OpenCAE富山勉強会
2014/02/08

パラメータ最適化ツール DakotaとOpenMDAOについて

OpenCAE学会

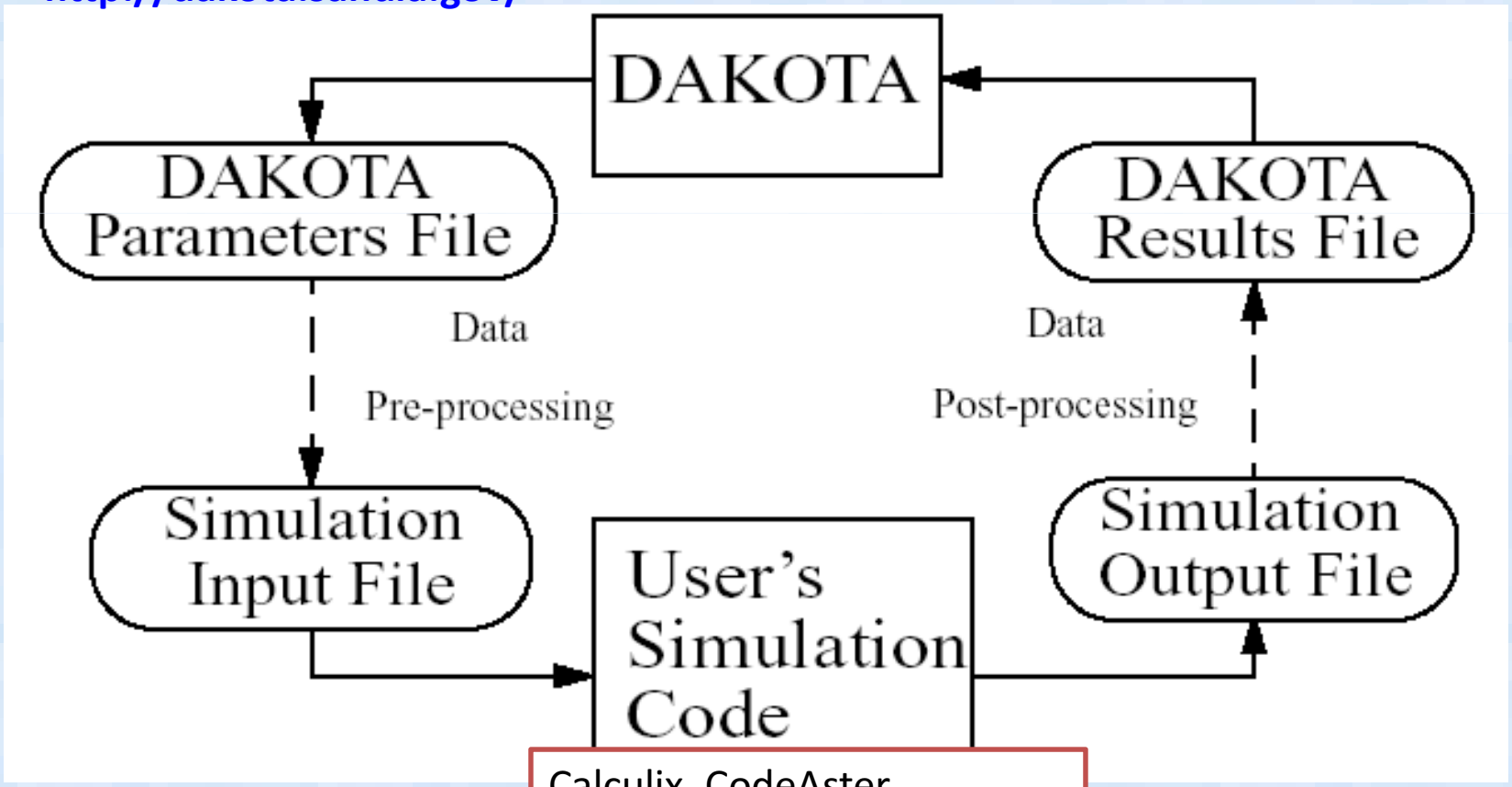
SH

本日の発表内容

- Dakotaとは？
- Dakotav5.4インストール方法など
  -DakotaV5.4の機能追加概要
- Dakotaと各ツールとの連携
  -Calculix連携事例
  -OpenFOAM連携
- DakotaGUIツール：Jaguar　について
  -OpenFOAM連携でJaguarの使用例
- OpenMODについて
- OpenMODインストールメモ
- まとめ

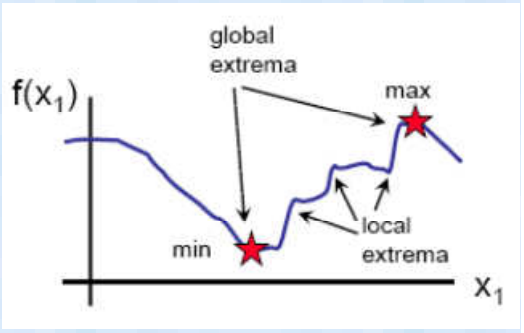
Dakotaとは①？

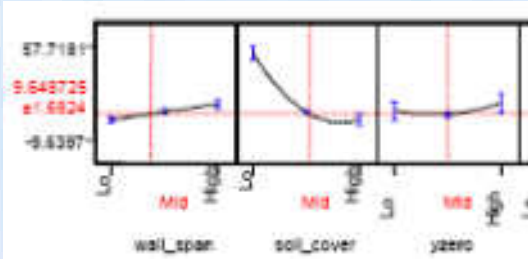
マニュアルのp.20に書かれているように自動的に解析に入力するパラメータを変更したパラメータSTUDYや目的となる指標（目的関数）を与えると自動的にそのような目的の値になるようなパラメータを見つけてくれるツールです。
一般的には総称して最適化ツールとか呼ばれています。
-ParaView と同じ　SandiaNational Laboで開発されている
- http://dakota.sandia.gov/

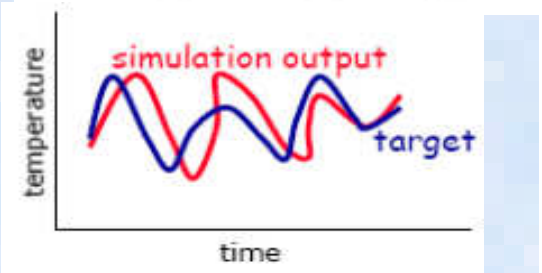


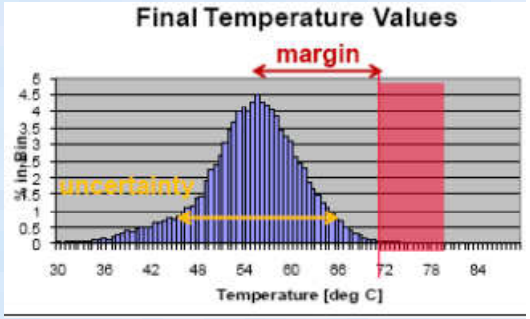
Dakotaとは②？

- **DAKOTAは具体的には何に使えるのか？**
  - Optimization **設計最適化:**
    最適寸法や最適パラメータ決定
  - Sensitivity Analysis **感度解析:**
    入力パラメータ変動に対する出力感度
  - Calibration **パラメータ同定:**
    未知パラメータ決定(物性値同定など)
  - Uncertainty Quantification **信頼度：**
    入力パラメータが変動した場合の応答変動の確率分布推定









Dakotav5.4 の機能変更概要

- 信頼性評価機能の追加： New PoFDarts method
- 最適化: 直接サーチ整数最適化手法
  New direct search mixed integer optimization method (NOMAD)
  extended Bayesian calibration methods and handling of experimental data(QUESO)
- Boostパッケージ1.49が必要

