

PCクラスタコンソーシアム HPCオープンソースソフトウェア普及部会ワークショップ  
「高性能数値計算ライブラリLexADV」  
2021年9月10日 オンライン (Zoom)

# 粒子法MPSライブラリ LexADV\_EMPSの概要と実装

鄭 宏杰 (東洋大学)

# 本日の内容

- LexADV\_EMPSの概要
- LexADV\_EMPSのインストール及び使い方
- LexADV\_EMPSの応用例
- まとめ

# LexADV\_EMPSの概要

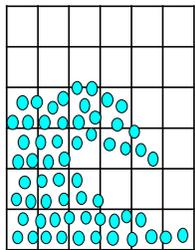
- LexADV\_EMPS: Explicit MPS (Moving Particle Simulation) Solver framework
- 高並列な分散メモリ環境での粒子法ソルバー
- オープンソースソフトウェア <http://adventure.sys.t.u-tokyo.ac.jp/lexadv>
- プログラム言語: C, MPI
- Platform: Linux, Mac OS X, Windows (Cygwin)
- 公開履歴: Ver.0.1.2.b, 2014/12/01



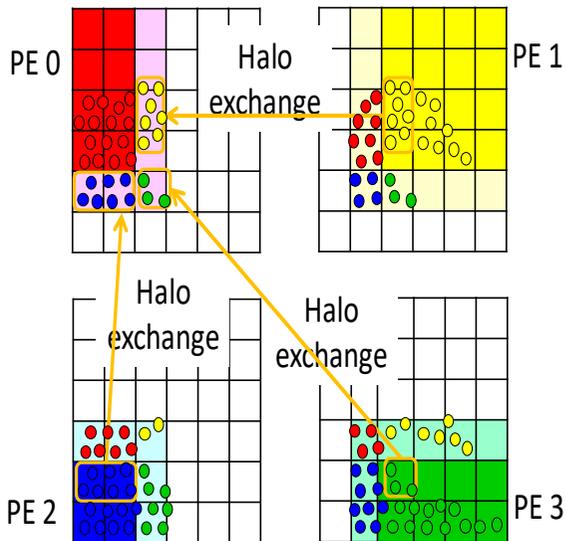
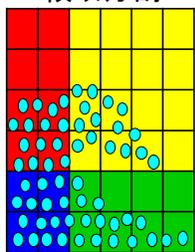
# LexADV\_EMPSの概要

