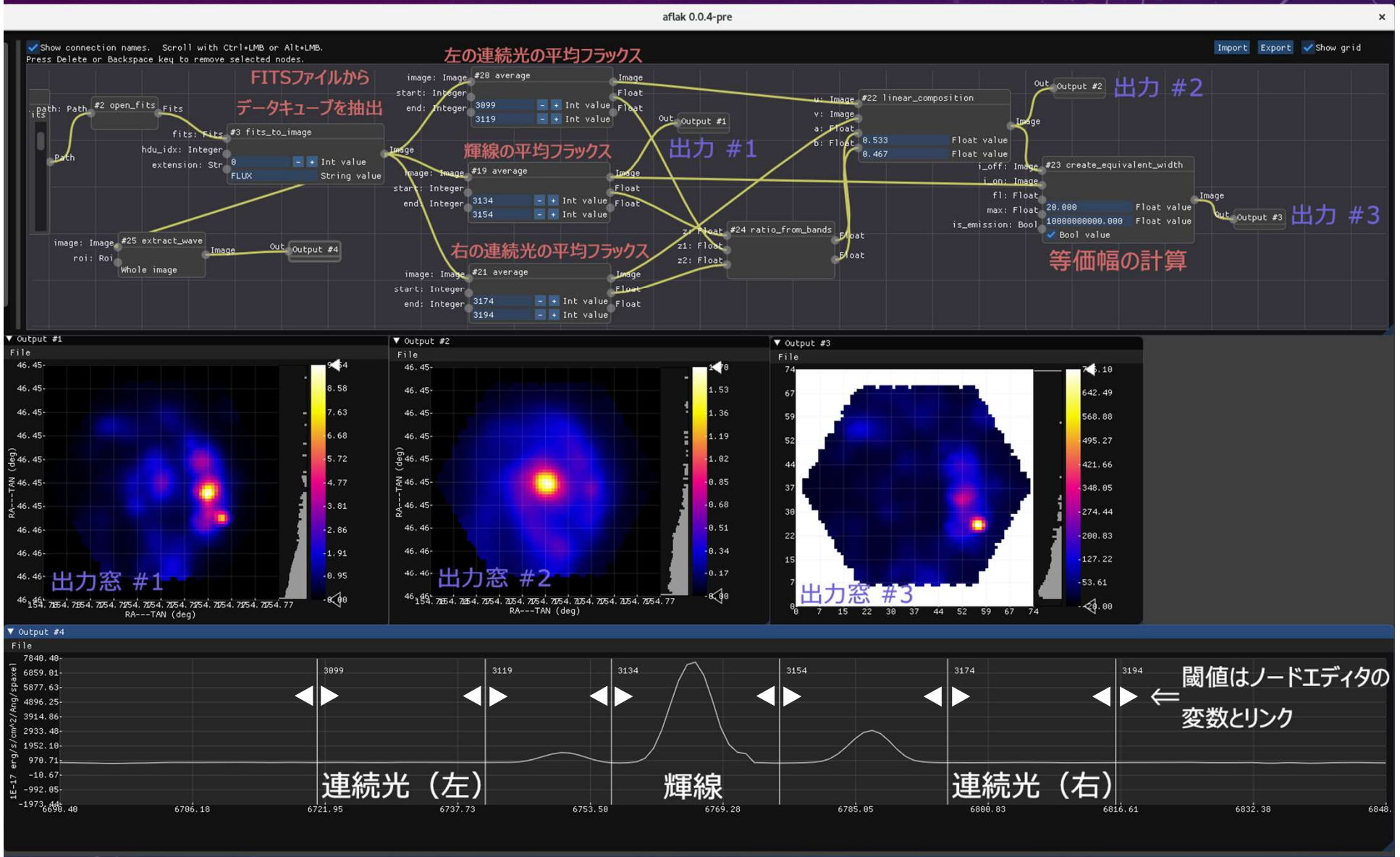
```
# Then compute average on each band

imcomb @file-off1.list off1-average.fits combine=average
imcomb @file-on.list on-average.fits combine=average
imcomb @file-off2.list off2-average.fits combine=average
```

IRAFによる等価幅の計算例（cont'd）

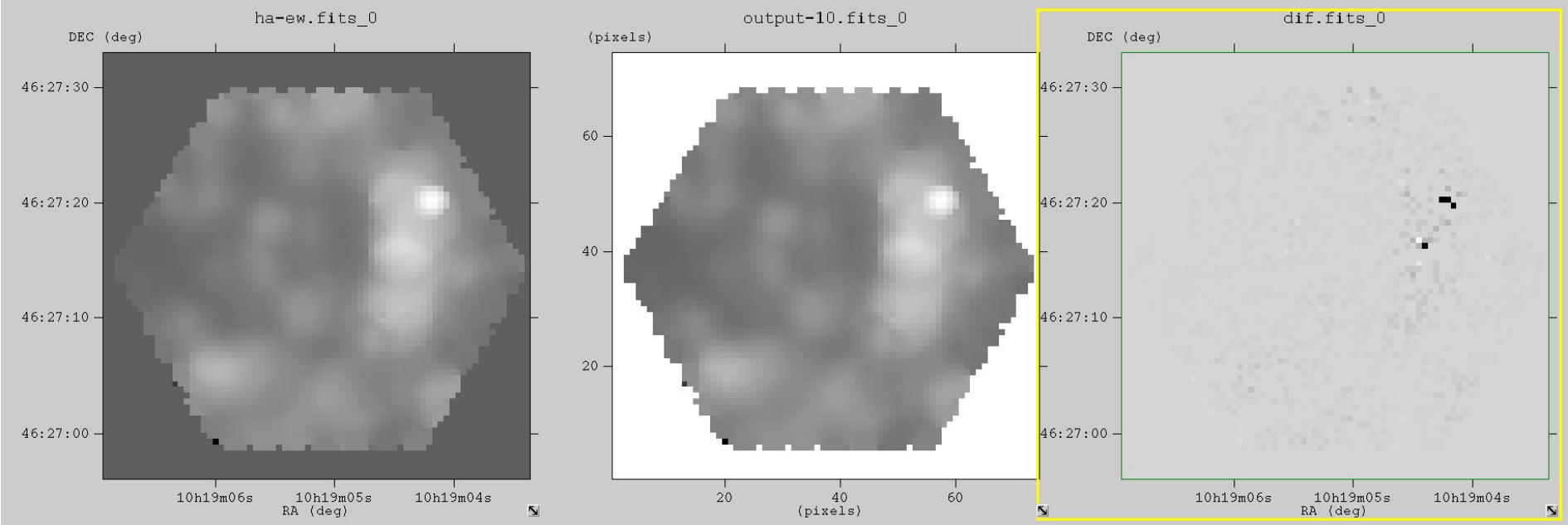
```
# Combine off-band images

echo "0.533333" > scale-off.dat
echo "0.466667" >> scale-off.dat
imcomb off1-average.fits,off2-average.fits off-average.fits \
    combine=average weight=@scale-off.dat
```

```
# Make equivalent-width map

imarith off-average.fits - on-average.fits off-on-average.fits
imarith off-on-average.fits * 20 flux.fits
imarith flux.fits / off-average.fits equivalent-width.fits
```

