HydroCODE

制作者 Doxygen 1.9.3

Chapter 1

1D Godunov/GRP scheme for Lagrangian/Eulerian hydrodynamics

This is an implementation of fully explict forward Euler scheme for 1-D Euler equations of motion on Lagrangian/↩ Eulerian coordinate.

版本

0.1

1.1 File directory

data_in/	Folder to store input files RHO/U/P/config.txt
data_out/	Folder to store output files RHO/U/P/E/X/log.txt
doc/	Code documentation generated by doxygen
src/	Folder to store C source code

1.2 Program structure

include/	Header files
file_io/	Program reads and writes files
finite_volume/	Finite volume scheme programs
Riemann_solver/	Riemann solver programs
tools/	Tool functions
hydrocode/hydrocode.c	Main program
hydrocode/make.sh	Bash script compiles and runs programs

1.3 Program exit status code

exit(0)	EXIT_SUCCESS
exit(1)	File directory error

exit(2)	Data reading error
exit(3)	Calculation error
exit(4)	Arguments error
exit(5)	Memory error

1.4 Compile environment

- Linux/Unix: gcc, glibc, MATLAB/Octave
  - Compile in 'src/hydrocode': Run './make.sh' command on the terminal.
- Winodws: Visual Studio, MATLAB/Octave
  - Create a C++ Project from Existing Code in 'src/'.
  - Compile in 'x64/Debug' using shortcut key 'Ctrl+B' with Visual Studio.

1.5 Usage description

- Input files are stored in folder '/data_in/one-dim/name_of_test_example'.
- Input files may be produced by MATLAB/Octave script 'value_start.m'.
- Description of configuration file 'config.txt' refers to 'doc/config.csv'.
- Run program:
  - Linux/Unix: Run 'hydrocode.out name_of_test_example name_of_numeric_result dimension order[_↩ scheme] coordinate config[n]=(double)C' command on the terminal.
    e.g. 'hydrocode.out GRP_Book/6_1 GRP_Book/6_1 1 2[_GRP] LAG 5=100' (second-order Lagrangian GRP scheme).
    - dim: Dimension of test example (= 1 or 2).
    - order: Order of numerical scheme (= 1 or 2).
    - scheme: Scheme name (= Riemann_exact/Godunov, GRP or . . .)
    - coordinate: Lagrangian/Eulerian coordinate framework (= LAG or EUL).
  - Windows: Run 'hydrocode.exe name_of_test_example order coordinate' command on the terminal.
    [Debug] Project -> Properties -> Configuration Properties -> Debugging

Command Arguments	name_of_test_example order coordinate (e.g. 'GRP_Book_6_1 1 EUL')
Working Directory	$(SolutionDir)$(Platform)\$(Configuration)\

[Run] Project -> Properties -> Configuration Properties -> Linker -> System

Subsystem	控制台 (/SUBSYSTEM:CONSOLE)

- Output files can be found in folder '/data_out/one-dim/'.
- Output files may be visualized by MATLAB/Octave script 'value_plot.m'.

Chapter 2

弃用列表

全局 **format_string (char *str)**

This function has been replaced by the variable 'errno' in the standard Library <errno.h>.

全局 **str2num (char *number)**

This function has been replaced by the 'strtod()' function in the standard Library <stdio.h>.

Chapter 3

结构体索引

3.1 结构体

这里列出了所有结构体，并附带简要说明:

Chapter 4

文件索引

4.1 文件列表

这里列出了所有文件，并附带简要说明:

Chapter 5

结构体说明

5.1 cell_var结构体 参考

Pointer structural body of variables on computational cells.

`#include <var_struc.h>`

成员变量

- double ** RHO
- double ** U
- double ** V
- double ** P
- double ** E

5.1.1 详细描述

Pointer structural body of variables on computational cells.

5.1.2 结构体成员变量说明

5.1.2.1 E

`double ** E`

5.1.2.2 P

```
double ** P
```

5.1.2.3 RHO

```
double** RHO
```

5.1.2.4 U

```
double ** U
```

5.1.2.5 V

```
double ** V
```

该结构体的文档由以下文件生成:

- src/include/var_struc.h

5.2 flu_var结构体 参考

Pointer structural body of fluid variables.

```
#include <var_struc.h>
```

成员变量

- double * RHO
- double * U
- double * P

5.2.1 详细描述

Pointer structural body of fluid variables.