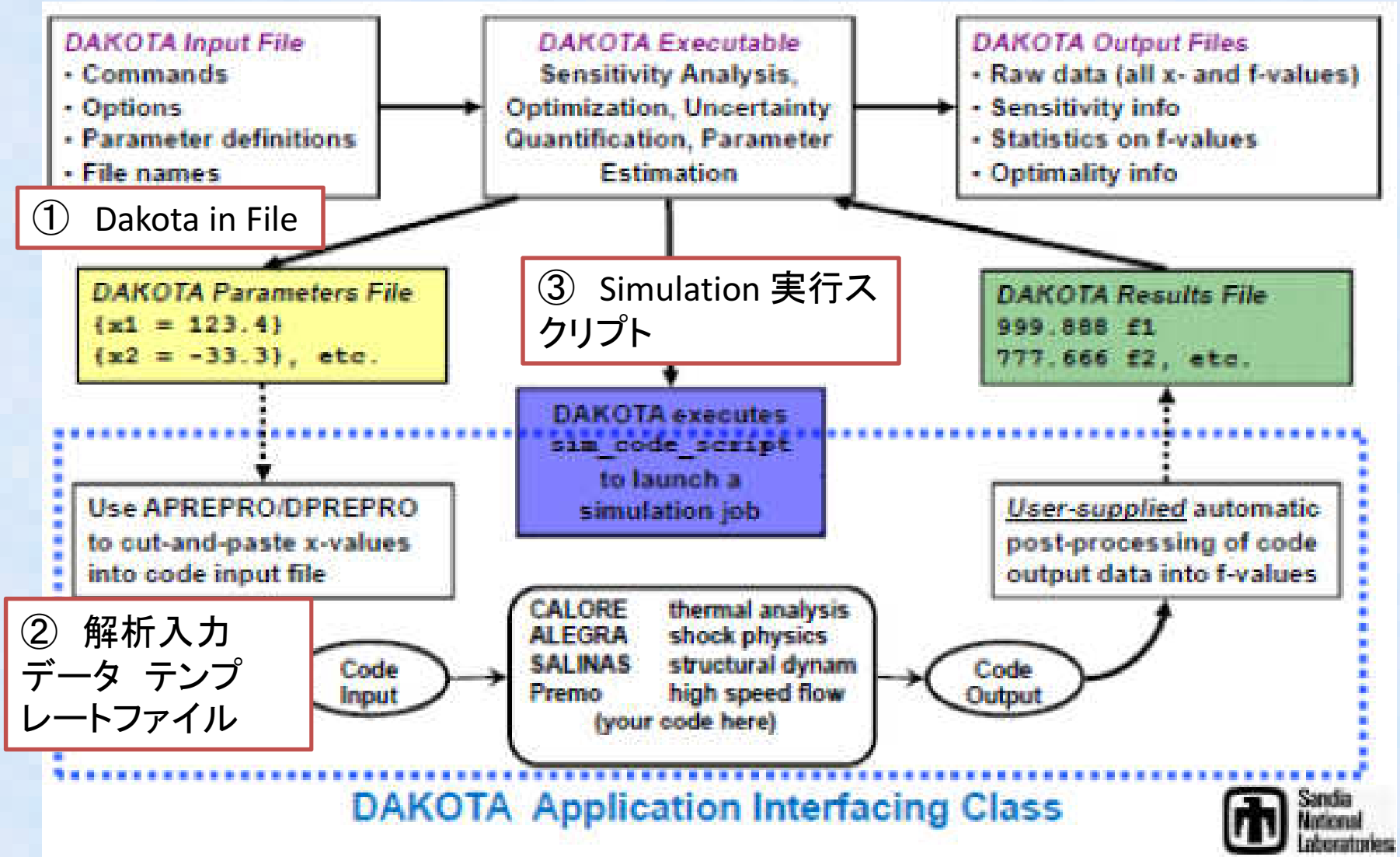
V5.4 についてそれほど大きな変更はなし

Dakotaと各ツールとの連携①

- Dakotaは任意のプログラムや解析ツールと連成する場合、スクリプトなど外部インターフェースを使って自動実行させる。
- 幾つかのスクリプトやbinary 連携ツールがexamples Directory　以下に提供されているのでこれを使って連携実行させる compiled interface 以下にはmatlab 以外はつかえそうもなくscript_interfacesの下のサンプルを利用することにした(V5.3～Matlab以外にOSS scalibの直接interface が準備された)。
- ABAQUS, NASTRANなど解析ツールの他、Python script, Excel Visual Basic script と　連携する例のsample がある。

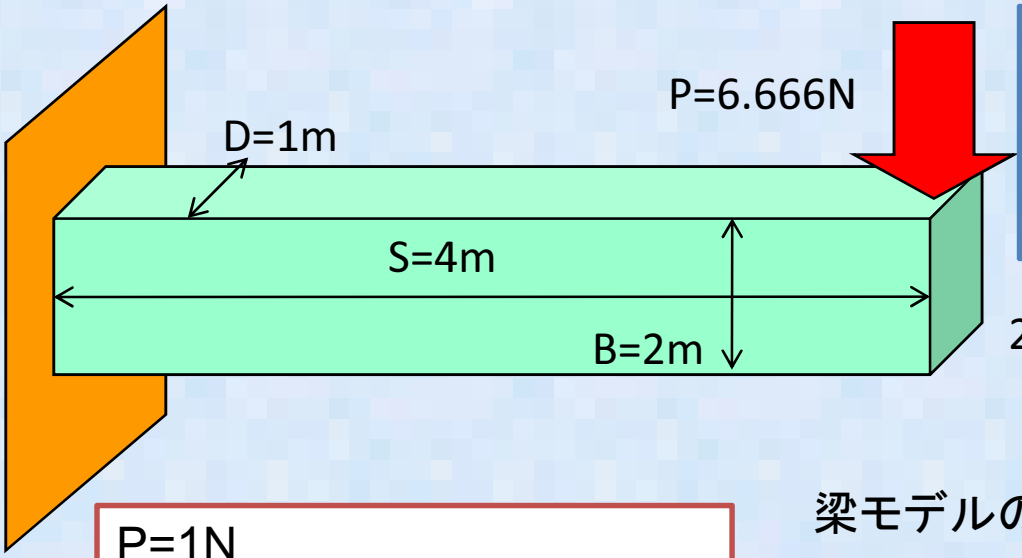
Dakota と各ツールとの連携②

- Dakotaと解析ツールは下記のように連携ユーザは以下3つファイルを準備する必要あり。



Dakota と各ツールとの連携③

- **片持ちはりの反り計算の例題でABAQUSのSample が入っていたので、これをOSS有限要素解析ソフト　Calculix　で動くように書き換えて実行してみる**
- **前提:Linux 環境 (Ubuntsu)　Calculix linux版　install 済み**
- **Calculix V2.1 を使用**



設計変数: S, B, P
S=2-6m, B=1-3m, P=4-10N
目的関数: 梁の最大応力
σ→最小化

2次元平面応力要素を利用

P=1N
S= 4m
B= 2m
D= 1m
E=30e3Pa
I= $bt^3/12$

梁モデルの最大反り理論解

$$\delta = \frac{WL^3}{3EI}$$

M=PS (モーメント)

σ = M/Z =PS/Z
= PS/(DB^2/6)
(最大応力)

Dakota と各ツールとの連携④

- ①dakota_abaq_opt.in の中身を書き換える

```
strategy,
        single_method graphics
method,
        dot_mmfd
variables,
        continuous_design = 3
         cdv_initial_point =  4.0  2.0  6.66
         cdv_lower_bounds  =  2.0  1.0  4.0
         cdv_upper_bounds  =  6.0  3.0  10.0
         cdv_descriptor    =  'S' 'B' 'P'
interface,
        application system #asynch evaluation_concurrency = 5
         analysis_driver = 'abaq_driver'
         parameters_file = 'params.in'
         results_file = 'results.out'
         aprepro
responses,
        num_objective_functions = 1
        numerical_gradients
         method_source dakota
         interva
         fd_step
        no_hess
```

今はこのモジュールなし。とりあえず下記入れ替え

```
method,
    conmin_frcg
      max_iterations = 100
      convergence_tolerance = 1e-4
```

Application は現在認識しないので　消す system だけ残す

Dakotaに今入っていない
Internet から　Source Download しinstall
aprepro-2.01.tar.gz
(他のVersionはmake に失敗)

Deprepro(dakotaに入っている Perl script file)にて置き換えることが可能

Dakotaと各ツールとの連携⑤

- ②fe.inp.appの書き換え(fe.inp.appはCalculix（abaqus）の解析入力ファイルのテンプレート

```
{ECHO(OFF)}
{include(params.in)}
{ECHO(ON)}
*HEADING
*NODE
    1,        0.0,        0.0
    2,      {S/4},        0.0
    3,      {S/2},        0.0
    4,        0.0,      {B/2}
    5,      {S/4},      {B/2}
    6,      {S/2},      {B/2}
    7,        0.0,        {B}
    8,      {S/4},        {B}
    9,      {S/2},        {B}
*ELEMENT,TYPE=CPS8,ELSET=BEAM
 1, 1, 3, 9, 7, 2, 6, 8, 4
**ELEMENT,TYPE=CPS4,ELSET=BEAM
** 1, 1, 2, 5, 4
** 2, 2, 3, 6, 5
** 3, 4, 5, 8, 7
** 4, 5, 6, 9, 8
*SOLID SECTION,ELSET=BEAM,MATERIAL=AL2024T3
1.0
*MATERIAL,NAME=AL2024T3
*ELASTIC
30.000000e3,     0.3
*BOUNDARY
 1, 2, , 0.0
 3, 1, , 0.0
 6, 1, , 0.0
 9, 1, , 0.0
*STEP
*STATIC
*CLOAD
** 9, 2,   3.3333333,
 9, 2,   {P/2},
*EL PRINT, POSITION=AVERAGED AT NODES, ELSET=BEAM
S
*END STEP
```