結果（等価幅マップ）


aflak 0.0.4-pre
Show connection names. Scroll with Ctrl+LMB or Alt+LMB.
Press Delete or Backspace key to remove selected nodes.
Import
Export
Show grid
FITSファイルから
データキューブを抽出
左の連続光の平均フラックス
輝線の平均フラックス
右の連続光の平均フラックス
出力 #1
出力 #2
出力 #3
等価幅の計算
#2 open_fits
#3 fits_to_image
#19 average
#20 average
#21 average
#22 linear_composition
#23 create_equivalent_width
#24 ratio_from_bands
#25 extract_wave
Output #1
File
出力窓 #1
出力窓 #2
出力窓 #3
RA---TAN (deg)
Output #4
1E-17 erg/s/cm^2/Ang/spaxel
3099
3119
3134
3154
3174
3194
閾値はノードエディタの
変数とリンク
連続光（左）
輝線
連続光（右）

評価（等価幅）



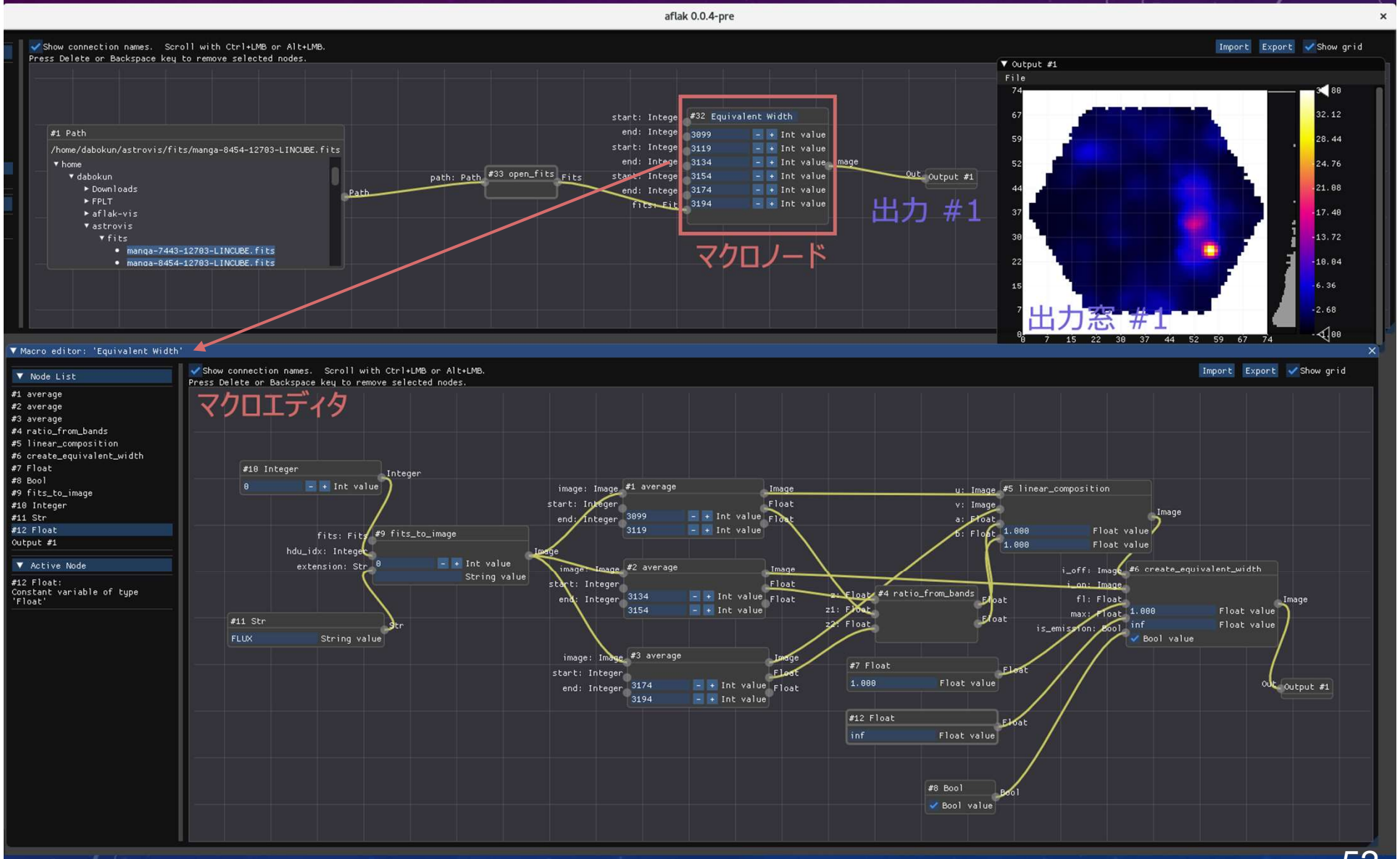
松林（京都大学）
IRAFを使用して計算

aflak

差分
平均値:約10^{-6}

誤差は約2.2×10^{-6}%

等価幅計算のマクロ

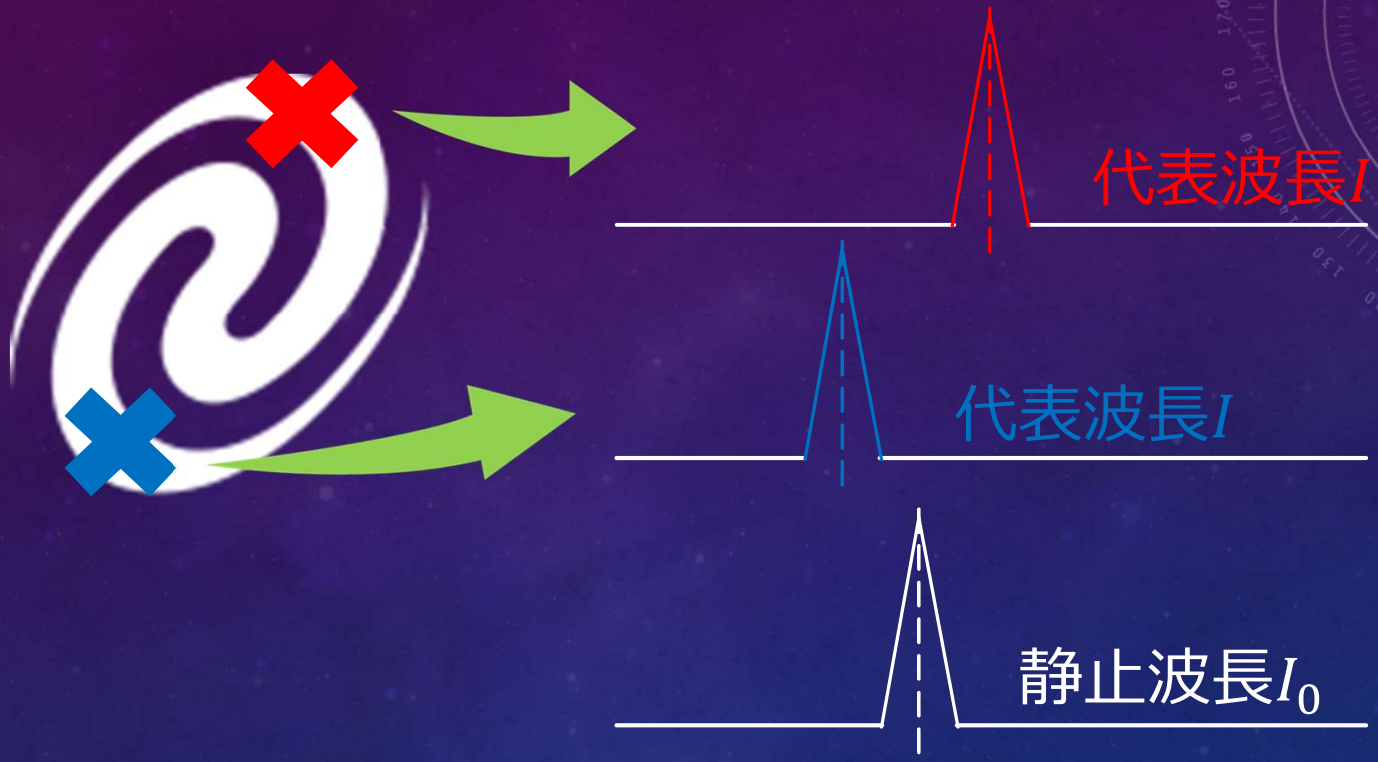

aflak 0.0.4-pre
Show connection names. Scroll with Ctrl+LMB or Alt+LMB.
Press Delete or Backspace key to remove selected nodes.
Import
Export
Show grid
#1 Path
/home/dabokun/astrovis/fits/manga-8454-12703-LINCUBE.fits
#33 open_fits
#32 Equivalent Width
マクロノード
出力 #1
Output #1
出力窓 #1
Macro editor: 'Equivalent Width'