5.2.2 结构体成员变量说明

5.2.2.1 P

```
double * P
```

5.2.2.2 RHO

```
double* RHO
```

5.2.2.3 U

```
double * U
```

该结构体的文档由以下文件生成:

- src/include/var_struc.h

Chapter 6

文件说明

6.1 src/file_io/_1D_file_in.c 文件参考

This is a set of functions which control the read-in of one-dimensional data.

```
#include <errno.h>
#include <math.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include "../include/var_struc.h"
#include "../include/file_io.h"
```

宏定义

- #define STR_FLU_INI(sfv)

  Count out and read in data of the initial fluid variable 'sfv'.

函数

- static int _1D_flu_var_read (FILE *fp, double *U, const int num)

  This function reads the 1D initial data file to generate the initial data.
- void _1D_initialize (const char *name, struct flu_var *FV0)

  This function reads the 1D initial data file of velocity/pressure/density.

6.1.1 详细描述

This is a set of functions which control the read-in of one-dimensional data.

6.1.2 宏定义说明

6.1.2.1 STR_FLU_INI

```
#define STR_FLU_INI(
              sfv )
```

Count out and read in data of the initial fluid variable 'sfv'.

6.1.3 函数说明

6.1.3.1 _1D_flu_var_read()

```
static int _1D_flu_var_read (
             FILE * fp,
             double * U,
             const int num )  [static]
```

This function reads the 1D initial data file to generate the initial data.

参数

in	*fp*	The pointer to the input file.
out	*U*	The pointer to the data array of fluid variables.
in	*num*	The number of the numbers in the input file.

返回

It returns 0 if successfully read the file, while returns the index of the wrong entry.

6.1.3.2 _1D_initialize()

```
void _1D_initialize (
             const char * name,
             struct flu_var * FV0 )
```

This function reads the 1D initial data file of velocity/pressure/density.

The function initialize the extern pointer FV0->RHO/U/P pointing to the position of a block of memory consisting (m+1) variables* of type double. The value of first of these variables is m. The following m variables are the initial value.

参数

in	*name*	Name of the test example.
out	*FV0*	Structural body pointer of initial data array pointer.

6.2 src/file_io/_1D_file_out.c 文件参考

This is a set of functions which control the readout of one-dimensional data.

```
#include <math.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "../include/var_struc.h"
#include "../include/file_io.h"
```

宏定义

- #define PRINT_NC(v, v_print)

  Print out fluid variable 'v' with array data element 'v_print'.

函数

- void _1D_file_write (const int m, const int N, struct cell_var CV, double *X[], const double *cpu_time, const char *name)

  This function write the solution into output files.

6.2.1 详细描述

This is a set of functions which control the readout of one-dimensional data.

6.2.2 宏定义说明

6.2.2.1 PRINT_NC

```
#define PRINT_NC(
              v,
              v_print )
```

值:

```
    do {                                \
    strcpy(file_data, add_out);                 \
    strcat(file_data, "/");                     \
    strcat(file_data, #v);                      \
    strcat(file_data, ".dat");                  \
    if((fp_write = fopen(file_data, "w")) == NULL)          \
        {                               \
        printf("Cannot open solution output file: %s!\n", #v);  \
        exit(1);                        \
        }                               \
    for(n = 0; n < N; ++n)                      \
        {                               \
        for(j = 0; j < m; ++j)                  \
            fprintf(fp_write, "%.10g\t", (v_print));        \
        fprintf(fp_write, "\n");                \
        }                               \
    fclose(fp_write);                       \
    } while (0)
```

Print out fluid variable 'v' with array data element 'v_print'.

6.2.3 函数说明

6.2.3.1 _1D_file_write()

```
void _1D_file_write (
            const int m,
            const int N,
            struct cell_var CV,
            double * X[],
            const double * cpu_time,
            const char * name )
```

This function write the solution into output files.

注解

It is quite simple so there will be no more comments.

参数

in	*m*	The number of spatial points in the output data.
in	*N*	The number of time steps in the output data.
in	*CV*	Structural body of grid variable data.
in	*X[]*	Array of the coordinate data.
in	*cpu_time*	Array of the CPU time recording.
in	*name*	Name of the numerical results.