S, B, P はDakotaとAPREPROで変更する
Parameter　{S} , {B}, {P} のように
中括弧でパラメータを指定すると
Aprepro(or Deprepro) がパラメータを自動で置き換える

Calculix　V2.1 ではCPS4 が無いのでCPS8に変更した

Calculix　ではELSET 指定が必要

Dakotaと各ツールとの連携⑥

- ③ Abaqus_driver(解析自動実行スクリプト) を編集

```
#!/bin/csh -f

# $argv[1] is params.in FROM Dakota
# $argv[2] is results.out returned to Dakota

# Workdir setup for running in parallel (file_tag option turned on)
# In this simple case, all templatedir would contain is fe.inp.app
#set num = `echo $argv[1] | cut -c 11-`
#cp -r templatedir workdir.$num
#mv $argv[1] workdir.$num/params.in
#cd workdir.$num

# Pre-processing

aprepro --nowarning -q fe.inp.app fe.inp

# Run ABAQUS

#rm -f *.dat *.sta
/usr/bin/ccx_2.1 -i fe
####>>&! abaqus.out

# Post-processing

grep '    1  27' fe.dat | head -n 1 | awk '{print $3}' > $argv[2]

# Results file move :and workdir cleanup for running in parallel
#mv $argv[2] ../.
#cd ..
#rm -rf workdir.$num
```

このScriptはcsh 向けに書かれているため、bsh 系は書式を少し変える必要あり

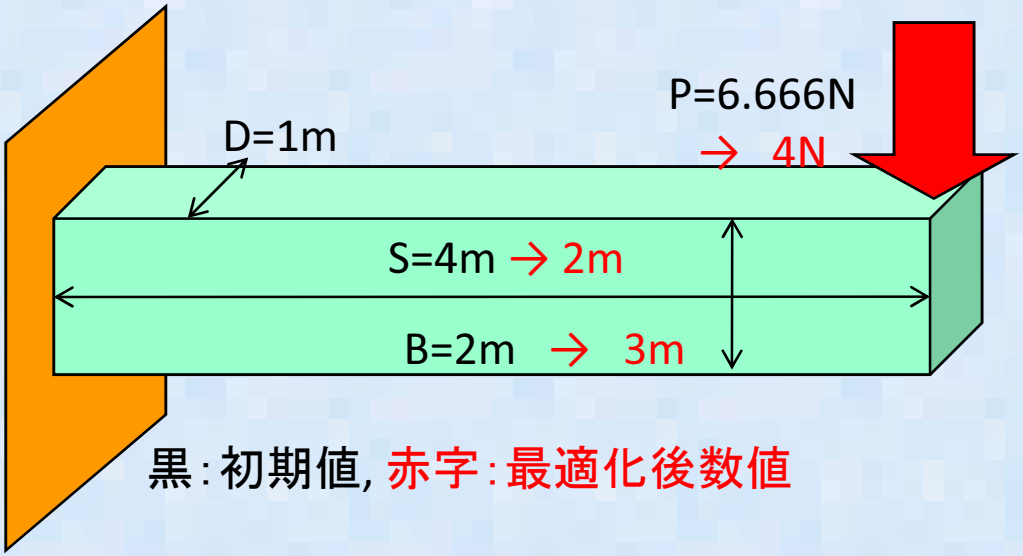
このｺﾏﾝﾄﾞでaprepro が各パラメータ{S}などを実際の数字に置き換えたファイルを作成する

ここをCalculix 用に編集

ここをCalculix 用に編集

Dakotaと各ツールとの連携⑦

- 最適化計算結果



黒：初期値, 赤字：最適化後数値

グラフ作成出力

%eval_id	S	B	P	obj_fn
1	4	2	6.66667	6.8376
2	3.977324	2.090318	6.653404	6.1956
3	3.886618	2.451589	6.600339	4.2731
4	3.763126	2.94344	6.528094	2.6934
5	3.741134	3	6.517772	2.5533
6	3.65915	3	6.479296	2.4655
7	3.331218	3	6.32539	2.123
8	2	3	5.70062	0.96106
9	2	3	4.560425	0.76884
10	2	3	4	0.67436
11	2	3	4	0.67436
12	2	3	4	0.67436
13	2	3	4	0.67436

Dakota テキスト出力結果

```
FINAL OPTIMIZATION INFORMATION     OBJ =   0.674360E+00
DECISION VARIABLES (X-VECTOR)      1)    0.20000E+01  0.30000E+01  0.40000E+01
```

梁の長さを短く　厚さ(高さ)を大きく、荷重を小さくすれば応力が最小になるという、常識的な結果が得られる。

Dakotaと各ツールとの連携⑧

- 次にOpenFOAMとの連携を検討。例題は何でも良かったのだが、TutorialのCavityFlowの例題を実施

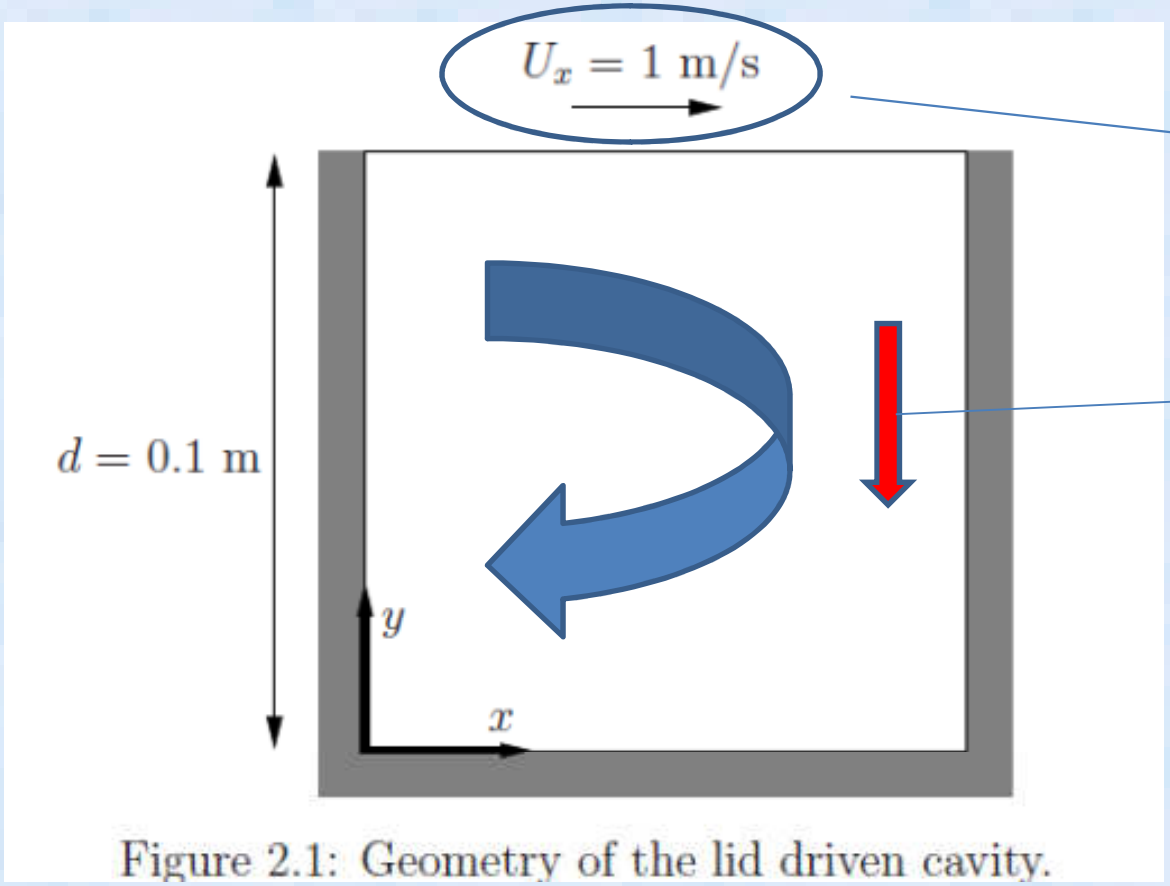


Figure 2.1: Geometry of the lid driven cavity.