- バケットベースの領域分割と通信方法

粒子をバケットに格納

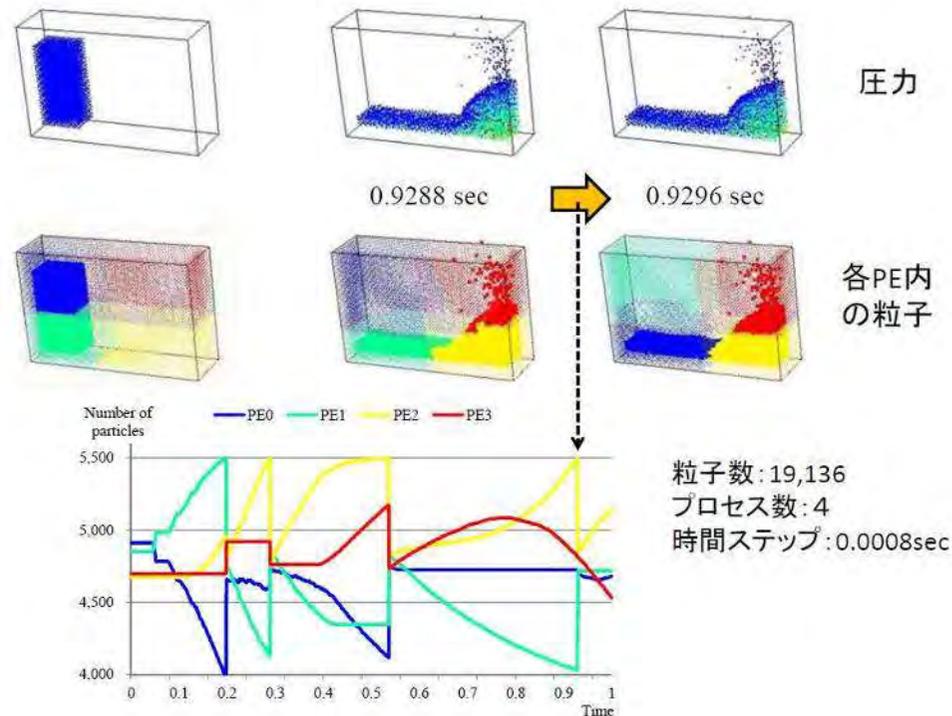


領域分割



隣接プロセス間通信を少なくするバケットベース  
2階層領域分割とHalo通信パターン生成機能

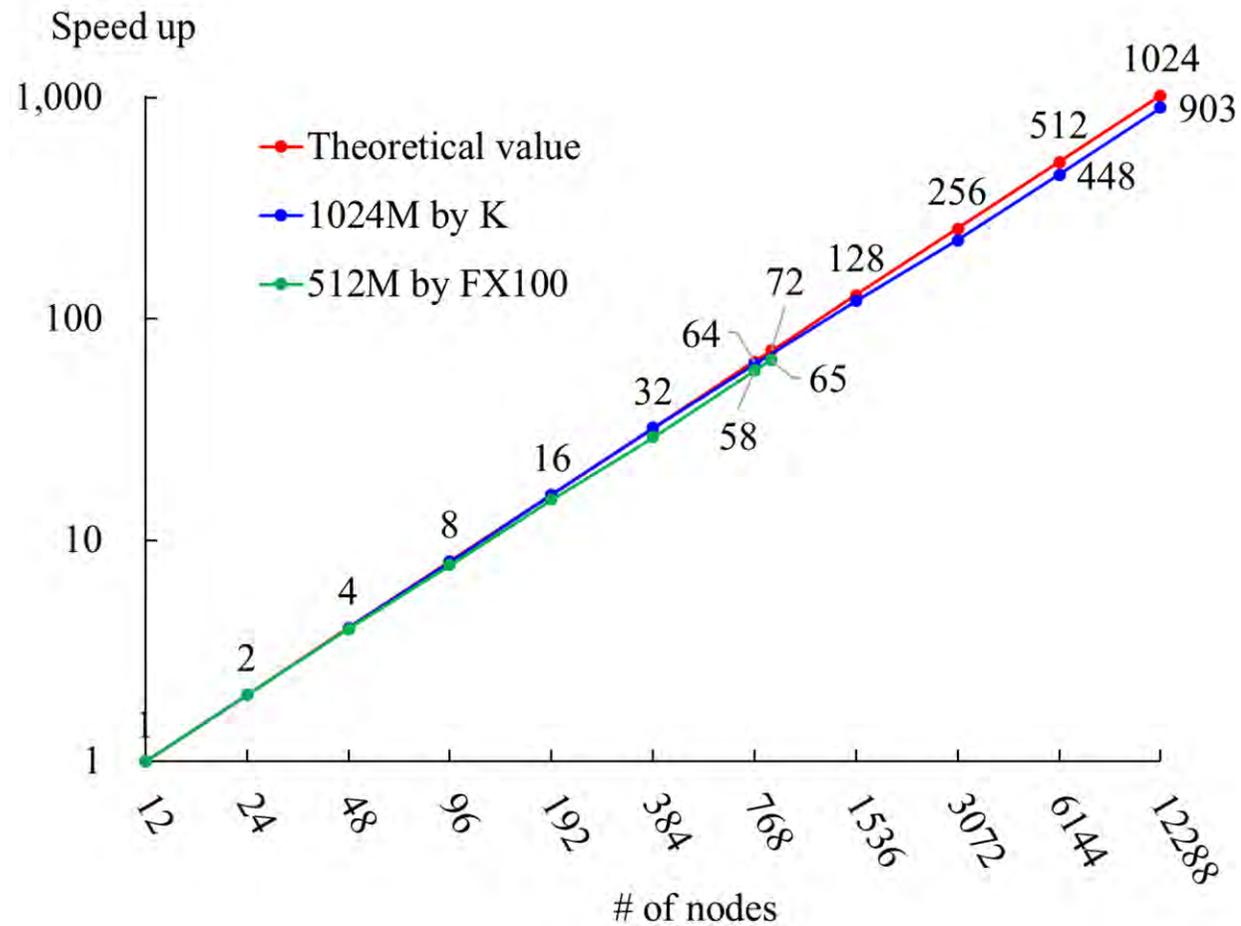
- 粒子の分散メモリ並列処理向け動的負荷分散



1つの計算ノードの粒子数が、5,500になったら、  
領域再分割を行う

# LexADV\_EMPSの概要

- Strong scaling by K computer and FX100



# LexADV\_EMPSの概要

## 支配方程式： ナビエス・ストークス方程式

連続の式

陽解法

微圧縮性を仮定

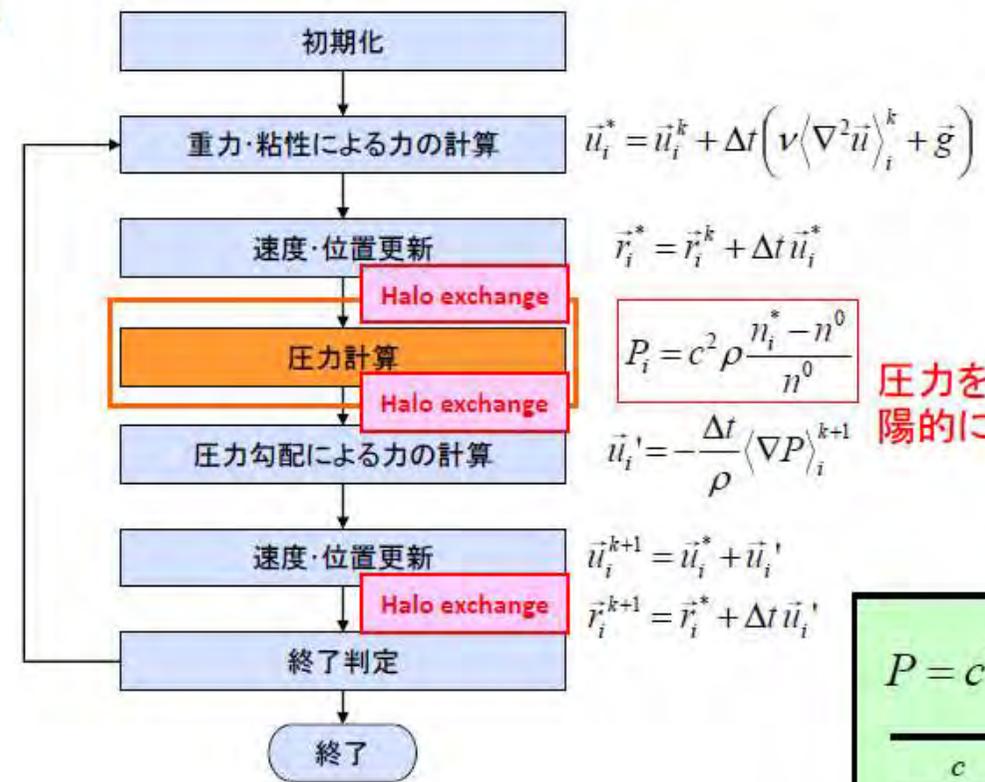
$$\frac{\partial P}{\partial \rho} = c^2$$

$c$  : 音速  
 $\rho$  : 密度

運動方程式

$$\frac{d\vec{v}}{dt} = -\frac{1}{\rho} \nabla P + \nu \Delta \vec{v} + \vec{g}$$

## MPS陽解法の通信箇所



- 密度:  $\rho$
- 位置:  $\vec{r}$
- 速度:  $\vec{u}$
- 圧力:  $P$
- 動粘性係数:  $\nu$
- 重力加速度:  $\vec{g}$
- 粒子数密度:  $n^*$
- 粒子数密度の基準値:  $n^0$