6.3 src/file_io/config_in.c 文件参考

This is a set of functions which control the read-in of configuration data.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <stdbool.h>
#include <errno.h>
#include <ctype.h>
#include <limits.h>
#include "../include/var_struc.h"
```

函数

- static void config_check (void)

  This function check whether the configuration data is reasonable and set the default.

- static int config_read (FILE *fp)

  This function read the configuration data file, and store the configuration data in the array "config".
- void configurate (const char *add_in)

  This function controls configuration data reading and validation.

6.3.1 详细描述

This is a set of functions which control the read-in of configuration data.

6.3.2 函数说明

6.3.2.1 config_check()

```
static void config_check (
            void  )  [static]
```

This function check whether the configuration data is reasonable and set the default.

6.3.2.2 config_read()

```
static int config_read (
            FILE * fp )  [static]
```

This function read the configuration data file, and store the configuration data in the array "config".

参数

in	*fp*	The pointer to the configuration data file.

返回

Configuration data file read status.

返回值

1	Success to read in configuration data file.
0	Failure to read in configuration data file.

6.3.2.3 configurate()

```
void configurate (
            const char * add_in )
```

This function controls configuration data reading and validation.

The parameters in the configuration data file refer to 'doc/config.csv'.

参数

in	*add_in*	Adress of the initial data folder of the test example.

6.4 src/file_io/io_control.c 文件参考

This is a set of common functions which control the input/output data.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <ctype.h>
#include "../include/var_struc.h"
#include "../include/tools.h"
```

函数

- void example_io (const char *example, char *add_mkdir, const int i_or_o)
  This function produces folder path for data input or output.
- int flu_var_count (FILE *fp, const char *add)
  This function counts how many numbers are there in the initial data file.

6.4.1 详细描述

This is a set of common functions which control the input/output data.

6.4.2 函数说明

6.4.2.1 example_io()

```
void example_io (
            const char * example,
            char * add_mkdir,
            const int i_or_o )
```

This function produces folder path for data input or output.

参数

in	*example*	Name of the test example/numerical results.
out	*add_mkdir*	Folder path for data input or output.
in	*i_or_o*	Conversion parameters for data input/output. • 0: data output. • else (e.g. 1): data input.

6.4.2.2 flu_var_count()

```
int flu_var_count (
            FILE * fp,
            const char * add )
```

This function counts how many numbers are there in the initial data file.

参数

in	*fp*	The pointer to the input file.
in	*add*	The address of the input file.

返回

The number of the numbers in the initial data file.

返回值

-1	If the given number of column is not coincided with that in the data file.

6.5 src/finite_volume/Godunov_solver_EUL_source.c 文件参考

This is an Eulerian Godunov scheme to solve 1-D Euler equations.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <time.h>
#include <stdbool.h>
#include "../include/var_struc.h"
#include "../include/Riemann_solver.h"
#include "../include/tools.h"
```

函数

- void Godunov_solver_EUL_source (const int m, struct cell_var CV, double *X[], double *cpu_time)
  This function use Godunov scheme to solve 1-D Euler equations of motion on Eulerian coordinate.

6.5.1 详细描述

This is an Eulerian Godunov scheme to solve 1-D Euler equations.

6.5.2 函数说明

6.5.2.1 Godunov_solver_EUL_source()

```
void Godunov_solver_EUL_source (
            const int m,
            struct cell_var CV,
            double * X[],
            double * cpu_time )
```

This function use Godunov scheme to solve 1-D Euler equations of motion on Eulerian coordinate.

参数

in	*m*	Number of the grids.
in,out	*CV*	Structural body of cell variable data.
in,out	*X[]*	Array of the coordinate data.
out	*cpu_time*	Array of the CPU time recording.

6.6 src/finite_volume/Godunov_solver_LAG_source.c 文件参考

This is a Lagrangian Godunov scheme to solve 1-D Euler equations.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <time.h>
#include <stdbool.h>
#include "../include/var_struc.h"
#include "../include/Riemann_solver.h"
#include "../include/tools.h"
```

函数

- void Godunov_solver_LAG_source (const int m, struct cell_var CV, double *X[], double *cpu_time)
  This function use Godunov scheme to solve 1-D Euler equations of motion on Lagrangian coordinate.