駆動速度:U
をパラメータ〈設計変数)

Y 方向のマイナス方向速度
の最大値を目的関数とする。

Dakotaと各ツールとの連携⑨

- Abaqus(calculix)のスクリプトインターフェース例題をベースにファイルを編集する。編集するファイルは以下の3つ

  - Dakotaの入力ファイル(dakota_abaq_opt.in)
  - ソルバインターフェーススクリプト (abaq_driver)
  - OpenFOAMの入力ファイルをベーステンプレートファイルとして編集。
  (今回は境界速度をパラメータにしているので、0/Uをテンプレートとして編集する)

Dakota と各ツールとの連携⑩

- Dakota の入力ファイル(dakota_abaq_opt.in)

```
strategy,
                single_method
      graphics,tabular_graphics_data
method,
      conmin_frcg
       max_iterations = 100
       convergence_tolerance = 1e-4

#method,
#dot_mmfd

variables,
                continuous_design = 1
                 cdv_initial_point =  1.0
                 cdv_lower_bounds  =  0.5
                 cdv_upper_bounds  =  2.0
                 cdv_descriptor    =  'U'

interface,
                system #asynch evaluation_concurrency = 5
                 analysis_driver = 'abaq_driver'
                 parameters_file = 'params.in'
                 results_file = 'results.out'
                 aprepro

responses,
                num_objective_functions = 1
                numerical_gradients
                 method_source dakota
                 interval_type central
                 fd_step_size = .01
                no_hessians
```

グラフデータを出力するように指定

設計変数は1つU
初期値は1
最小値は0.5
最大値は2.0

ソルバーを実行する
スクリプトファイルを指定

Dakotaと各ツールとの連携⑪

- ソルバインターフェーススクリプト(abaq_driver)

```
#!/bin/csh –f
# $argv[1] is params.in FROM Dakota
# $argv[2] is results.out returned to Dakota

mkdir work
cp -r ./0 ./work/.
cp -r ./constant ./work/.
cp -r ./system ./work/.
mv $argv[1] ./work/0/params.in
cd ./work/0

# Pre-processing
dprepro params.in U.temp U
#cat U-1 U-2 U-3 > U

#run OpenFOAM
cd ..
blockMesh
icoFoam

# Post-processing
cd 0.5
more +381 U | head -n 1 | awk '{print $2}' > $argv[2]

# Results file move :and workdir cleanup for running in parallel
mv $argv[2] ../.
cd ..
mv $argv[2] ../.
cd ..
rm -r ./work
```

Work directory にOpenFOAMの入力ファイルをコピーする

Dakota のプリプログラム **Deprepro** にて OpenFOAM速度ファイルU の一部を自動で入れ替える設定をする(Calculix の時使ったAprepro はOpenFOAMでは正常に動かなかった)

```
type          fixedValue;
value         uniform ({U} 0 0);
```

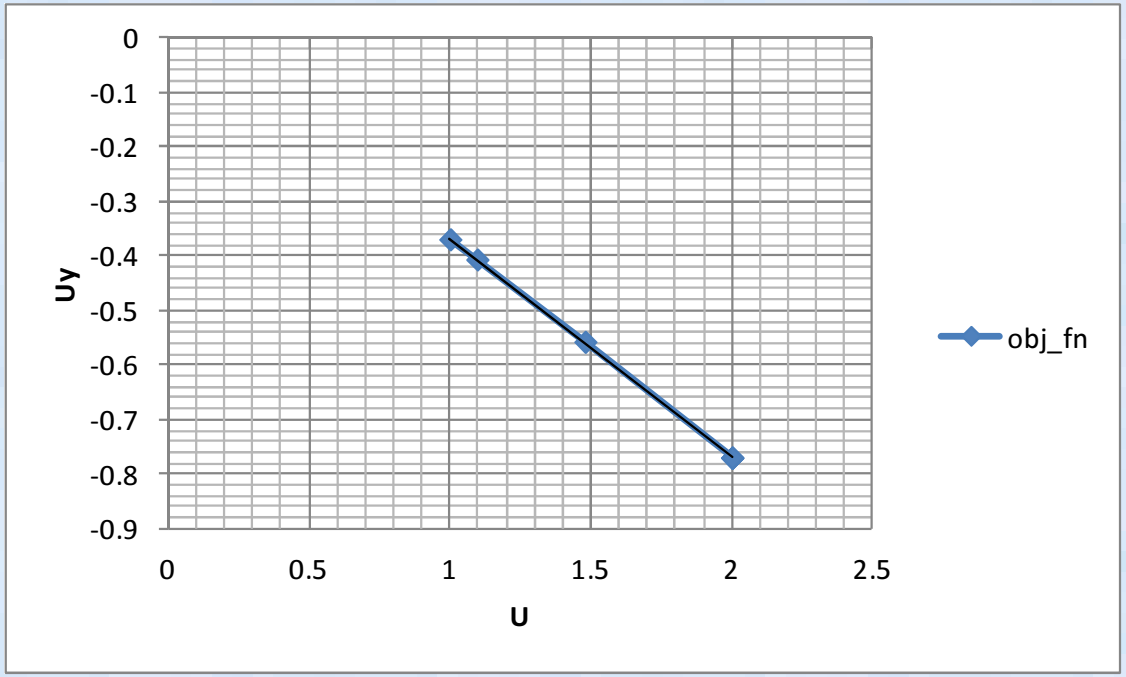
```
type          fixedValue;
value         uniform (1 0 0);
```

Dakotaと各ツールとの連携⑫

- 計算結果

```
FINAL OPTIMIZATION INFORMATION
OBJ =  -0.769003E+00
DECISION VARIABLES (X-VECTOR)
     1)    0.20000E+01
```

境界速度を　2m/s で内部Y方向流速の最小値(最大値)は-0.769m/s　が得られる。



%eval_id	U	obj_fn
1	1	−0.36861
2	1.095793	−0.40562
3	1.478966	−0.55656
4	2	−0.769
5	1.999999	−0.769
6	1.999999	−0.769
7	2	−0.769

Dakotaと各ツールとの連携⑬

- 同様にCavityの問題でX,Y寸法をパラメータにして計算させてみる。

とりあえず最適化ではなく単純なパラメータSTUDYを実施してみた

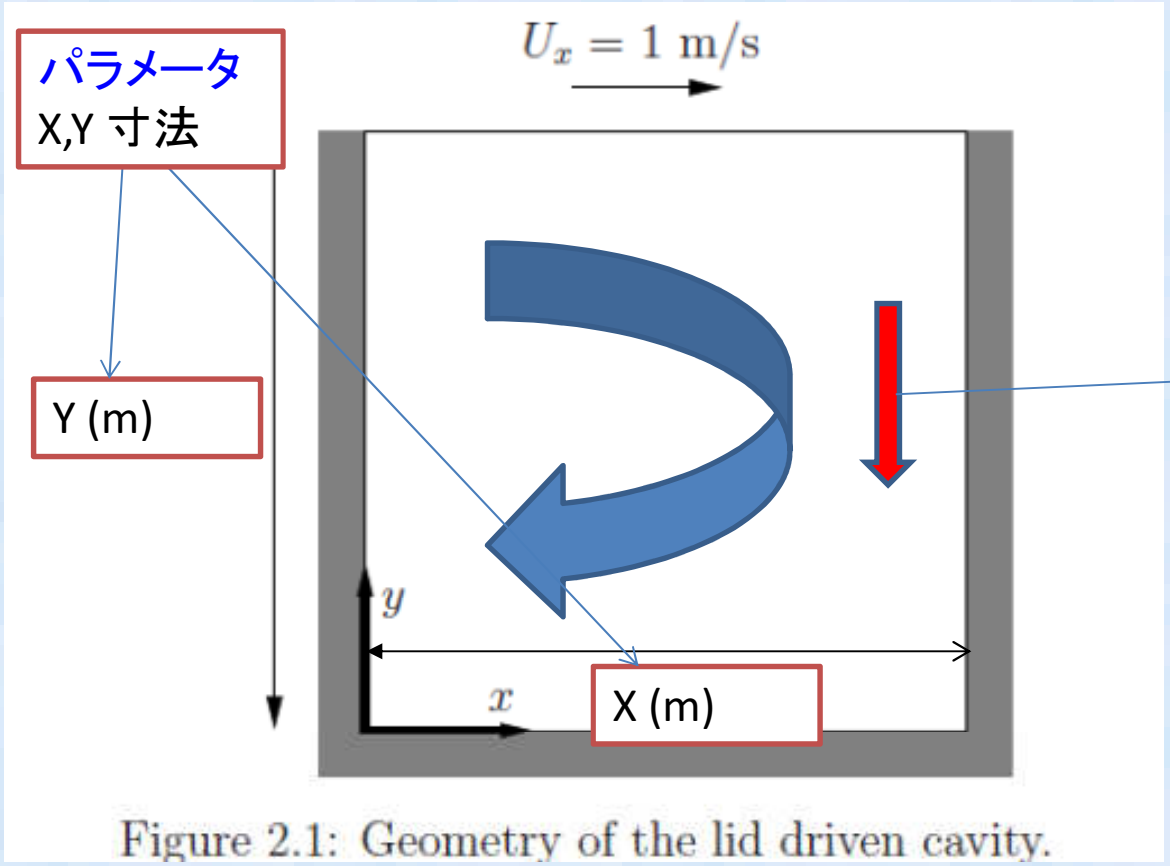


Figure 2.1: Geometry of the lid driven cavity.