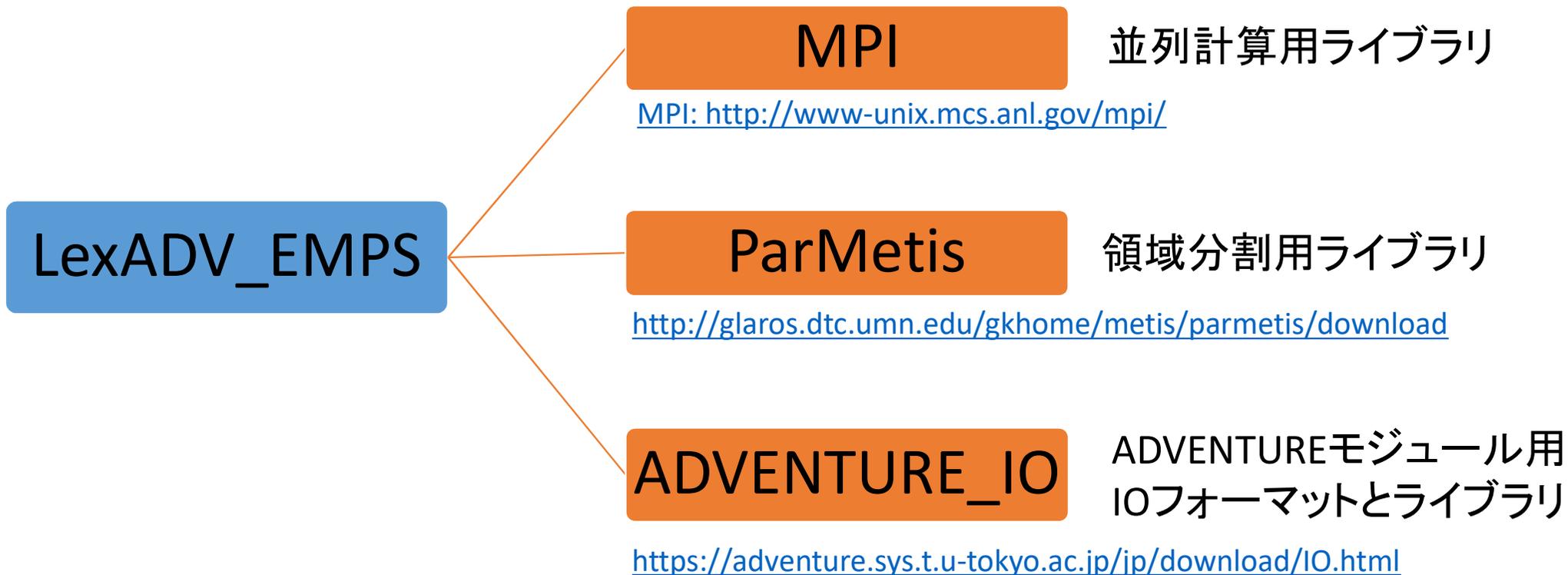
圧力を粒子数密度差から陽的に求める

$$P = c^2 \rho \frac{n^* - n^0}{n^0} \leftarrow \frac{dP}{d\rho} = c^2$$

$c$	音速 [m/s]
$n^*$	粒子数密度
$n^0$	粒子数密度の基準値

# LexADV\_EMPSのインストール

- LexADV\_EMPSをインストールする前に、依存する外部ライブラリをインストール



# LexADV\_EMPSのインストール

- LexADV\_EMPSをダウンロードする

[https://adventure.sys.t.u-tokyo.ac.jp/lexadv/lexadv\\_EMPS.html](https://adventure.sys.t.u-tokyo.ac.jp/lexadv/lexadv_EMPS.html)

The screenshot shows a web browser window with the URL `adventure.sys.t.u-tokyo.ac.jp/lexadv/lexadv_EMPS.html`. The page title is "LexADV\_EMPS download". Below the title, there is a blue box with the following text: "To download LexADV libraries, user registration is required. LexADV and ADVENTURE project uses the download account information mutually. If you do not have your account for LexADV or ADVENTURE, click the following link and sign up. User registration". A red circle highlights the "User registration" link. Below this, there is a blue box with the text "LexADV\_EMPS Explicit MPS (moving particle simulation) solver framework." and a "Download" button with a download icon. A red circle highlights the "Download" button. Below the download button, there is a list of features for LexADV\_EMPS.

初めてのユーザーに registrationが必要

コードをDownload する

LexADV\_EMPS download

LexADV\_EMPS can be downloaded from this page.

To download LexADV libraries, user registration is required.  
LexADV and ADVENTURE project uses the download account information mutually.  
If you do not have your account for LexADV or ADVENTURE, click the following link and sign up.  
User registration

LexADV\_EMPS  
Explicit MPS (moving particle simulation) solver framework.

Download / ver. 0.1.2b / released: 2014/12/01

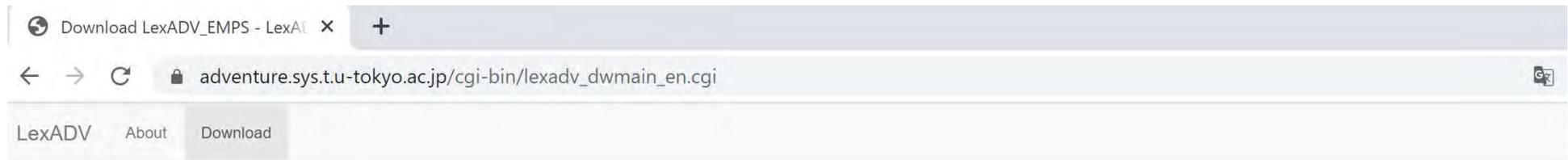
LexADV\_EMPS has the following features.

- LexADV\_EMPS solves the explicit MPS (moving particle simulation) method.
- LexADV\_EMPS supports two level domain decomposition.
- LexADV\_EMPS supports dynamic load balance.
- LexADV\_EMPS supports halo exchange (communication among nodes).
- LexADV\_EMPS supports high order polynomial approximation.
- LexADV\_EMPS supports interaction functions between fluid and rigid bodies.
- LexADV\_EMPS uses the ADVENTURE\_IO (AdvIO) library for reading input data.
- LexADV\_EMPS uses the ParMETIS library for domain decomposition.
- LexADV\_EMPS is written by C language.
- LexADV\_EMPS uses the Message Passing Interface (MPI) library for parallel processing.

# LexADV\_EMPSのインストール

- LexADV\_EMPSをダウンロードする

[https://adventure.sys.t.u-tokyo.ac.jp/lexadv/lexadv\\_EMPS.html](https://adventure.sys.t.u-tokyo.ac.jp/lexadv/lexadv_EMPS.html)



## LexADV download

After confirming the following specifications of this module, download the program.

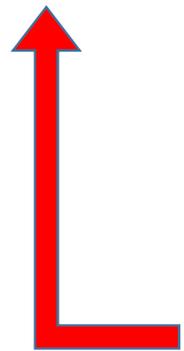
LexADV_EMPS Explicit MPS (moving particle simulation) solver framework	
 Download	
Version	0.1.2b
Date	01/Dec/2014
Function	Solves interaction between fluid and rigid bodies by the explicit MPS (moving particle simulation) method.
Platform	Unix, Linux
Compiler	C
Depends on	MPI, ADVENTURE_IO, ParMETIS(A part of ADVENTURE_Metis)



LexADV\_EMPS-0.1.2b.tar.gz

# LexADV\_EMPSのインストール

- LexADV\_EMPSを展開する



```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS
```

```
$ ls
```

```
LexADV_EMPS-0.1.2b.tar.gz
```

```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS
```

```
$ tar -zxvf LexADV_EMPS-0.1.2b.tar.gz
```

```
LexADV_EMPS_v0.1.2b_141201/
```

```
LexADV_EMPS_v0.1.2b_141201/make_floating_advio/
```

```
LexADV_EMPS_v0.1.2b_141201/doc/
```

```
LexADV_EMPS_v0.1.2b_141201/make_dambreak_advio/
```

```
LexADV_EMPS_v0.1.2b_141201/LexADV_EMPS/
```

```
LexADV_EMPS_v0.1.2b_141201/HDDM_EMPS_with_LexADV_EMPS/
```

```
LexADV_EMPS_v0.1.2b_141201/HDDM_EMPS/
```

```
LexADV_EMPS_v0.1.2b_141201/adv2vtu/
```

```
LexADV_EMPS_v0.1.2b_141201/README.txt
```

```
LexADV_EMPS_v0.1.2b_141201/copyright.txt
```

```
LexADV_EMPS_v0.1.2b_141201/ChangeLog.txt
```

# LexADV\_EMPSのインストール

- LexADV\_EMPSを展開した中身

LexADV\_EMPS



doc

LexADV\_EMPSの概要と使い方



LexADV\_EMPS

LexADV\_EMPSライブラリ



HDDM\_EMPS

LexADV\_EMPSを利用したMPS陽解法解析コード



HDDM\_EMPS\_with\_LexADV\_EMPS

LexADV\_EMPSをHDDM\_EMPSのディレクトリに入れたコード



make\_dambreak\_advio

ダムブレイク解析用の粒子生成コード



make\_floating\_advio

2つ剛体が浮遊する解析用の粒子生成コード



adv2vtu

ParaViewで可視化するためのvtuファイル生成コード

# LexADV\_EMPSのインストール

- LexADV EMPSの中身を確認する

cm@DESKTOP-1B1M200 LexADV\_EMPS/

```
ChangeLog.txt  mps_chk_overlapping_Bucket.c  mps_domainDecomposition.c  mps_read.c
copyright.txt  mps_communication.c           mps_func.h                 mps_rigid.c
makefile       mps_define.h                  mps_globalValues.h        mps_tools.c
makefile.fx10  mps_define_fixed.h           mps_init.c                 mps_write.c
makefile.gcc   mps_define_macro.h           mps_memory.c               mps_parameter.c
mps_bucket.c   mps_distribution_particle.c   mps_parameter.c
```