6.6.1 详细描述

This is a Lagrangian Godunov scheme to solve 1-D Euler equations.

6.6.2 函数说明

6.6.2.1 Godunov_solver_LAG_source()

```
void Godunov_solver_LAG_source (
            const int m,
            struct cell_var CV,
            double * X[],
            double * cpu_time )
```

This function use Godunov scheme to solve 1-D Euler equations of motion on Lagrangian coordinate.

参数

in	*m*	Number of the grids.
in,out	*CV*	Structural body of cell variable data.
in,out	*X[]*	Array of the coordinate data.
out	*cpu_time*	Array of the CPU time recording.

6.7 src/finite_volume/GRP_solver_EUL_source.c 文件参考

This is an Eulerian GRP scheme to solve 1-D Euler equations.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <time.h>
#include <stdbool.h>
#include "../include/var_struc.h"
#include "../include/Riemann_solver.h"
#include "../include/tools.h"
```

函数

- void GRP_solver_EUL_source (const int m, struct cell_var CV, double ∗X[ ], double ∗cpu_time)

  This function use GRP scheme to solve 1-D Euler equations of motion on Eulerian coordinate.

6.7.1 详细描述

This is an Eulerian GRP scheme to solve 1-D Euler equations.

6.7.2 函数说明

6.7.2.1 GRP_solver_EUL_source()

```
void GRP_solver_EUL_source (
            const int m,
            struct cell_var CV,
            double * X[],
            double * cpu_time )
```

This function use GRP scheme to solve 1-D Euler equations of motion on Eulerian coordinate.

参数

in	*m*	Number of the grids.
in,out	*CV*	Structural body of cell variable data.
in,out	*X[]*	Array of the coordinate data.
out	*cpu_time*	Array of the CPU time recording.

6.8 src/finite_volume/GRP_solver_LAG_source.c 文件参考

This is a Lagrangian GRP scheme to solve 1-D Euler equations.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <time.h>
#include <stdbool.h>
#include "../include/var_struc.h"
#include "../include/Riemann_solver.h"
#include "../include/tools.h"
```

函数

- void GRP_solver_LAG_source (const int m, struct cell_var CV, double *X[ ], double *cpu_time)

  This function use GRP scheme to solve 1-D Euler equations of motion on Lagrangian coordinate.

6.8.1 详细描述

This is a Lagrangian GRP scheme to solve 1-D Euler equations.

6.8.2 函数说明

6.8.2.1 GRP_solver_LAG_source()

```
void GRP_solver_LAG_source (
            const int m,
            struct cell_var CV,
            double * X[],
            double * cpu_time )
```

This function use GRP scheme to solve 1-D Euler equations of motion on Lagrangian coordinate.

参数

in	*m*	Number of the grids.
in,out	*CV*	Structural body of cell variable data.
in,out	*X[]*	Array of the coordinate data.
out	*cpu_time*	Array of the CPU time recording.

6.9 src/hydrocode/hydrocode.c 文件参考

This is a C file of the main function.

```
#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "../include/var_struc.h"
#include "../include/file_io.h"
#include "../include/finite_volume.h"
```

宏定义

- #define CV_INIT_MEM(v, N)

  N memory allocations to the initial fluid variable 'v' in the structural body cell_var.

函数

- int main (int argc, char ∗argv[ ])

  This is the main function which constructs the main structure of the Lagrangian/Eulerian hydrocode.

变量

- double config [N_CONF]

  Initial configuration data array.

6.9.1 详细描述

This is a C file of the main function.

6.9.2 宏定义说明

6.9.2.1 CV_INIT_MEM

```
#define CV_INIT_MEM(
                v,
                N )
```

值:

```
    do {                                \
    CV.v = (double **)malloc(N * sizeof(double *));         \
    CV.v[0] = FV0.v + 1;                                \
    for(k = 1; k < N; ++k)                              \
        {                               \
        CV.v[k] = (double *)malloc(m * sizeof(double));     \
        if(CV.v[k] == NULL)                 \
            {                           \
            printf("NOT enough memory! %s[%d]\n", #v, k);   \
            goto return_NULL;                   \
            }                           \
        }                               \
    } while (0)
```

N memory allocations to the initial fluid variable 'v' in the structural body cell_var.