Node List
#1 average
#2 average
#3 average
#4 ratio_from_bands
#5 linear_composition
#6 create_equivalent_width
#7 Float
#8 Bool
#9 fits_to_image
#10 Integer
#11 Str
#12 Float
Output #1
Active Node
マクロエディタ

ケーススタディ

三次元面分光データの応用三例

- 等価幅マップの生成
- 速度場マップの生成
- 任意スライシング

スペクトルから速度場の計算



cを光速として，

(その場における速度) $\approx c \times \frac{I-I_0}{I_0}$　($\frac{I-I_0}{I_0}$ が十分小さい場合の近似)

スペクトルから速度場の計算（cont'd）

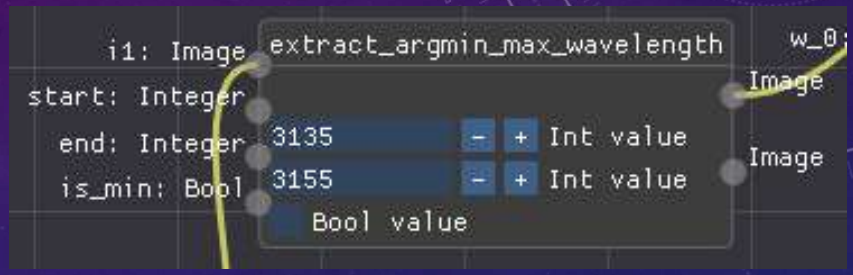
①代表波長の決定
（extract_argmin_max_wavelength ノード,
extract_centrobaric_wavelength ノード）
②速度場マップの生成
（create_velocity_field_map ノード）

※基準波長はユーザ指定

①代表波長の決定

extract_argmin_max_wavelength ノード

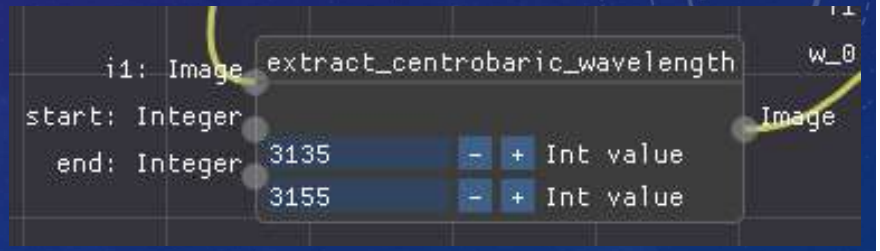
入力: n次元イメージ i
開始波長 $start$
終了波長 end $(start < end)$
最小値か否か is_min



出力: indices and values of $\max_{start\sim end}(flux_k)$ or $\min_{start\sim end}(flux_k)$

extract_centrobaric_wavelength ノード

入力: n次元イメージ i
開始波長 $start$
終了波長 end $(start < end)$



出力: $\frac{\sum_{start}^{end} flux_k \times wave_k}{\sum_{start}^{end} flux_k}$ for all pixels