- makefileを編集し、外部ライブラリのパスを設定する

```
PROGRAMS = liblexadv_emps.a
```

```
CC = mpicc
```

```
CFLAGS = -fopenmp -O3
```

```
AR = ar
```

```
ARFLAGS = crsv
```

```
ADVSYS = /home/muro/ADVENTURE
```

```
ADVLIB = $(ADVSYS)/lib
```

```
ADVINC = $(ADVSYS)/include
```

```
#METIS = /home/muro/programs/metis/mpich2/ParMetis-3.1.1
```

```
METIS = ../../ParMetis-3.1.1
```

```
#ADV_IO
```

```
INCDIR += -I$(ADVINC)
```

```
#LIBDIR += -L$(ADVLIB)
```

ADVIOのパスを設定する

ParMetisのパスを設定する

# LexADV\_EMPSのインストール

- 計算条件は、mps\_define.hのパラメータを設定する

cm@DESKTOP-1B1M200 LexADV\_EMPS/

```
ChangeLog.txt  mps_chk_overlapping_Bucket.c  mps_domainDecomposition.c  mps_read.c
copyright.txt  mps_communication.c             mps_func.h                  mps_rigid.c
makefile       mps_define.h                    mps_globalValues.h         mps_tools.c
makefile.fx10  mps_define_fixed.h             mps_init.c                  mps_write.c
makefile.gcc   mps_define_macro.h             mps_memory.c                mps_parameter.c
mps_bucket.c   mps_distribution_particle.c     mps_parameter.c
```

```
#define KinematicViscosity 0.000001 // 動粘性係数(水)
#define CollisionRatio 0.2 // 衝突係数
#define DistanceLimitRatio 0.9 // 衝突半径
#define Gravity_x 0.0 // 重力
#define Gravity_y 0.0
#define Gravity_z -9.8
#define CourantNumber 0.1 // クーラン数
.....
.....
```

# LexADV\_EMPSのインストール

- LexADV\_EMPSをコンパイル

```
cm@DESKTOP-1B1M200 LexADV_EMPS
```

```
$ make
```

```
Building file: mps_init.c
```

```
mpicc -I/home/cm/ADVENTURE/include -I/home/cm/ParMetis-3.1.1 -  
I/home/cm/ParMetis-3.1.1/METISLib -I/home/cm/ParMetis-  
3.1.1/ParMETISLib -c -fopenmp -O3 -o"mps_init.o" "mps_init.c"
```

```
Finished building: mps_init.c
```

```
mpicc -I/home/cm/ADVENTURE/include -I/home/cm/ParMetis-3.1.1 -  
I/home/cm/ParMetis-3.1.1/METISLib -I/home/cm/ParMetis-3.1.1/ParMETISLib  
-c -fopenmp -O3 -o"mps_distribution_particle.o"
```

```
"mps_distribution_particle.c"
```

```
Finished building: mps_distribution_particle.c
```

# LexADV\_EMPSのインストール

- MPS陽解法ソルバーHDDM\_EMPSの中身を確認してコンパイルする

cm@DESKTOP-1B1M200 HDDM\_EMPS

\$ ls

```
ChangeLog.txt          fx10_floating.sh      mps_cal_PressureGradient.c
EMPSsettings_r001.txt  main.c                 mps_cal_PressureSmoothing.c
EMPSsettings_r002.txt  makefile             mps_cal_ViscosityAndGravity.c
copyright.txt          makefile.fx10         mps_cal_func.h
dam_r2.sh              makefile.gcc          mps_simulation.c
floating_r2.sh         mps_cal_Collision.c
fx10_dam.sh           mps_cal_NumberDensityAndPressure.c
```

```
PROGRAMS = hddm_emps
```

```
CC = mpicc
```

```
COPTIONS = -fopenmp -O3
```

```
CFLAGS = $(OPTFLAGS) $(COPTIONS)
```

```
LIBS += -lm
```

```
ADVSYS = /home/muro/ADVENTURE
```

```
ADVLIB = $(ADVSYS)/lib
```

```
ADVINC = $(ADVSYS)/include
```

```
#METIS = ../../ParMetis-3.1.1
```

```
METIS = /work/muro/programs/metis/mpicc2/ParMetis-3.1.1
```

- makefileを編集し、外部ライブラリのパスを設定する

- HDDM\_EMPSをコンパイルする

cm@DESKTOP-1B1M200 HDDM\_EMPS

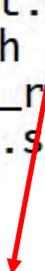
\$ make

# LexADV\_EMPSのインストール

- EMPSSettings\_r002.txtの計算条件を設定する

```
cm@DESKTOP-1B1M200 HDDM_EMPS
```

```
$ ls
ChangeLog.txt          fx10_floating.sh      mps_cal_PressureGradient.c
EMPSSettings_r001.txt  main.c                 mps_cal_PressureSmoothing.c
EMPSSettings_r002.txt  makefile               mps_cal_ViscosityAndGravity.c
copyright.txt          makefile.fx10         mps_cal_func.h
dam_r2.sh              makefile.gcc          mps_simulation.c
floating_r2.sh          mps_cal_Collision.c
fx10_dam.sh            mps_cal_NumberDensityAndPressure.c
```



<code>#TimeStep</code>	最大の時間間隔(最大速度で決定される速度と比べて小さい方が選択される)
0.0005	
<code>#FinishTime</code>	解析終了時刻
1.0	
<code>#SoundSpeed</code>	疑似音速
22.0	
<code>#OutputFrequency</code>	<code>TimeStep × OutputFrequency</code> の値間隔でファイル出力される
100	

# ダムブ레이크計算例を実行する

make\_dambreak\_advio/の中身を確認する

```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS_v0.1.2b_141201/make_dambreak_advio
$ ls
make_dambreak_advio.c makefile
```

make\_dambreak\_advio/のmakefileにADVIOのパスを設定する

```
BIN := make_dambreak_advio
all :$(BIN)

# compilers
CC = gcc

# flags
CFLAGS += -O3
LDFLAGS += -lm

#ADV_IO
ADVSYSD = /home/muro/ADVENTURE

ADVLIB = $(ADVSYS)/lib
ADVINC = $(ADVSYS)/include
INCDIR += -I$(ADVINC)
LIBDIR += -L$(ADVLIB)
LDFLAGS += -lAdvDocIO -lAdvFileIO -lAdvBase
```

# ダムブレイクの計算例を実行する

make\_dambreak\_advio.cの中身を確認する

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include "Adv/AdvDocument.h"

#define OUTPUT_ADV "dambreak.adv"
#define OUTPUT_VTU "dambreak.vtu"
FILE* fp;