6.9.3 函数说明

6.9.3.1 main()

```
int main (
              int argc,
              char * argv[] )
```

This is the main function which constructs the main structure of the Lagrangian/Eulerian hydrocode.

参数

in	*argc*	ARGument counter.
in	*argv*	ARGument values. • argv[1]: Folder name of test example (input path). • argv[2]: Folder name of numerical results (output path). • argv[3]: Dimensionality (= 1). • argv[4]: Order of numerical scheme[_scheme name] (= 1[_Riemann_exact] or 2[_GRP]). • ~~argv[5]: Lagrangian/Eulerian coordinate framework (= LAG or EUL).~~
		• argv[6,7,...]: Configuration supplement config[n]=(double)C (= n=C).

返回

Program exit status code.

6.9.4 变量说明

6.9.4.1 config

```
double config[N_CONF]
```

Initial configuration data array.

6.10 src/include/file_io.h 文件参考

This file is the header file that controls data input and output.

函数

- void example_io (const char *example, char *add_mkdir, const int i_or_o)

  This function produces folder path for data input or output.
- int flu_var_count (FILE *fp, const char *add)

  This function counts how many numbers are there in the initial data file.
- void _1D_initialize (const char *name, struct flu_var *FV0)

  This function reads the 1D initial data file of velocity/pressure/density.
- void _1D_file_write (const int m, const int N, struct cell_var CV, double *X[ ], const double *cpu_time, const char *name)

  This function write the solution into output files.
- void configurate (const char *name)

  This function controls configuration data reading and validation.

6.10.1 详细描述

This file is the header file that controls data input and output.

This header file declares functions in the folder 'file_io'.

6.10.2 函数说明

6.10.2.1 _1D_file_write()

```
void _1D_file_write (
            const int m,
            const int N,
            struct cell_var CV,
            double * X[],
            const double * cpu_time,
            const char * name )
```

This function write the solution into output files.

注解

It is quite simple so there will be no more comments.

参数

in	*m*	The number of spatial points in the output data.
in	*N*	The number of time steps in the output data.
in	*CV*	Structural body of grid variable data.
in	*X[]*	Array of the coordinate data.
in	*cpu_time*	Array of the CPU time recording.
in	*name*	Name of the numerical results.

6.10.2.2 _1D_initialize()

```
void _1D_initialize (
            const char * name,
            struct flu_var * FV0 )
```

This function reads the 1D initial data file of velocity/pressure/density.

The function initialize the extern pointer FV0->RHO/U/P pointing to the position of a block of memory consisting (m+1) variables* of type double. The value of first of these variables is m. The following m variables are the initial value.

参数

in	*name*	Name of the test example.
out	*FV0*	Structural body pointer of initial data array pointer.

6.10.2.3 configurate()

```
void configurate (
            const char * add_in )
```

This function controls configuration data reading and validation.

The parameters in the configuration data file refer to 'doc/config.csv'.

参数

in	*add_in*	Adress of the initial data folder of the test example.

6.10.2.4 example_io()

```
void example_io (
            const char * example,
            char * add_mkdir,
            const int i_or_o )
```

This function produces folder path for data input or output.

参数

in	*example*	Name of the test example/numerical results.
out	*add_mkdir*	Folder path for data input or output.
in	*i_or_o*	Conversion parameters for data input/output. • 0: data output. • else (e.g. 1): data input.

6.10.2.5 flu_var_count()

```
int flu_var_count (
            FILE * fp,
            const char * add )
```

This function counts how many numbers are there in the initial data file.

参数

in	*fp*	The pointer to the input file.
in	*add*	The address of the input file.

返回

The number of the numbers in the initial data file.

返回值

-1	If the given number of column is not coincided with that in the data file.