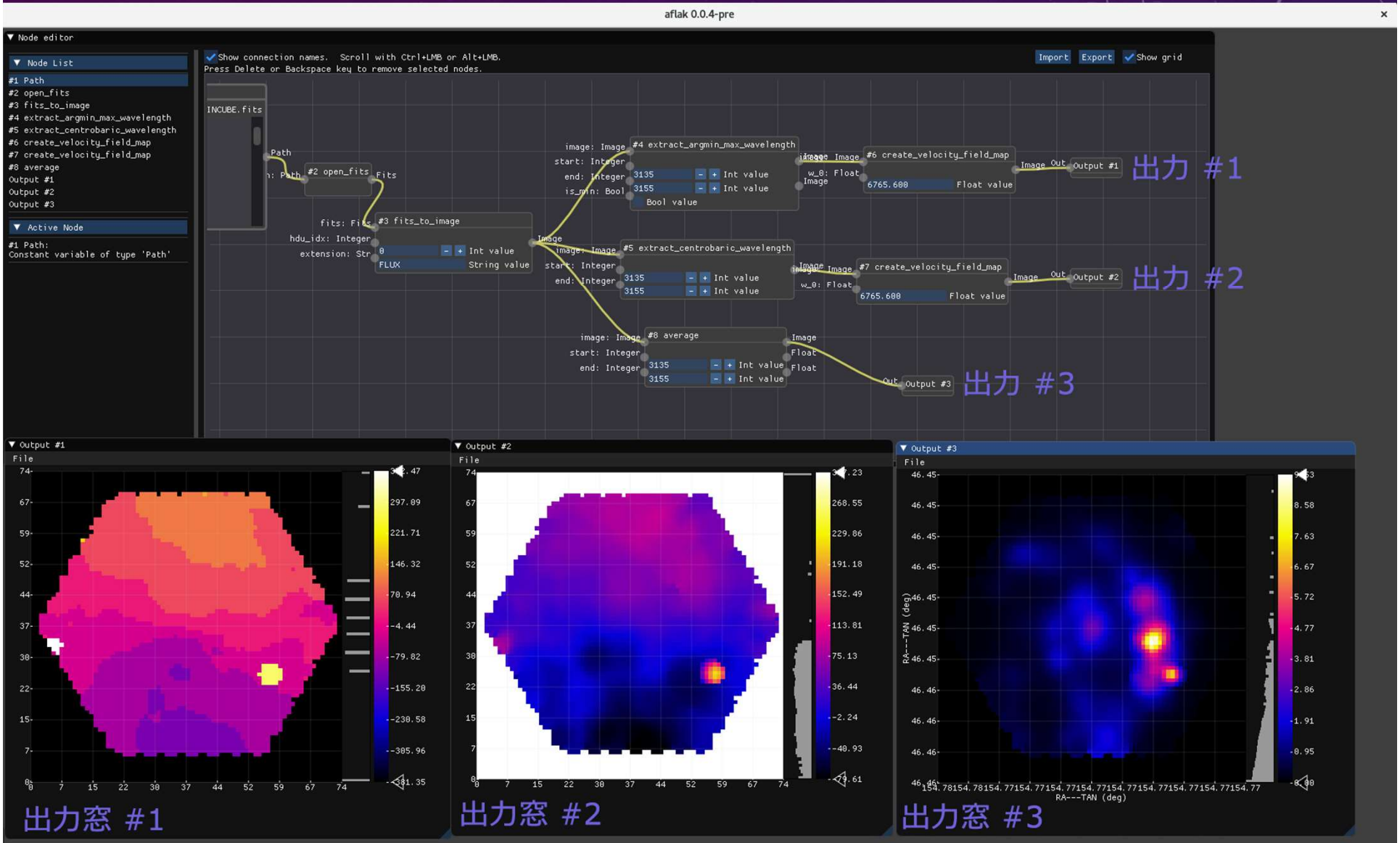
②速度場マップの作成

create_velocity_field_map ノード

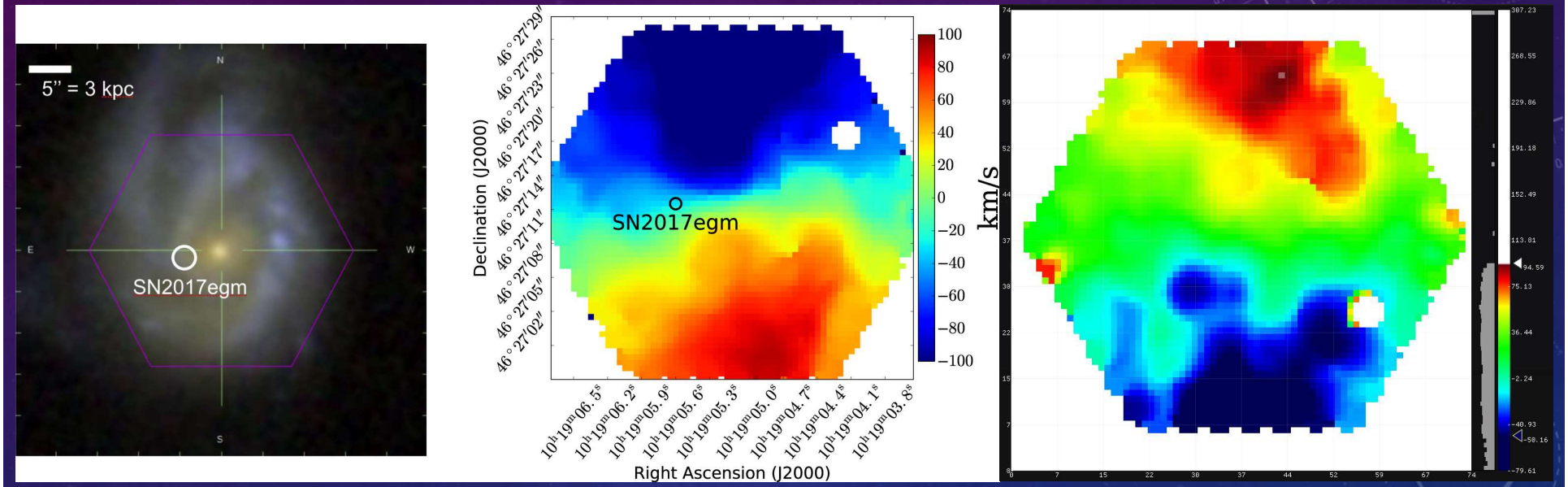
入力: n次元イメージ i
基準波長 w_0

出力: $c \times \frac{i_k - w_0}{w_0}$ for all pixels
$c = 3.0 \times 10^5$ [km/s]