#define MIN_X 0.0
#define MIN_Y 0.0
#define MIN_Z 0.0
#define MAX_X 1.0
#define MAX_Y 0.2
#define MAX_Z 0.6
#define WAVE_HEIGHT 0.5
#define WAVE_WIDTH 0.25
#define AVERAGE_PARTICLE_DISTANCE 0.02 //0.02 0.01 0.005
#define RE 5 //3 4 5
#define GHOST -1
#define FLUID 0
```

入力用のadvファイル  
可視化確認用のvtuファイル

壁+ダミー粒子の厚さ

# ダムブレイクの計算例を実行する

make\_dambreak\_advio.cを実行し入力用のadvファイルを生成する

```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS_v0.1.2b_141201/make_dambreak_advio
```

```
$ ls
```

```
Make_dambreak_advio.c  makefile
```

```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS_v0.1.2b_141201/make_dambreak_advio
```

```
$ make
```

```
gcc -O3 -lm -lAdvDocIO -lAdvFileIO -lAdvBase -c make_dambreak_advio.c -
```

```
I/home/cm/ADVENTURE/include
```

```
gcc -O3 -o make_dambreak_advio make_dambreak_advio.o -lm -lAdvDocIO -lAdvFileIO -
```

```
lAdvBase -I/home/cm/ADVENTURE/include -L/home/cm/ADVENTURE/lib
```

```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS_v0.1.2b_141201/make_dambreak_advio
```

```
$ ./make_dambreak_advio
```

```
start mk_particle
```

```
nx:60 ny:20 nz:40 nxyz:48000
```

```
NumberOfParticles: 33880
```

```
FLUID: 3380 WALL: 4964 DUMMY: 25536
```

```
Creating dambreak.vtu ... done.
```

```
end mk_particle
```

```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS_v0.1.2b_141201/make_dambreak_advio
```

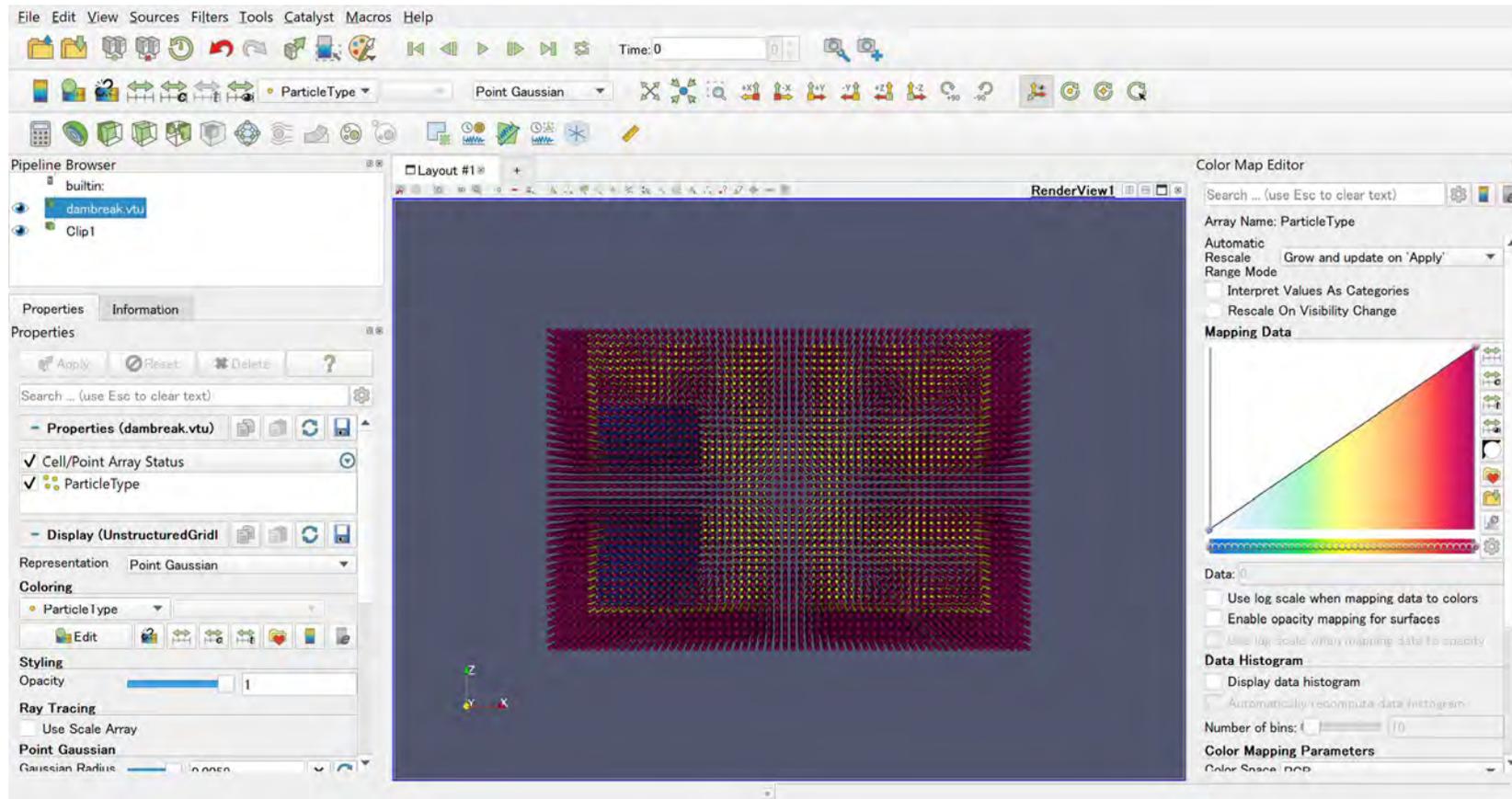
```
$ ls
```

```
dambreak.adv dambreak.vtu make_dambreak_advio.c make_dambreak_advio
```

```
make_dambreak_advio.o makefile
```

# ダムブレイクの計算例を実行する

## Paraviewでdambreak.vtuを確認する



# ダムブレークの計算例を実行する

- 実行用Shell Script「dam\_r2.sh」を使って実行する

```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS_v0.1.2b_141201/HDDM_EMPS  
$ vi dam_r2.sh
```

```
mpirun -np 4 ./hddm_emps . ./EMPSsettings_r002.txt 0 --pre ../make_dambreak_advio/dambreak.adv
```

出力ファイルを書き出す場所

入力ファイルを利用する場合は、通常は0

入力ファイルを読み込むためのオプション

入力ファイル名

# ダムブレイクの計算例を実行する

- 解析結果を確認する

```
floating_r2.sh      mps_cal_Collision.o      mps_simulation.c
fx10_dam.sh        mps_cal_NumberDensityAndPressure.c  mps_simulation.o
fx10_floating.sh   mps_cal_NumberDensityAndPressure.o  particle ←
llnl/adv/emp/      mps_cal_PressureGradient.c
main.c             mps_cal_PressureGradient.o
```

cm@DESKTOP-1B1M200 ~/LexADV\_EMPS\_v0.1.2b\_141201/HDDM\_EMPS\_dam/particle

\$ ls