Y方向のマイナス方向速度の最大値を目的関数とする。

Dakotaと各ツールとの連携⑭

Dakotaの入力ファイル(dakota_abaq_opt.in) ファイル中身

```
strategy
   graphics
   tabular_graphics_data
   single_method
method
       multidim_parameter_study
      partitions = 4 4
variables
continuous_design = 2
        lower_bounds    0.5     0.5
        upper_bounds    2.0     2.0
        descriptors     'X'     'Y'
interface
   analysis_drivers 'abaq_driver'
      system #asynch evaluation_concurrency = 5
         parameters_file 'params.in'
         results_file 'results.out'
         aprepro
responses
   objective_functions 1
   numerical_gradients
      method_source
      dakota
      interval_type
      central
      fd_step_size 0.01
   no_hessians
```

Methodをパラメータ STUDYに切り替える
4 4 はX, Y 最大最小値間隔をそれぞれ4分割して変化させる意味

Parameter はX,Y 寸法で最小値は0.5, 最大値は2とした

X(m)
0.5
0.875
1.25
1.625
2

×

Y(m)
0.5
0.875
1.25
1.625
2

5×5 =25 ケース のパラメータstudy を実施する

multidim_parameter_study
は総当たりで解析を実施するので次元数が大きくなると解析ケース数が膨大となるので注意が必要

Dakotaと各ツールとの連携⑮

- ソルバインターフェーススクリプト(abaq_driver)

```
#!/bin/bash –f
mkdir work
cp -r ./0 ./work/.
cp -r ./constant ./work/.
cp -r ./system ./work/.
mv params.in ./work/constant/polyMesh/params.in
cd ./work/constant/polyMesh

# Pre-processing
dprepro params.in blockMeshDict.temp blockMeshDict

#run OpenFOAM
cd ..
cd ..
blockMesh
icoFoam

# Post-processing
cd 0.5
more +381 U | head -n 1 | awk '{print $2}' > results.out

# Results file move :and workdir cleanup for running in parallel
mv results.out ../.
cd ..
mv results.out ../.
cd ..
rm -r ./work
```

./work に計算フォルダを作成し、必要なファイルをコピーする

Dakota .パラメータファイル(params.in)を ./work/constant/polyMesh にコピーする

blockMesh のテンプレートをdeprepro にてパラメータ数字に置き換える

OpenFOAM実行

最終計算時間0.5 フォルダの流速Uの381行目にU の最大値が 出ているのでそれを拾ってきて Dakota の結果ファイル（results.out)に渡す

./work 以下の作業フォルダから結果ファイルをDakota 実行フォルダに回収し、作業フォルダ./workを消す

Dakota と各ツールとの連携⑯

blockMeshDict の template file 中身

```
vertices
(
    (0 0 0)
    ({X} 0 0)
    ({X} {Y} 0)
    (0 {Y} 0)
    (0 0 0.1)
    ({X} 0 0.1)
    ({X} {Y} 0.1)
    (0 {Y} 0.1)
);
```

パラメータ変更させたいXY座標値を{X}, {Y} の記号に置き換える

実行コマンド
% dakota -i dakota_abaq_opt.in

Dakota コマンド実行 Directory

-Dakota input file: dakota_abaq_opt.in
-OpenFOAM 実行 Script : abaq_driver (Bash script)

- OpenFOAM 入力file 元Directory :cavity
  - 0
  - Constant
    - polyMesh
  - system

-blockMeshDictのテンプレート
blockMeshDict.temp

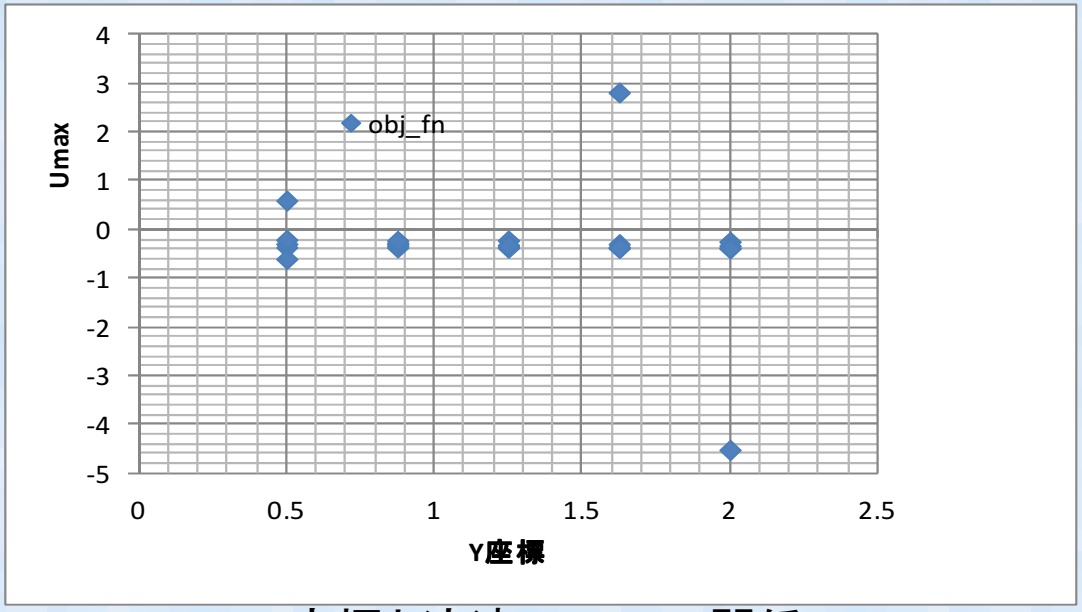
計算時にコピー

Work Directory
- 0
- Constant
  - polyMesh
- system

Dakota と各ツールとの連携⑰

- しばらく計算すると以下の結果が得られる。

%eval_id	X	Y	obj_fn
1	0.5	0.5	−0.360726
2	0.875	0.5	−0.290734
3	1.25	0.5	−0.206985
4	1.625	0.5	−0.60085
5	2	0.5	0.596946
6	0.5	0.875	−0.300514
7	0.875	0.875	−0.366666
8	1.25	0.875	−0.335465
9	1.625	0.875	−0.279466
10	2	0.875	−0.225393
11	0.5	1.25	−0.219457
12	0.875	1.25	−0.342922
13	1.25	1.25	−0.372454
14	1.625	1.25	−0.354241
15	2	1.25	−0.317243
16	0.5	1.625	2.81395
17	0.875	1.625	−0.295194
18	1.25	1.625	−0.361606
19	1.625	1.625	−0.378085
20	2	1.625	−0.36565
21	0.5	2	−4.51749
22	0.875	2	−0.245306
23	1.25	2	−0.33235
24	1.625	2	−0.373322
25	2	2	−0.383536