6.11 file_io.h

浏览该文件的文档.

```
1
7 #ifndef FILEIO_H
8 #define FILEIO_H
9
10 // io_control.c
11 void example_io(const char * example, char * add_mkdir, const int i_or_o);
12
13 int flu_var_count(FILE * fp, const char * add);
14
15
16 // _1D_file_in.c
17 void _1D_initialize(const char * name, struct flu_var * FV0);
18
19
20 // _1D_file_out.c
21 void _1D_file_write(const int m, const int N, struct cell_var CV, double * X[],
22                     const double * cpu_time, const char * name);
23
24
25 // config_in.c
26 void configurate(const char * name);
27
28 #endif
```

6.12 src/include/finite_volume.h 文件参考

This file is the header file of Lagrangian/Eulerian hydrocode in finite volume framework.

```
#include "../include/var_struc.h"
```

函数

- void Godunov_solver_LAG_source (const int m, struct cell_var CV, double ∗X[ ], double ∗cpu_time)
  This function use Godunov scheme to solve 1-D Euler equations of motion on Lagrangian coordinate.
- void GRP_solver_LAG_source (const int m, struct cell_var CV, double ∗X[ ], double ∗cpu_time)
  This function use GRP scheme to solve 1-D Euler equations of motion on Lagrangian coordinate.
- void Godunov_solver_EUL_source (const int m, struct cell_var CV, double ∗X[ ], double ∗cpu_time)
  This function use Godunov scheme to solve 1-D Euler equations of motion on Eulerian coordinate.
- void GRP_solver_EUL_source (const int m, struct cell_var CV, double ∗X[ ], double ∗cpu_time)
  This function use GRP scheme to solve 1-D Euler equations of motion on Eulerian coordinate.

6.12.1 详细描述

This file is the header file of Lagrangian/Eulerian hydrocode in finite volume framework.

This header file declares functions in the folder 'finite_volume'.

6.12.2 函数说明

6.12.2.1 Godunov_solver_EUL_source()

```
void Godunov_solver_EUL_source (
            const int m,
            struct cell_var CV,
            double * X[],
            double * cpu_time )
```

This function use Godunov scheme to solve 1-D Euler equations of motion on Eulerian coordinate.

参数

in	*m*	Number of the grids.
in,out	*CV*	Structural body of cell variable data.
in,out	*X[]*	Array of the coordinate data.
out	*cpu_time*	Array of the CPU time recording.

6.12.2.2 Godunov_solver_LAG_source()

```
void Godunov_solver_LAG_source (
            const int m,
            struct cell_var CV,
            double * X[],
            double * cpu_time )
```

This function use Godunov scheme to solve 1-D Euler equations of motion on Lagrangian coordinate.

参数

in	*m*	Number of the grids.
in,out	*CV*	Structural body of cell variable data.
in,out	*X[]*	Array of the coordinate data.
out	*cpu_time*	Array of the CPU time recording.

6.12.2.3 GRP_solver_EUL_source()

```
void GRP_solver_EUL_source (
            const int m,
            struct cell_var CV,
```

```
            double * X[],
            double * cpu_time )
```

This function use GRP scheme to solve 1-D Euler equations of motion on Eulerian coordinate.

参数

in	*m*	Number of the grids.
in,out	*CV*	Structural body of cell variable data.
in,out	*X[]*	Array of the coordinate data.
out	*cpu_time*	Array of the CPU time recording.

6.12.2.4 GRP_solver_LAG_source()

```
void GRP_solver_LAG_source (
            const int m,
            struct cell_var CV,
            double * X[],
            double * cpu_time )
```

This function use GRP scheme to solve 1-D Euler equations of motion on Lagrangian coordinate.

参数

in	*m*	Number of the grids.
in,out	*CV*	Structural body of cell variable data.
in,out	*X[]*	Array of the coordinate data.
out	*cpu_time*	Array of the CPU time recording.

6.13 finite_volume.h

浏览该文件的文档.

```
1
7 #ifndef FINITEVOLUME_H
8 #define FINITEVOLUME_H
9
10 #include "../include/var_struc.h"
11
12 void Godunov_solver_LAG_source
13 (const int m, struct cell_var CV, double * X[], double * cpu_time);
14
15 void GRP_solver_LAG_source
16 (const int m, struct cell_var CV, double * X[], double * cpu_time);
17
18 void Godunov_solver_EUL_source
19 (const int m, struct cell_var CV, double * X[], double * cpu_time);
20
21 void GRP_solver_EUL_source
22 (const int m, struct cell_var CV, double * X[], double * cpu_time);
23
24 #endif
```

6.14 src/include/Riemann_solver.h 文件参考

This file is the header file of several Riemann solvers and GRP solvers.