```
output0000_0.adv  output0004_0.adv  output0008_0.adv  output0012_0.adv  output0016_0.adv  output0020_0.adv  output0024_0.adv  output0028_0.adv
output0000_1.adv  output0004_1.adv  output0008_1.adv  output0012_1.adv  output0016_1.adv  output0020_1.adv  output0024_1.adv  output0028_1.adv
output0000_2.adv  output0004_2.adv  output0008_2.adv  output0012_2.adv  output0016_2.adv  output0020_2.adv  output0024_2.adv  output0028_2.adv
output0000_3.adv  output0004_3.adv  output0008_3.adv  output0012_3.adv  output0016_3.adv  output0020_3.adv  output0024_3.adv  output0028_3.adv
output0001_0.adv  output0005_0.adv  output0009_0.adv  output0013_0.adv  output0017_0.adv  output0021_0.adv  output0025_0.adv  output0029_0.adv
output0001_1.adv  output0005_1.adv  output0009_1.adv  output0013_1.adv  output0017_1.adv  output0021_1.adv  output0025_1.adv  output0029_1.adv
output0001_2.adv  output0005_2.adv  output0009_2.adv  output0013_2.adv  output0017_2.adv  output0021_2.adv  output0025_2.adv  output0029_2.adv
output0001_3.adv  output0005_3.adv  output0009_3.adv  output0013_3.adv  output0017_3.adv  output0021_3.adv  output0025_3.adv  output0029_3.adv
output0002_0.adv  output0006_0.adv  output0010_0.adv  output0014_0.adv  output0018_0.adv  output0022_0.adv  output0026_0.adv  output0030_0.adv
output0002_1.adv  output0006_1.adv  output0010_1.adv  output0014_1.adv  output0018_1.adv  output0022_1.adv  output0026_1.adv  output0030_1.adv
output0002_2.adv  output0006_2.adv  output0010_2.adv  output0014_2.adv  output0018_2.adv  output0022_2.adv  output0026_2.adv  output0030_2.adv
output0002_3.adv  output0006_3.adv  output0010_3.adv  output0014_3.adv  output0018_3.adv  output0022_3.adv  output0026_3.adv  output0030_3.adv
output0003_0.adv  output0007_0.adv  output0011_0.adv  output0015_0.adv  output0019_0.adv  output0023_0.adv  output0027_0.adv
output0003_1.adv  output0007_1.adv  output0011_1.adv  output0015_1.adv  output0019_1.adv  output0023_1.adv  output0027_1.adv
output0003_2.adv  output0007_2.adv  output0011_2.adv  output0015_2.adv  output0019_2.adv  output0023_2.adv  output0027_2.adv
output0003_3.adv  output0007_3.adv  output0011_3.adv  output0015_3.adv  output0019_3.adv  output0023_3.adv  output0027_3.adv
```

# ダムブレイクの計算例を実行する

- リスタート機能を利用する

HDDM\_EMPS/EMPSSetting\_r002.txtを編集する

#TimeStep		#TimeStep
0.0005		0.0005
#FinishTime	→	#FinishTime
1.0		1.5
#SoundSpeed		#SoundSpeed
22.0		22.0
#OutputFrequency		#OutputFrequency
100		100

HDDM\_EMPS/dam\_r2.shを編集してリスタート機能を利用する

```
#mpirun -np 4 ./hddm_emps . ./EMPSSettings_r002.txt 0 --pre ../make_dambreak_a
```

```
mpirun -np 4 ./hddm_emps . ./EMPSSettings_r002.txt 20
```

出力ファイルを書き出す場所  
かつ、リスタートするための入  
力ファイルがある場所

リスタートするために読み込  
むファイル番号

# ダムブ레이크の計算例を実行する

- 可視化用ファイル生成コードadv2vtu/の中身を確認する

```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS_v0.1.2b_141201/adv2vtu
$ ls
adv2vtu.c  makefile
```

- Adv2vtuのmakefileにADVIOのパスを設定する

```
BIN := adv2vtu
all :$(BIN)

# compilers
CC = gcc

# flags
CFLAGS += -O3
#CFLAGS += -DMEMWATCH -DMW_STUDIO
LDFLAGS += -lm
LDFLAGS += -lz

#ADV_IO
#ADVSYSO = /group1/gc44/share/muro/gcc/AdvIO-1.2a
ADVSYSD = /home/muro/ADVENTURE

ADVLIB = $(ADVSYSO)/lib
ADVINC = $(ADVSYSO)/include
INCDIR += -I$(ADVINC)
LIBDIR += -L$(ADVLIB)
LDFLAGS += -lAdvDocIO -lAdvFileIO -lAdvBase
```

# ダムブレークの計算例を実行する

- 可視化用ファイルを作成する

```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS_v0.1.2b_141201/adv2vtu
$ ./adv2vtu ../HDDM_EMPS/particle/ vtu_dam_r002/
```

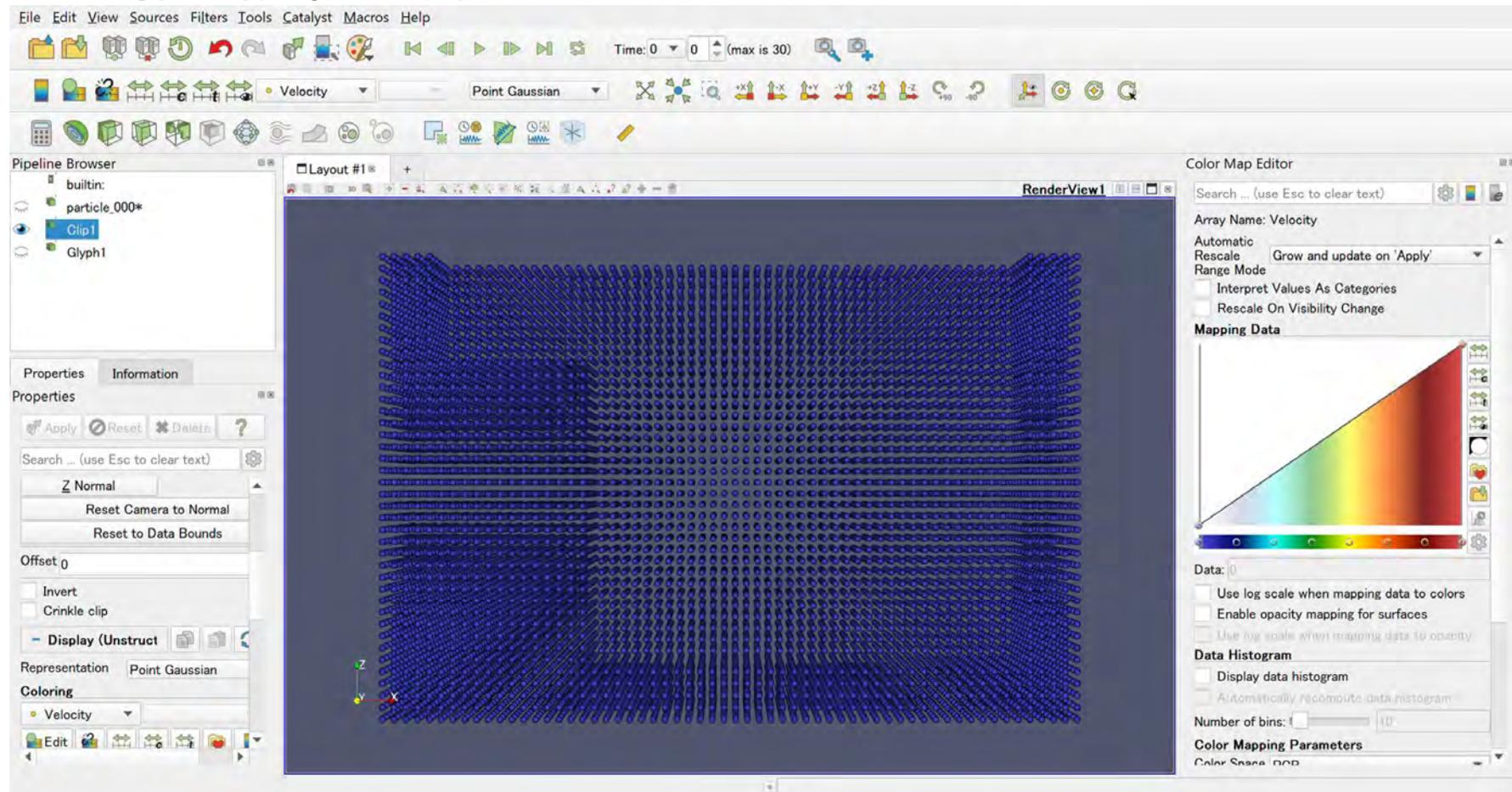
- 可視化用vtuファイルの確認