結果（速度場マップ）


aflak 0.0.4-pre
Node editor
Node List
#1 Path
#2 open_fits
#3 fits_to_image
#4 extract_argmin_max_wavelength
#5 extract_centrobaric_wavelength
#6 create_velocity_field_map
#7 create_velocity_field_map
#8 average
Output #1
Output #2
Output #3
Active Node
#1 Path:
Constant variable of type 'Path'
Show connection names. Scroll with Ctrl+LMB or Alt+LMB.
Press Delete or Backspace key to remove selected nodes.
Import
Export
Show grid
INCUBE.fits
出力 #1
出力 #2
出力 #3
Output #1
Output #2
Output #3
File
出力窓 #1
出力窓 #2
出力窓 #3
RA---TAN (deg)

評価（速度場）



Chen, et al.

aflak

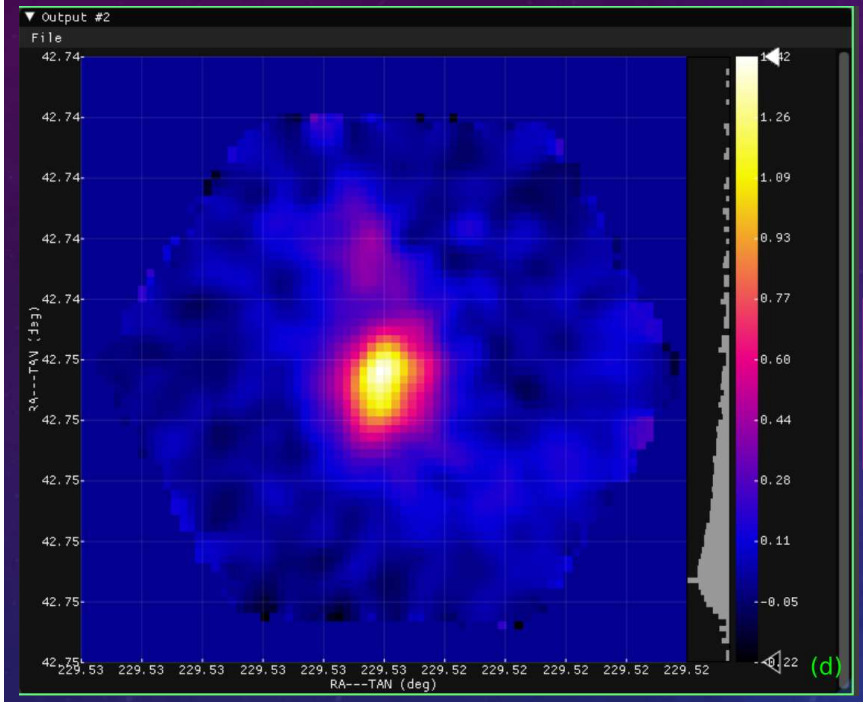
ケーススタディ

三次元面分光データの応用三例

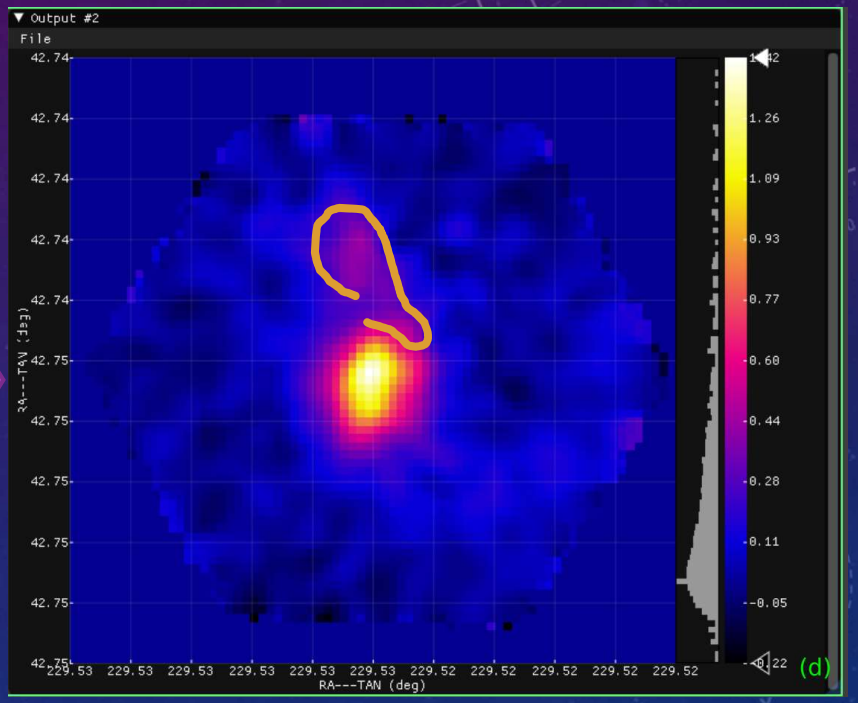
- 等価幅マップの生成
- 速度場マップの生成
- 任意スライシング

任意スライシング

“draw path”オプションを選択



任意の
パスを描く



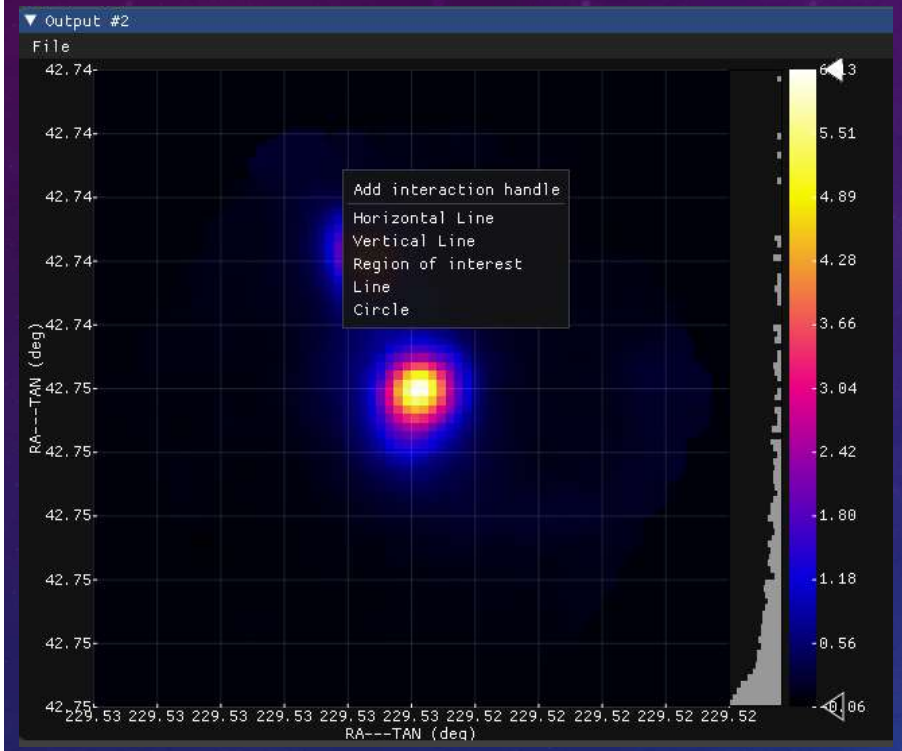
以下のようなノードを使用して，描画された
パスの平面に垂直な軸に沿ってパスの形状をもつ平面を切出し