Y座標と流速Uymax の関係

Y寸法と最大流速はあまり相関がみられないようだ。

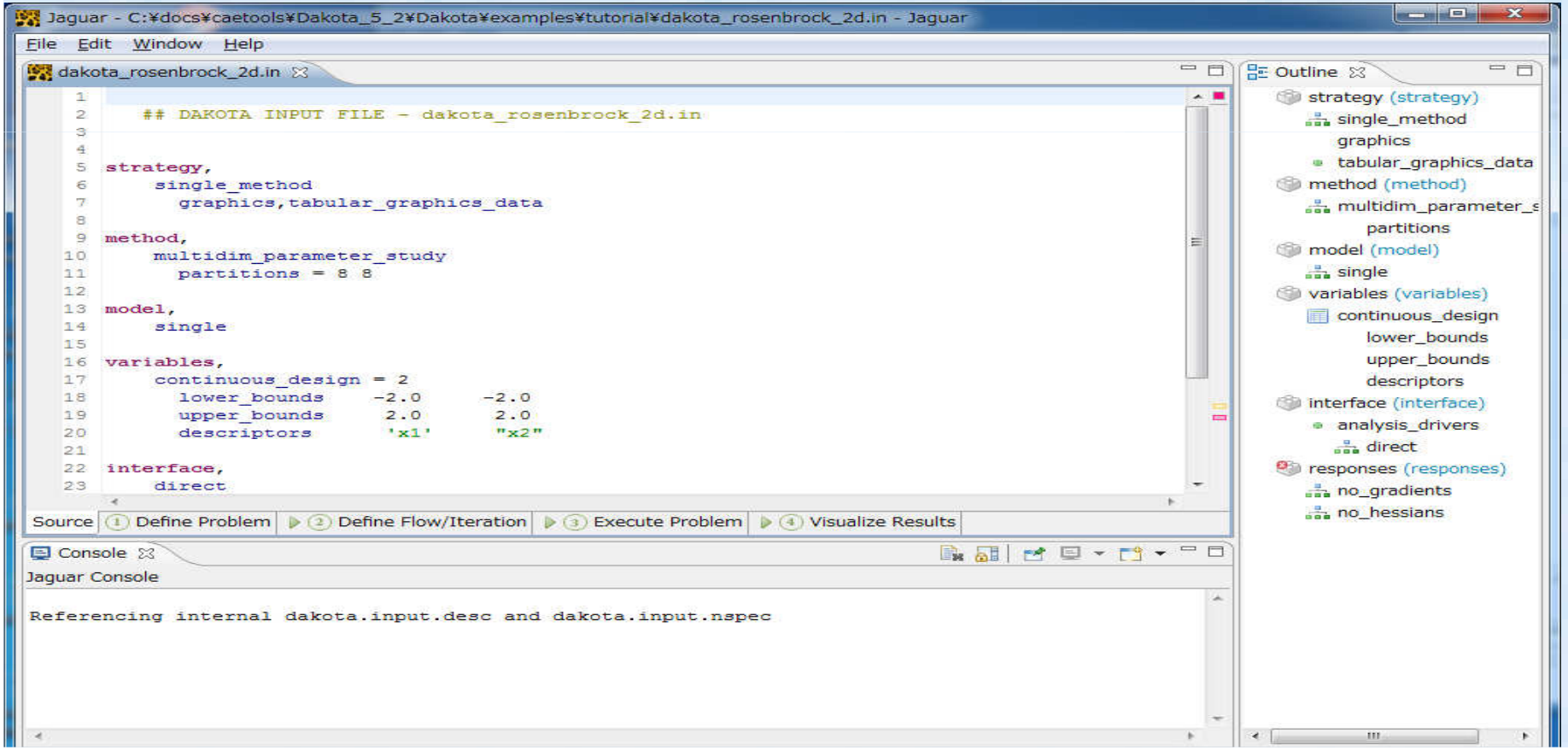
GUIツール：Jaguar について

-Jaguar とは：Dakota 専用のGUI tool

-JAVAにて書かれている

ある程度 input file 作成をSupportする。

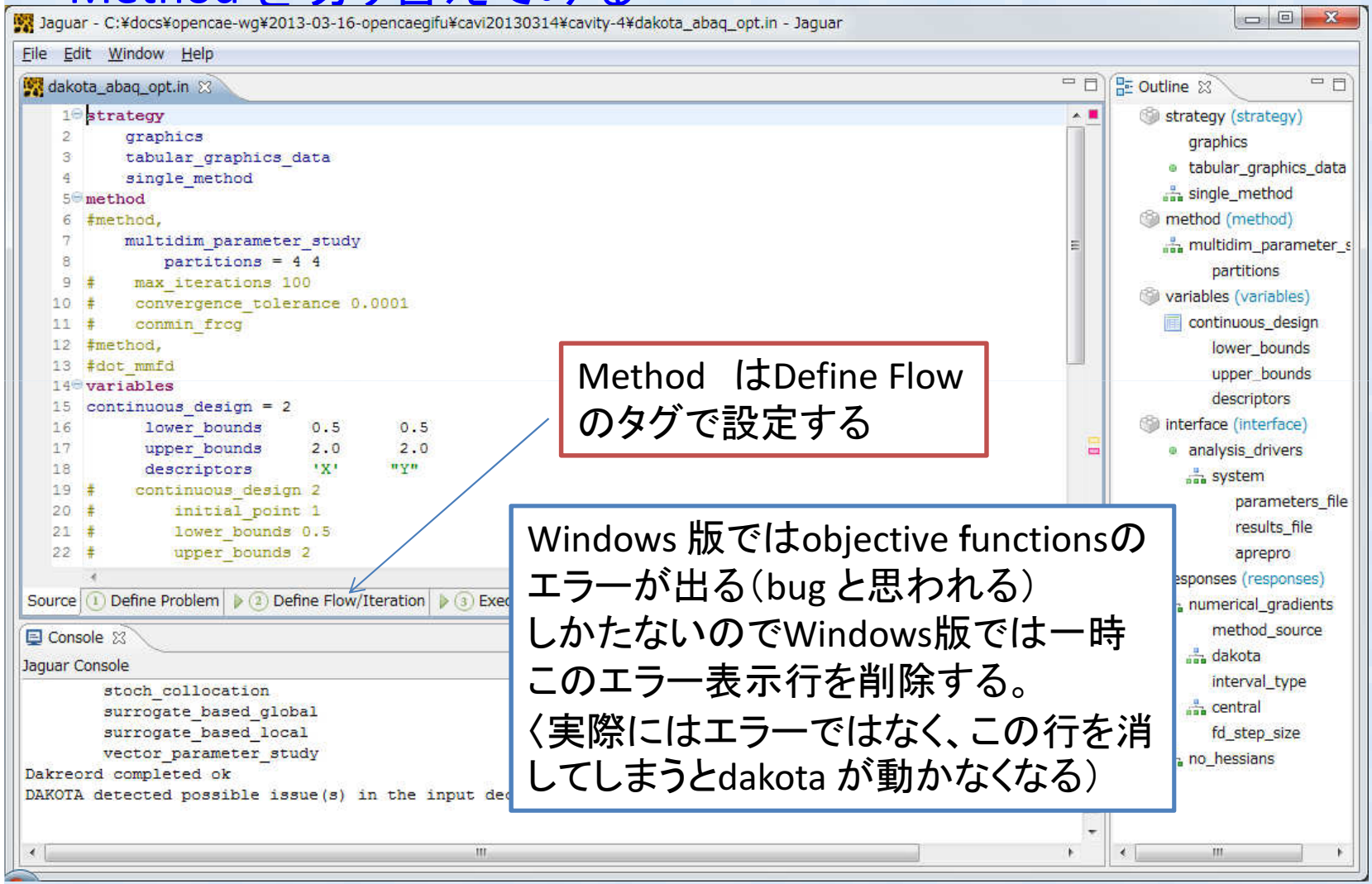
結果の分析機能もあるようだが未確認



GUIツール：Jaguarを使って入力ファイルを編集してみる①

Jaguar を起動し、dakota_abaq_opt.inを読み込む

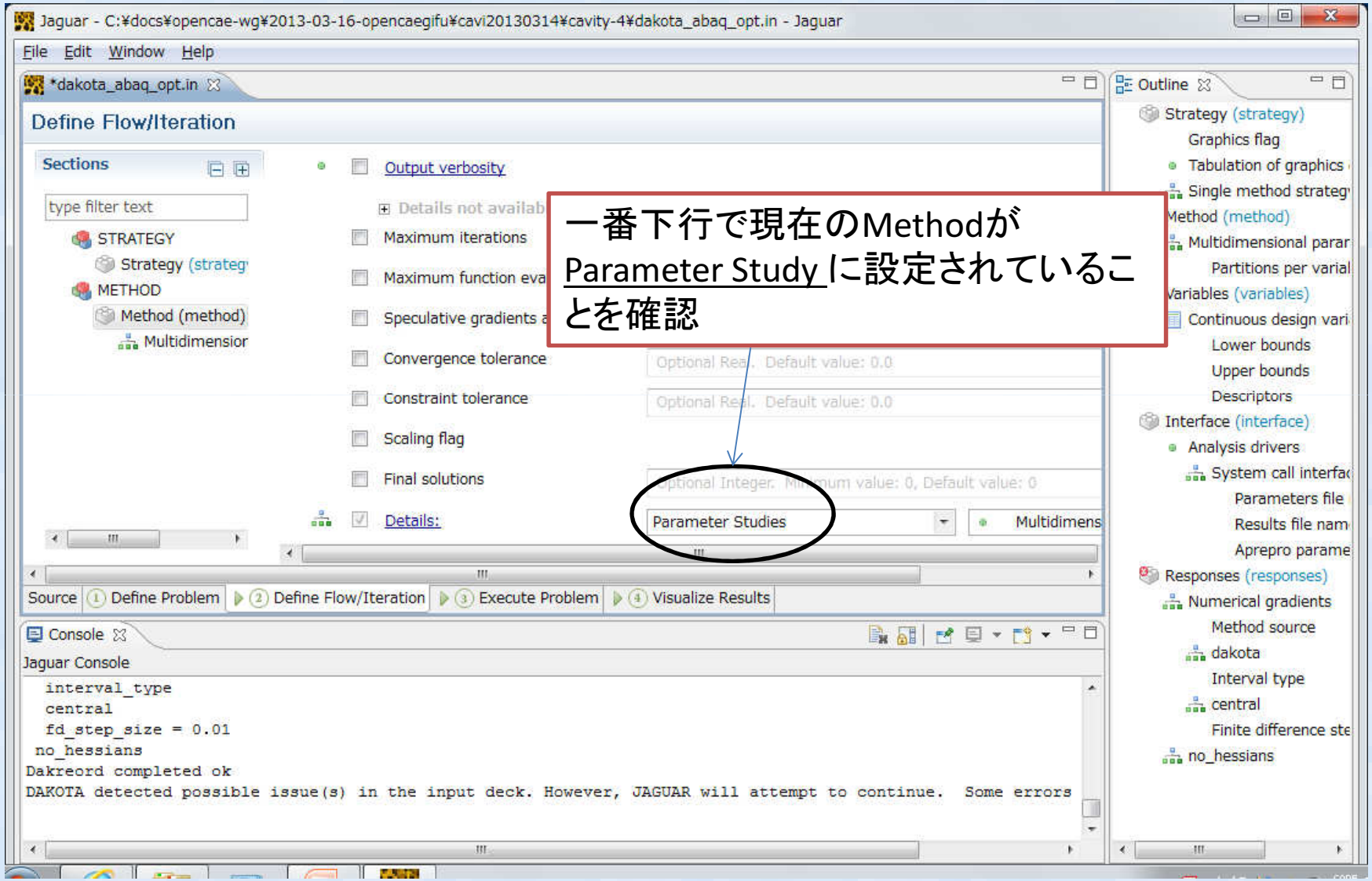
-Method を切り替えてみる



Method　はDefine Flow のタグで設定する

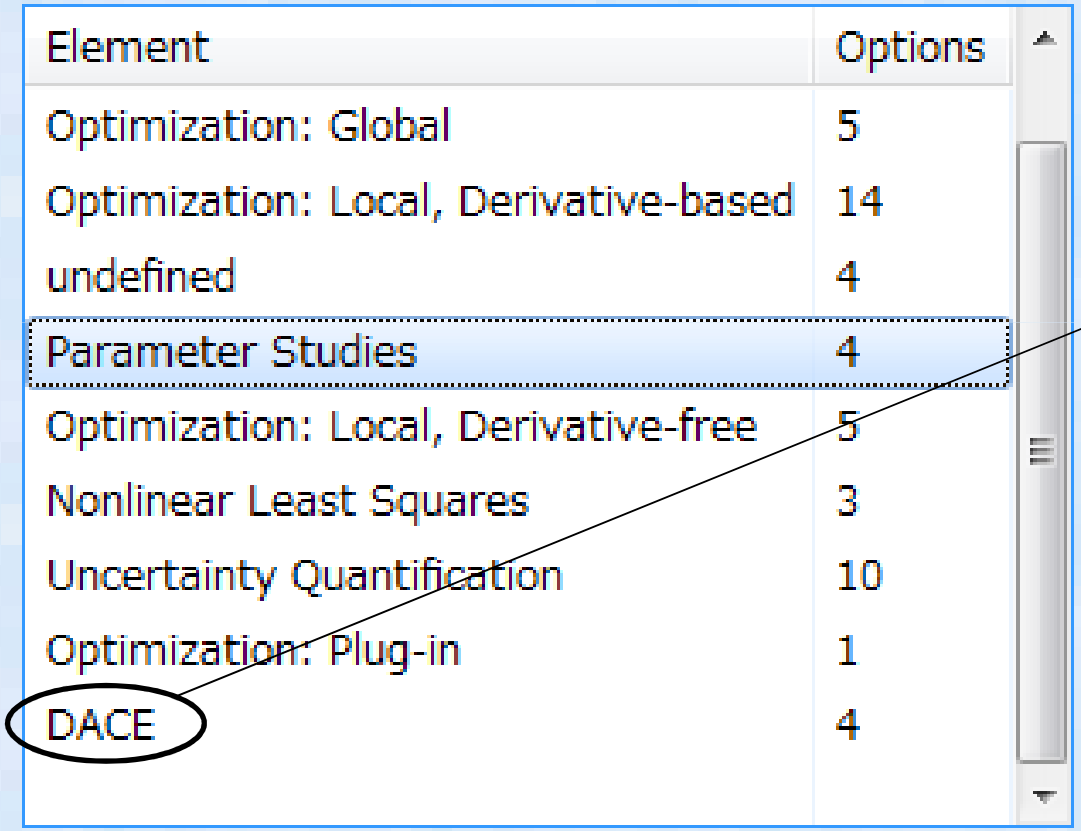
Windows 版ではobjective functionsのエラーが出る（bug と思われる）
しかたないのでWindows版では一時このエラー表示行を削除する。
〈実際にはエラーではなく、この行を消してしまうとdakota が動かなくなる）

GUIツール：Jaguarを使って入力ファイルを編集してみる②



GUIツール：Jaguarを使って入力ファイルを編集してみる③

このTag を選択すると他の手法に切り替えられる。