```
cm@DESKTOP-1B1M200 ~/LexADV_EMPS_v0.1.2b_141201/adv2vtu
$ ls vtu_dam_r002/
```

```
particle_00000.vtu  particle_00008.vtu  particle_00016.vtu  particle_00024.vtu
particle_00001.vtu  particle_00009.vtu  particle_00017.vtu  particle_00025.vtu
particle_00002.vtu  particle_00010.vtu  particle_00018.vtu  particle_00026.vtu
particle_00003.vtu  particle_00011.vtu  particle_00019.vtu  particle_00027.vtu
particle_00004.vtu  particle_00012.vtu  particle_00020.vtu  particle_00028.vtu
particle_00005.vtu  particle_00013.vtu  particle_00021.vtu  particle_00029.vtu
particle_00006.vtu  particle_00014.vtu  particle_00022.vtu  particle_00030.vtu
particle_00007.vtu  particle_00015.vtu  particle_00023.vtu
```

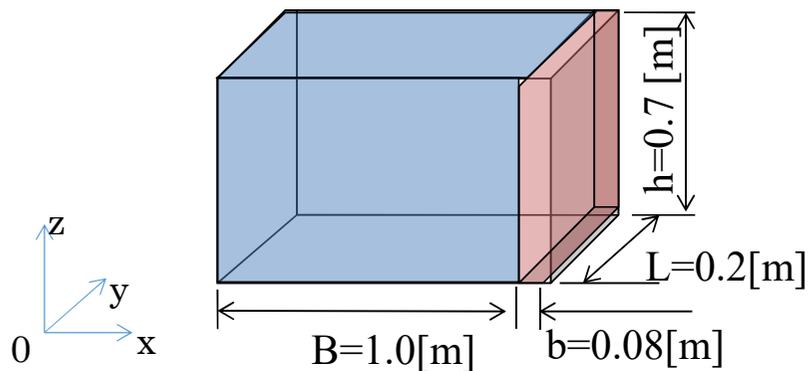
# ダムブレイクの計算例を実行する

- ParaViewで解析結果の確認



# LexADV\_EMPSの応用例

## 静水圧



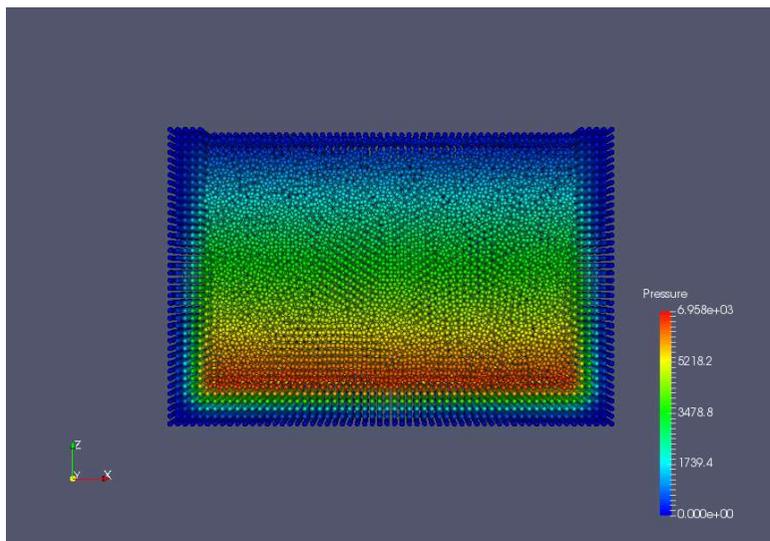
静水圧  $p$  の理論解:

$$p = \rho \cdot |g| \cdot (0.7 - h)$$

$p$ : 流体密度,  $g$ : 重力加速度ベクトル  
 $h$ : 水面からの深さ

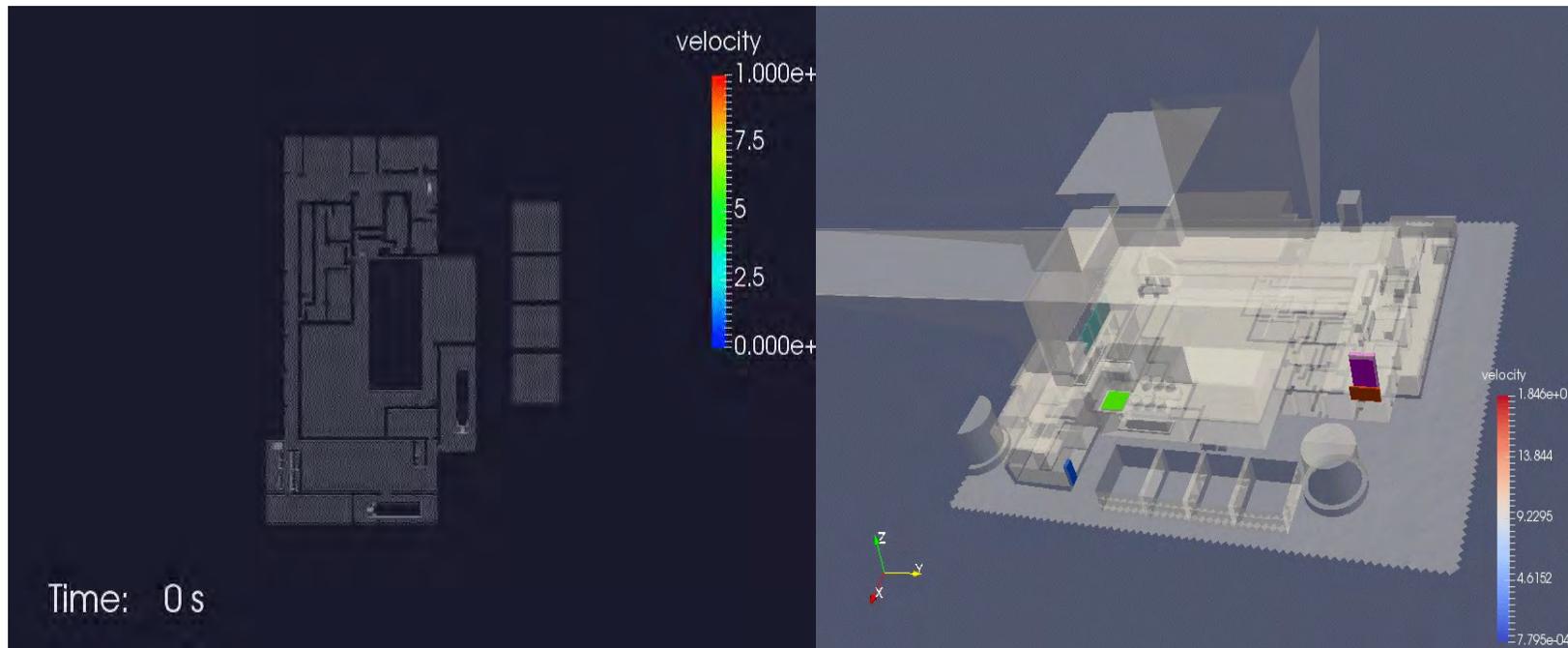
### 流体解析MPSの計算条件

時間刻み幅	0.0005 [s]
流体粒子数	17,500
壁粒子数	4,964
初期粒子間距離	0.02[m]
影響半径	0.01 [m]
流体密度	1000 [kg/m <sup>3</sup> ]
動粘性係数	0.0001 [m <sup>2</sup> /s]
重力加速度	9.8[m/s <sup>2</sup> ]