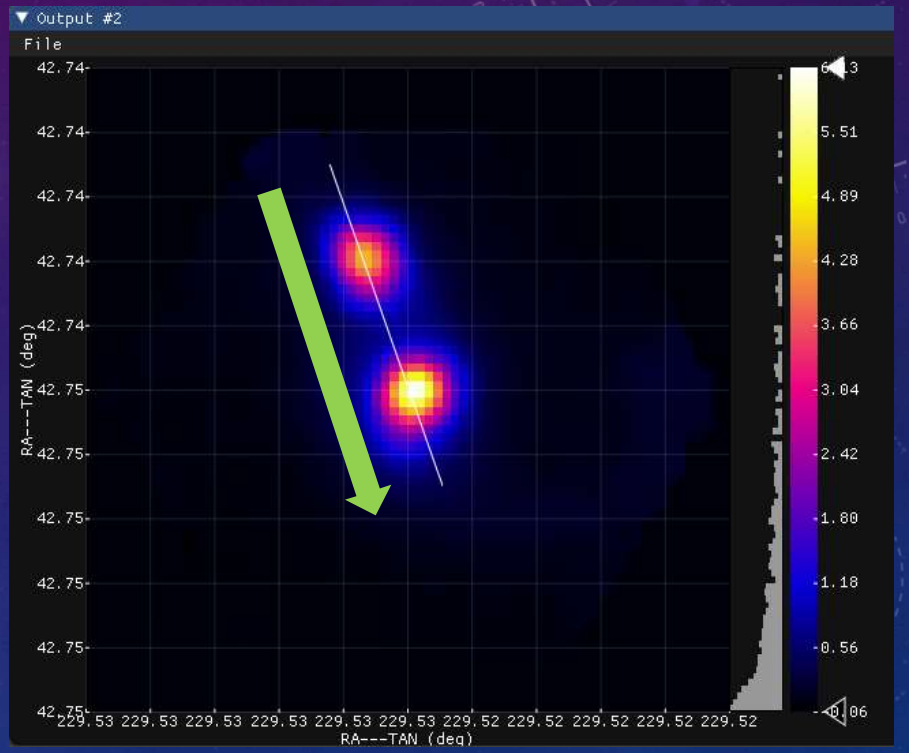
任意スライシングの計算手順

①形状の描画（現在，直線に限定）
（二次元プロット上に Line を描画）
②形状を通る画素の座標を
（ROI: Region Of Interest）として取得
③波長方向への切出し（現在，波長方向に限定）
（extrude ノード）