宏定义

- #define Riemann_solver_exact Riemann_solver_exact_Ben

函数

- double Riemann_solver_exact_Ben (double *U_star, double *P_star, const double gamma, const double u_L, const double u_R, const double p_L, const double p_R, const double c_L, const double c_R, _Bool *CRW, const double eps, const double tol, const int N)

  EXACT RIEMANN SOLVER FOR A γ -Law Gas
- double Riemann_solver_exact_Toro (double *U_star, double *P_star, const double gamma, const double U_l, const double U_r, const double P_l, const double P_r, const double c_l, const double c_r, _Bool *CRW, const double eps, const double tol, const int N)

  EXACT RIEMANN SOLVER FOR THE EULER EQUATIONS
- void linear_GRP_solver_LAG (double *dire, double *mid, const double rho_L, const double rho_R, const double s_rho_L, const double s_rho_R, const double u_L, const double u_R, const double s_u_L, const double s_u_R, const double p_L, const double p_R, const double s_p_L, const double s_p_R, const double gamma, const double eps, const double atc)

  A Lagrangian GRP solver for unsteady compressible inviscid flow in one space dimension.
- void linear_GRP_solver_Edir (double *direvative, double *mid, const double rho_L, const double rho_R, const double s_rho_L, const double s_rho_R, const double u_L, const double u_R, const double s_u_L, const double s_u_R, const double p_L, const double p_R, const double s_p_L, const double s_p_R, const double gamma, const double eps)

  A direct Eulerian GRP solver for unsteady compressible inviscid flow in one space dimension.

6.14.1 详细描述

This file is the header file of several Riemann solvers and GRP solvers.

This header file declares functions in the folder 'Riemann_solver'.

6.14.2 宏定义说明

6.14.2.1 Riemann_solver_exact

```
#define Riemann_solver_exact Riemann_solver_exact_Ben
```

6.14.3 函数说明

6.14.3.1 linear_GRP_solver_Edir()

```
void linear_GRP_solver_Edir (
            double * direvative,
            double * mid,
            const double rho_L,
            const double rho_R,
            const double s_rho_L,
            const double s_rho_R,
            const double u_L,
            const double u_R,
            const double s_u_L,
            const double s_u_R,
            const double p_L,
            const double p_R,
            const double s_p_L,
            const double s_p_R,
            const double gamma,
            const double eps )
```

A direct Eulerian GRP solver for unsteady compressible inviscid flow in one space dimension.

参数

out	*direvative*	the temporal derivative of fluid variables. [rho, u, p]_t
out	*mid*	the Riemann solutions. [rho_star, u_star, p_star]
in	*rho_L,u_L,p_L*	Left States.
in	*rho_R,u_R,p_R*	Right States.
in	*s_rho_L,s_u_L,s_p_L*	Left x-spatial derivatives.
in	*s_rho_R,s_u_R,s_p_R*	Right x-spatial derivatives.
in	*gamma*	the constant of the perfect gas.
in	*eps*	the largest value could be seen as zero.

Reference

Theory is found in Reference [1].
[1] M. Ben-Artzi, J. Li & G. Warnecke, A direct Eulerian GRP scheme for compressible fluid flows, Journal of Computational Physics, 218.1: 19-43, 2006.

6.14.3.2 linear_GRP_solver_LAG()

```
void linear_GRP_solver_LAG (
            double * dire,
            double * mid,
            const double rho_L,
            const double rho_R,
            const double s_rho_L,
```

```
            const double s_rho_R,
            const double u_L,
            const double u_R,
            const double s_u_L,
            const double s_u_R,
            const double p_L,
            const double p_R,
            const double s_p_L,
            const double s_p_R,
            const double gamma,
            const double eps,
            const double atc )
```

A Lagrangian GRP solver for unsteady compressible inviscid flow in one space dimension.