プロセッサ数	4



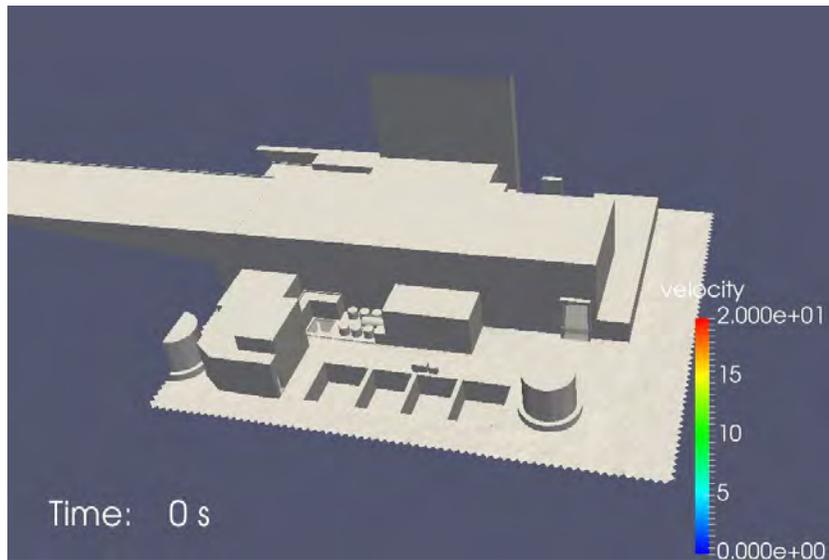
# LexADV\_EMPSの応用例

福島第一原子力発電所1号機タービン建屋の浸水解析  
粒子径:0.1[m], 粒子数:約1.3億



# LexADV\_EMPSの応用例

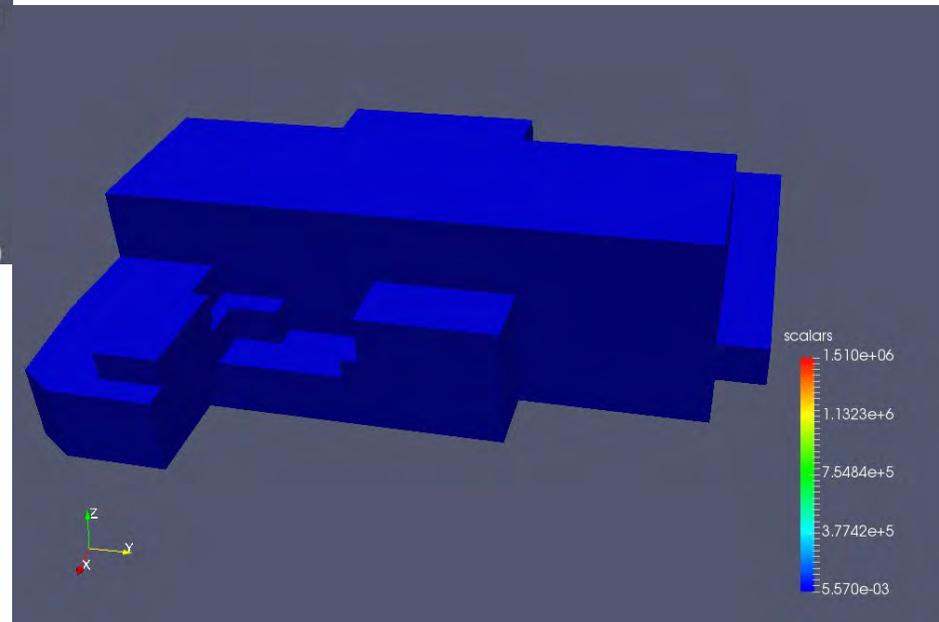
## 大規模流体—構造連成解析



計算機:スーパーコンピュータ京  
ノード数:4800  
計算時間:約11時間

粒子径:0.1[m]  
粒子数:約1.3億

建屋四面体2次要素数:1,694,962  
自由度数:8,402,583



# まとめ

- LexADV\_EMPSの概要
- LexADV\_EMPSのインストールから結果出るまでのデモ
- LexADV\_EMPSの応用例

# 参考文献

- ADVENTURE Project: <https://adventure.sys.t.u-tokyo.ac.jp/>
- LexADV\_HDDMPSS: <https://adventure.sys.t.u-tokyo.ac.jp/lexadv/index.html>
- 室谷浩平, 分散メモリ並列EMPSライブラリLexADV\_EMPS V0.1b, 第9回ADVENTURE定期セミナー, LexADV\_EMPSのマニュアル(LexADV\_EMPS-0.1.2b/doc/LexADV\_EMPS\_v0.1b\_20141113.pdf).
- Masao Ogino, Hongjie Zheng, Kohei Murotani, Seiichi Koshizuka, Ryuji Shioya, Liu Lijun, Tsunami Run-Up and Inundation Simulations Using LexADV\_EMPS Solver Framework on Fujitsu FX100, SC16, 2016.
- 室谷浩平, 荻野正雄, 塩谷隆二, MPS法を用いた東日本大震災の津波による福島第一原子力発電所1号機タービン建屋の浸水解析, HPCI利用研究成果集 hp150189, 2017.
- Hongjie Zheng, Ryuji Shioya, Naoto Mitsume, Large-Scale Parallel Simulation of Coastal Structures Loaded by Tsunami Wave Using FEM and MPS Method, Journal of Advanced Simulation in Science and Engineering, 5, 11 – 16, 2018.