※波長方向の範囲指定をするノードを別途用意
（range_specification ノード）

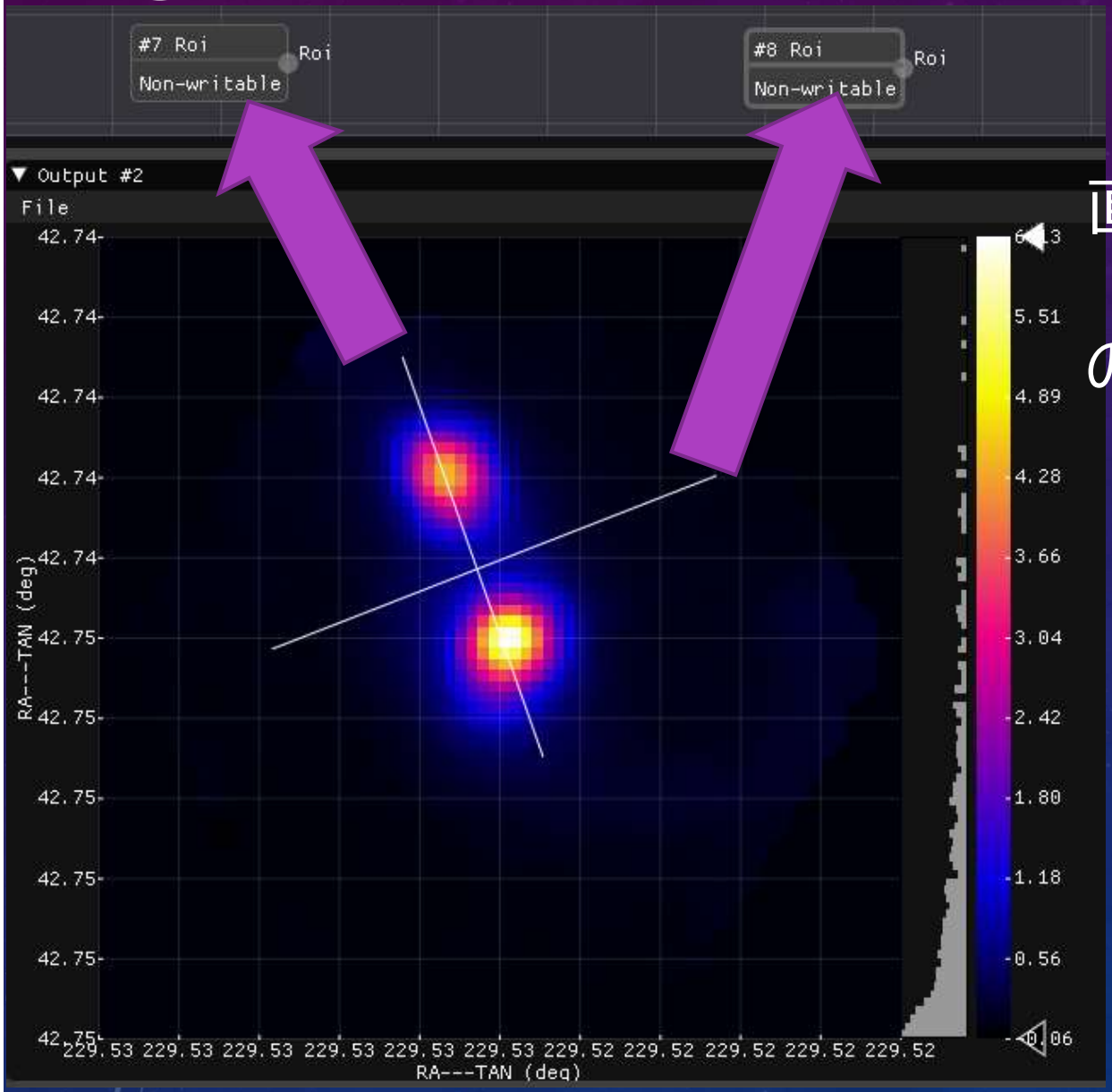
①形状の描画





複数の直線を描画可能

②画素情報の取得



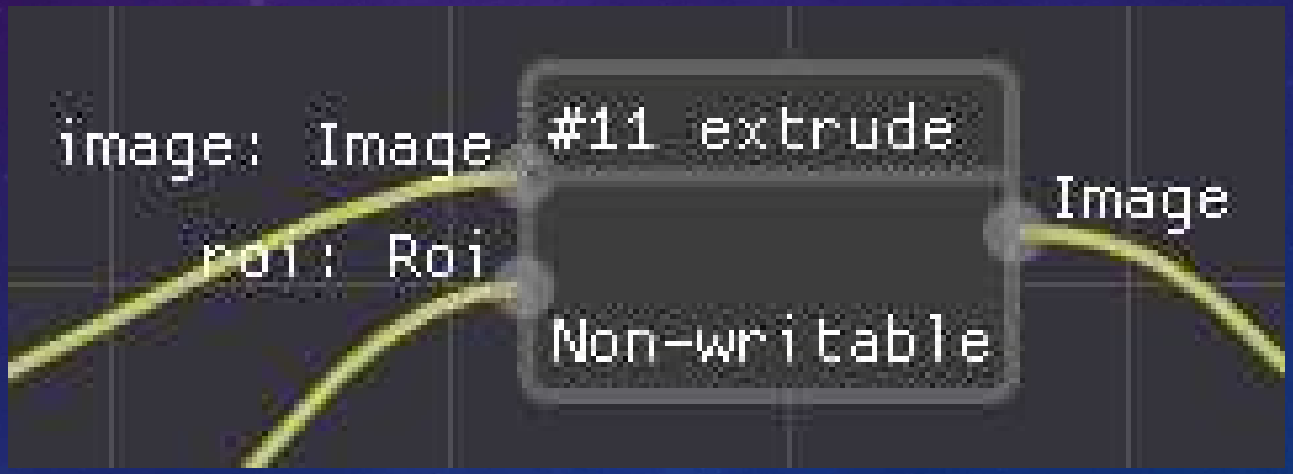
直線上の画素の情報を（座標，フラックス値）のベクタとして保持

③波長方向への切出し

extrude ノード

入力: 三次元イメージ i
画素の情報 roi

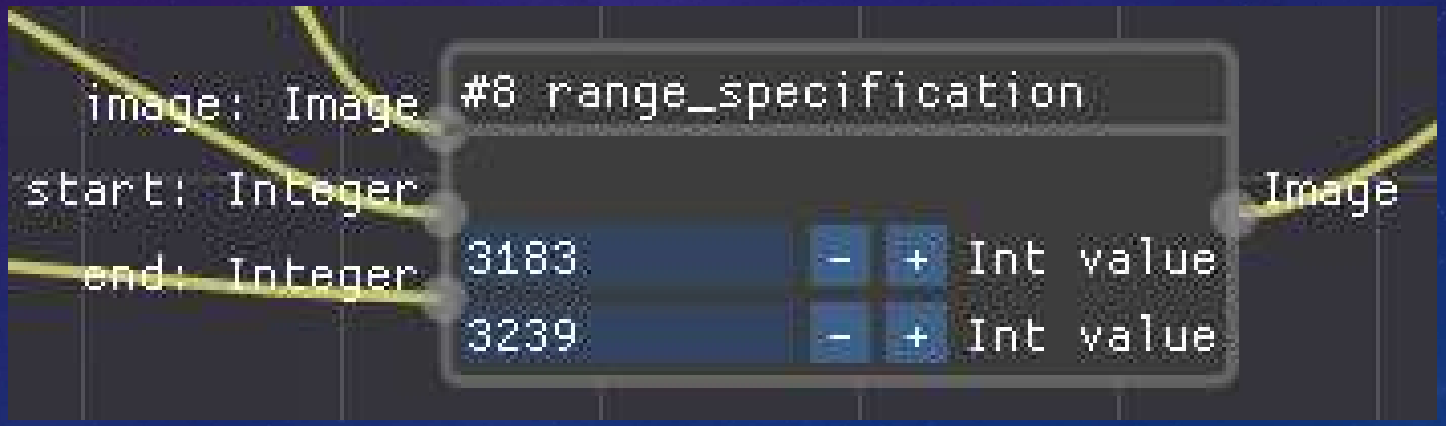
出力: 切り出された二次元イメージ



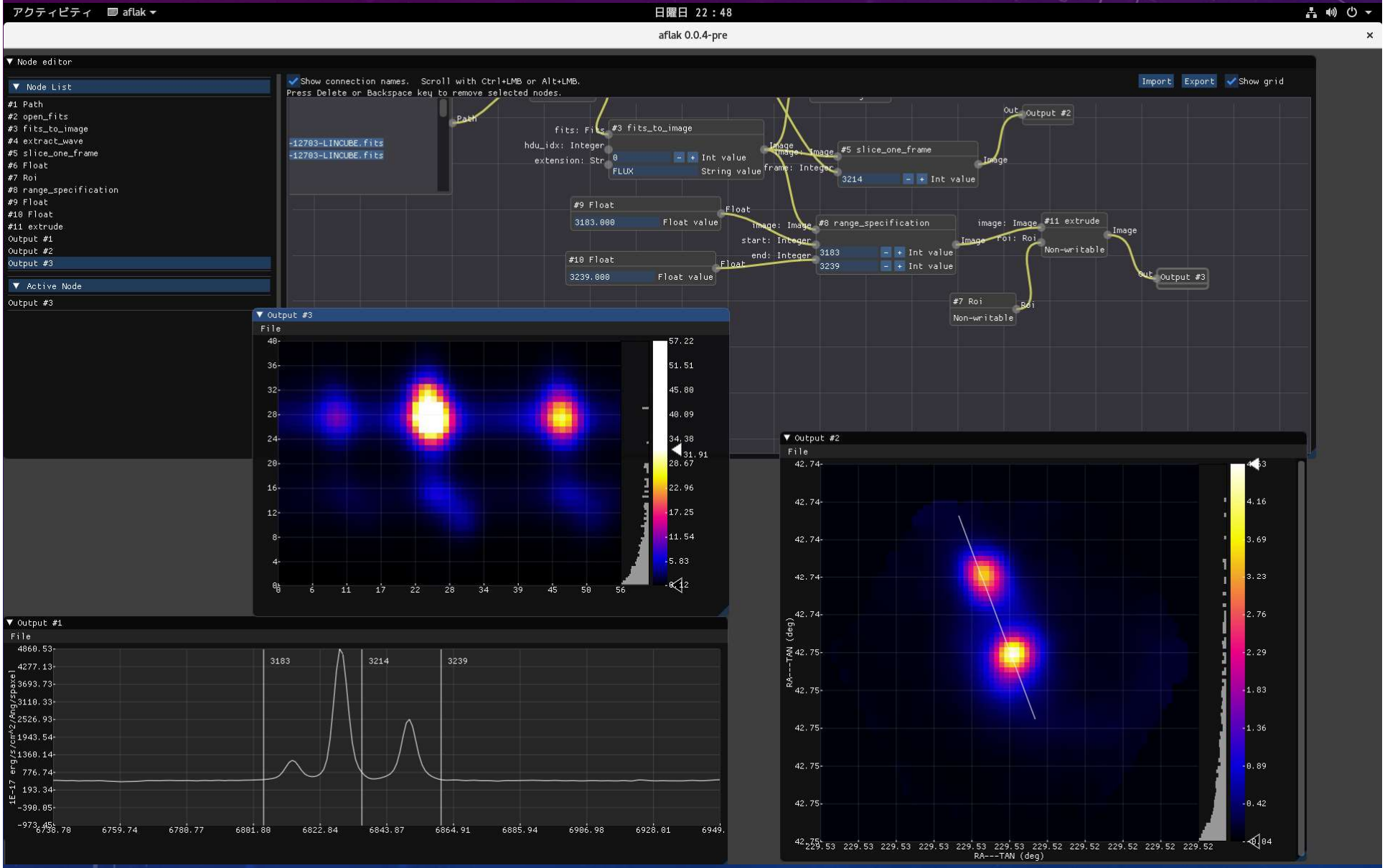
波長範囲の指定

range_specification ノード

入力：n次元イメージ i
　　開始波長 $start$
　　終了波長 end $(start < end)$
出力：$start$ から end までに範囲指定されたイメージデータ



結果（任意スライシング）


アクティビティ
aflak
日曜日 22：48
aflak 0.0.4-pre
Node editor
Node List
#1 Path
#2 open_fits
#3 fits_to_image
#4 extract_wave
#5 slice_one_frame
#6 Float
#7 Roi
#8 range_specification
#9 Float
#10 Float
#11 extrude
Output #1
Output #2
Output #3
Active Node
Output #3
Show connection names. Scroll with Ctrl+LMB or Alt+LMB.
Press Delete or Backspace key to remove selected nodes.
Import
Export
Show grid
Path
-12703-LINCUBE.fits
-12703-LINCUBE.fits
fits: Fits
#3 fits_to_image
hdu_idx: Integer
extension: Str
Int value
FLUX
String value
#5 slice_one_frame
frame: Integer
3214
Out
Output #2
#9 Float
3183.000
Float value
#10 Float
3239.000
#8 range_specification
start: Integer
end: Integer
3183
3239
#11 extrude
Non-writable
#7 Roi
Output #3
File
Output #1
RA---TAN (deg)
1E-17 erg/s/cm^2/Ang/spaxel
3183
3214
3239

現状と課題

主要な貢献

- 多数のパラメタの手動による段階的な微調整が必要な分析過程を支援
- Astronomer-in-the-loopの実現
- End-to-endな出自管理

今後の課題

http://aflak.jp

- https://github.com/aflak-vis/aflak/issues
- バッチ処理
- WCSの完全なサポート
- Aladin / Topcat 等とやりとりするための VO標準のサポート