ここではDACE (design and analysis of computer experiments
計算機支援実験計画法) に切り替える



- Central Composite Design
- Box-Behnken Design
- Orthogonal Array Designs
- Grid Design
- Monte Carlo Design
- LHS Design
- OA-LHS Design

CCD: Central Composite Design を設定する

GUIツール：Jaguarを使って入力ファイルを編集してみる④

strategy　　IN ファイルを別名で保存する

　graphics

　tabular_graphics_data

　single_method

method

　dace

　　central_composite

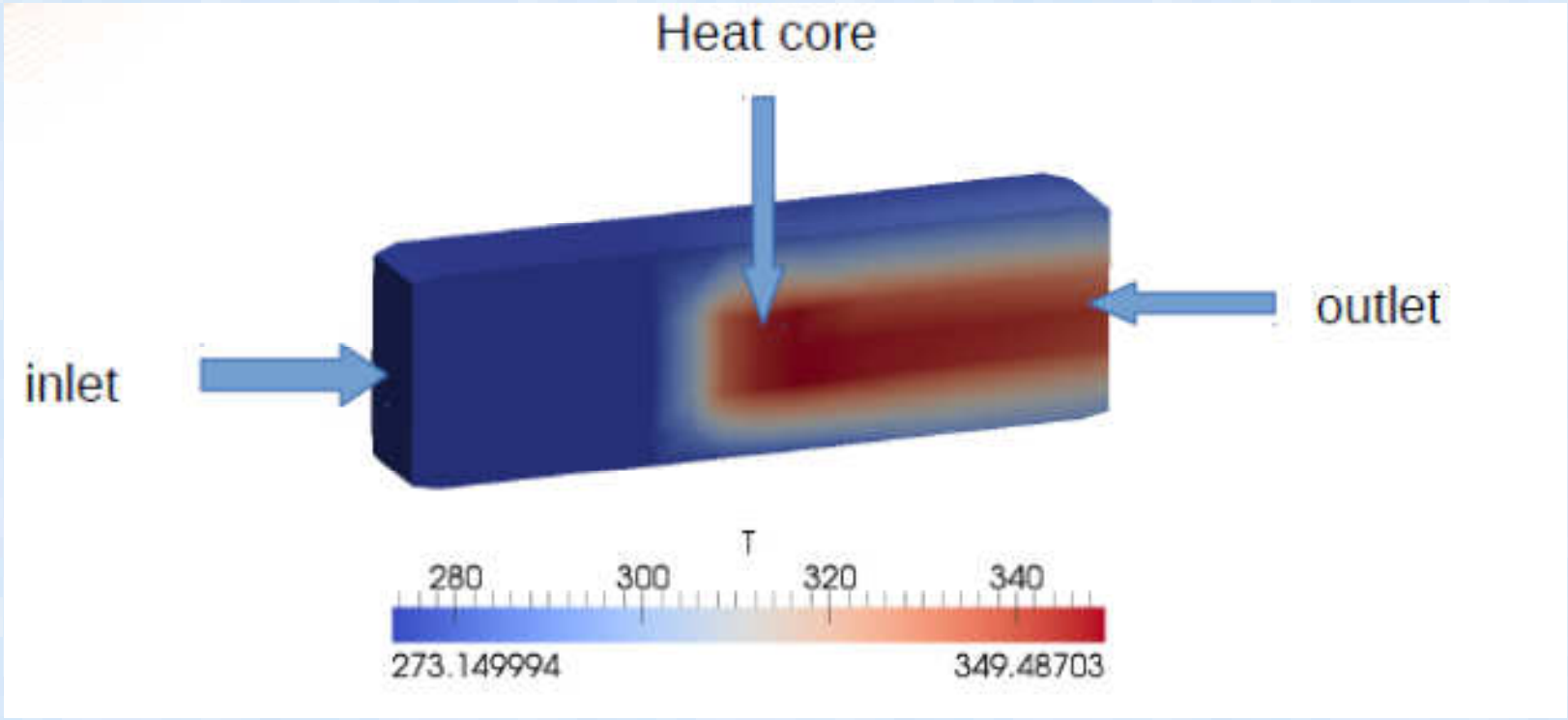
Method が　dace　のcentral_composite
に切り替わる

CCD：　中央複合実験計画によって
9ケースのサンプリングが実施される

%eval_id	X	Y	obj_fn
1	1.25	1.25	-0.37245
2	0.5	1.25	-0.21946
3	2	1.25	-0.31724
4	1.25	0.5	-0.20699
5	1.25	2	-0.33235
6	0.5	0.5	-0.36073
7	2	0.5	0.596946
8	0.5	2	-4.51749
9	2	2	-0.38354