- Rust 以外の言語によるノードプリミティブ実装 (Python, C等)
- プリミティブノードの網羅性の確立
- 1GB以上のデータセットへの対応

謝 辞

- Aflak研究開発チーム
  - 松林 和也（京都大学）
  - 竹川 俊也（国立天文台）
  - 竹島 由里子（東京工科大）
  - 朱　立宇（慶應義塾大学）
  - 植村 誠（広島大学）
- 科研費
  - 新学術領域計画研究25120014
  - 基盤(A)17H00737
  - 基盤(C)17K00173

公開文献

- M. O. Boussejra, R. Uchiki, S. Takekawa, K. Matsubayashi, Y. Takeshima, M. Uemura, I. Fujishiro: “afak: Visual programming environment with macro support for collaborative and exploratory astronomical analysis,” to appear in *IIEEJ Trans. Image Electronics and Visual Computing*, vol. 7, no. 2, December 2019.
- M. O. Boussejra, R. Uchiki, Y. Takeshima, K. Matsubayashi, S. Takekawa, M. Uemura, I. Fujishiro. “aflak: Visual programming environment enabling end-to-end provenance management for the analysis of astronomical datasets,” *Elsevier Journal of Visual Informatics*, vol. 3, no. 1, pp. 1–8, March 2019 [doi: 10.1016/j.visinf.2019.03.001].
- M. O. Boussejra, K. Matsubayashi, Y. Takeshima, S. Takekawa, R. Uchiki, M. Uemura, I. Fujishiro: “aflak: Visual programming environment with quick feedback loop tuned for multi-spectral astrophysical observations,” presented at *ADASS2018*, College Park, November 2018.
- M. O. Boussejra, K. Matsubayashi, Y. Takeshima, S. Takekawa, R. Uchiki, M. Uemura, I. Fujishiro: “aflak: Pluggable visual programming environment with quick feedback loop tuned for multi-spectral astrophysical observations,” in *Proc. IEEE SciVis 2018*, pp. 72-76, Berlin, October 2019 [doi: 10.1109/SciVis.2018.8823788].

Q&A