Dakota OpenFOAM vector_parameter_study Tutorial

- 12月のOpenCAEシンポジウム＠大阪にて Dakota-OpenFOAM 連携の紹介あり

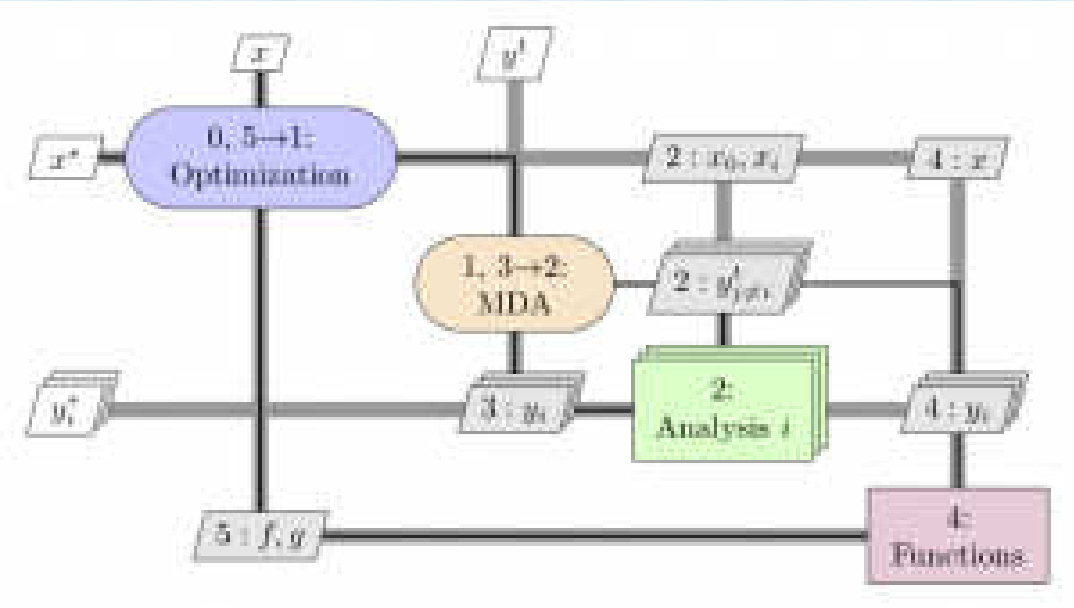


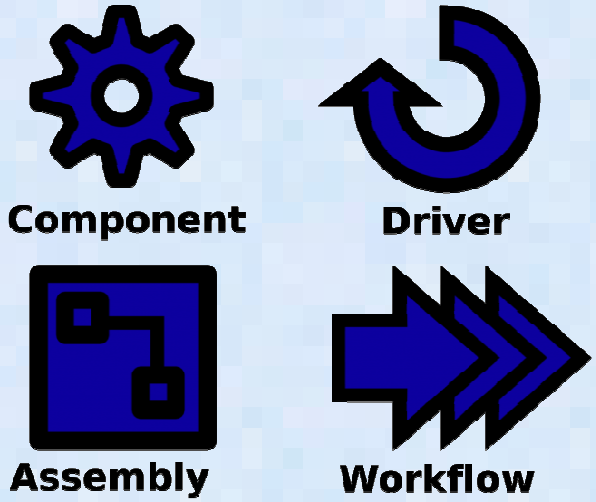
OpenMDAOとは？

・Dakotaと同様に自動的に解析に入力するパラメータを変更したパラメータSTUDYや目的となる指標（目的関数）を与えると自動的にそのような目的の値になるようなパラメータを見つけてくれるツール。Python で記述される。
・関西勉強会にて片山さんが資料をアップしていたので自分のUbuntu12.4LTSにインストールしてみた

- **http://openmdao.org/**

openMDAO





OpenMDAOインストールメモ

- インストールメモ
  - 基本的には片山さんが関西の勉強会
    2013/8/3 の資料にアップしているやりかた
    にて問題なし。。。。
  - **http://openmdao.org/**
- go-openmdao.py を上記HOMEページからDownload して、端末から実行するだけ
  abc$ python go-openmdao.py
- 上記 Python script は インストラーで本体は入っていない。ネットワークからダウンロードしつつインストールするので、ネット接続した状態で実行する必要がある。
- 私の環境では、python-scipy が無いよ！と怒られたので、これのみ追加インストールした
  abc$ sudo apt-get install python-scipy
- Python がInstall されていればWindows にもそのままインストール可能の模様だが、未確認

OpenMDAOテストメモ①

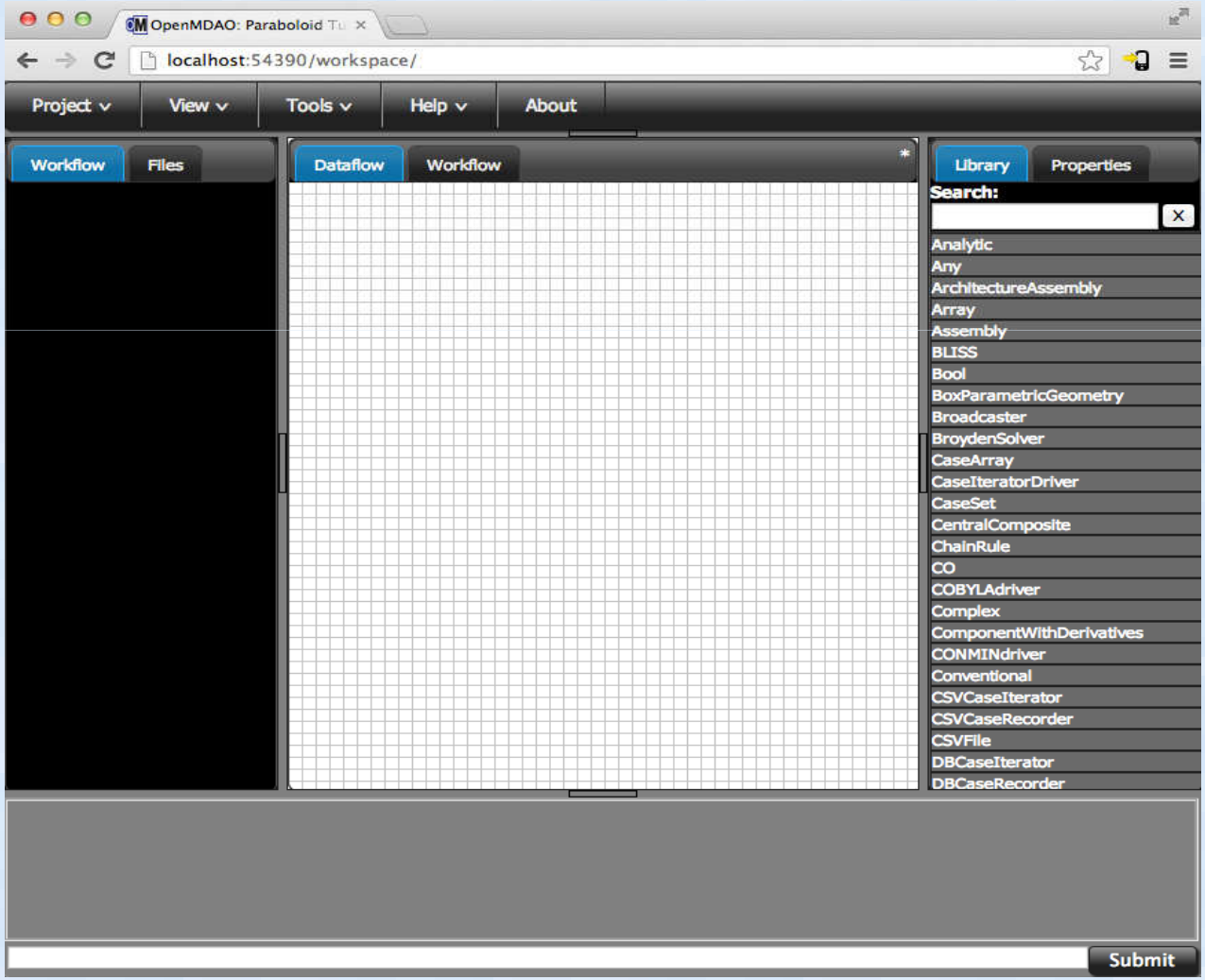
- . bin/activate
  にて仮想環境(virtualenv)に入る　。
  この仮想環境はどういう意味があるのか不明であるが、ともかくそういうものらしい

- (openmdao-0.9.5)dexcs@dexcs-laptop:~/openmdao-0.9.5$ openmdao test
  プロンプトが変わって仮想環境に入ったことが確認できる。
  テストコマンドを実行

- テストが無事終了すると
  “Ran 813 tests in 124.134”
  のメッセージが表示

OpenMDAOテストメモ②

- openmdao gui
  にてGUIを起動する(ブラウザが起動)



-WorkFlowにて動作手順が定義できるようでDakota よりは商用の最適化ツールに近いイメージになっているようだ。

まとめ

- Dakotaのv5.4機能について調査
- DakotaとCalculix/OpenFOAMの連携について簡単な例題を紹介
- Dakota　GUI Jaguarを使ってMethodの入れ替えの例を提示
- OpenMDAOのインストールについて調査
- 今後、DakotaとOpenMDAOの比較を実施予定