参数

out	*dire*	the temporal derivative of fluid variables. [rho_L, u, p, rho_R]_t
out	*mid*	the Riemann solutions. [rho_star_L, u_star, p_star, rho_star_R]
in	*rho_L,u_L,p_L*	Left States.
in	*rho_R,u_R,p_R*	Right States.
in	*s_rho_L,s_u_L,s_p_L*	Left ξ -Lagrangian spatial derivatives.
in	*s_rho_R,s_u_R,s_p_R*	Right ξ -Lagrangian spatial derivatives.
in	*gamma*	the constant of the perfect gas.
in	*eps*	the largest value could be seen as zero.
in	*atc*	Parameter that determines the solver type. • INFINITY: acoustic approximation • eps: GRP solver(nonlinear + acoustic case) • -0.0: GRP solver(only nonlinear case)

Reference

Theory is found in Reference [1].
[1] M. Ben-Artzi & J. Falcovitz, A second-order Godunov-type scheme for compressible fluid dynamics, Journal of Computational Physics, 55.1: 1-32, 1984

6.14.3.3 Riemann_solver_exact_Ben()

```
double Riemann_solver_exact_Ben (
            double * U_star,
            double * P_star,
            const double gamma,
            const double u_L,
            const double u_R,
            const double p_L,
```

```
            const double p_R,
            const double c_L,
            const double c_R,
            _Bool * CRW,
            const double eps,
            const double tol,
            const int N )
```

EXACT RIEMANN SOLVER FOR A γ -Law Gas

The purpose of this function is to solve the Riemann problem exactly, for the time dependent one dimensional Euler equations for a γ -law gas.

参数

out	*U_star,P_star*	Velocity/Pressure in star region.
in	*u_L,p_L,c_L*	Initial Velocity/Pressure/sound_speed on left state.
in	*u_R,p_R,c_R*	Initial Velocity/Pressure/sound_speed on right state.
in	*gamma*	Ratio of specific heats.
out	*CRW*	Centred Rarefaction Wave (CRW) Indicator of left and right waves. • true: CRW • false: Shock wave
in	*eps*	The largest value can be seen as zero.
in	*tol*	Condition value of 'gap' at the end of the iteration.
in	*N*	Maximum iteration step.

返回

gap: Relative pressure change after the last iteration.

Reference

Theory is found in Appendix C of Reference [1].
[1] M. Ben-Artzi & J. Falcovitz, ”Generalized Riemann problems in computational fluid dynamics”, Cambridge University Press, 2003

6.14.3.4 Riemann_solver_exact_Toro()

```
double Riemann_solver_exact_Toro (
            double * U_star,
            double * P_star,
            const double gamma,
            const double U_l,
            const double U_r,
            const double P_l,
            const double P_r,
            const double c_l,
```

```
const double c_r,
_Bool * CRW,
const double eps,
const double tol,
const int N )
```

EXACT RIEMANN SOLVER FOR THE EULER EQUATIONS

The purpose of this function is to solve the Riemann problem exactly, for the time dependent one dimensional Euler equations for an ideal gas.

参数

out	*U_star,P_star*	Velocity/Pressure in star region.
in	*U_l,P_l,c_l*	Initial Velocity/Pressure/sound_speed on left state.
in	*U_r,P_r,c_r*	Initial Velocity/Pressure/sound_speed on right state.
in	*gamma*	Ratio of specific heats.
out	*CRW*	Centred Rarefaction Wave (CRW) Indicator of left and right waves. • true: CRW • false: Shock wave
in	*eps*	The largest value can be seen as zero.
in	*tol*	Condition value of 'gap' at the end of the iteration.
in	*N*	Maximum iteration step.

返回

gap: Relative pressure change after the last iteration.

作者

E. F. Toro

日期

February 1st 1999

Reference

Theory is found in Chapter 4 of Reference [1].
[1] Toro, E. F., ”Riemann Solvers and Numerical Methods for Fluid Dynamics”, Springer-Verlag, Second Edition, 1999

6.15 Riemann_solver.h

浏览该文件的文档.

```
1
7 #ifndef RIEMANNSOLVER_H
8 #define RIEMANNSOLVER_H
9
10 double Riemann_solver_exact_Ben(double * U_star, double * P_star, const double gamma,
11                 const double u_L, const double u_R, const double p_L, const double p_R,
12                 const double c_L, const double c_R, _Bool * CRW,
13                 const double eps, const double tol, const int N);
14
15 double Riemann_solver_exact_Toro(double * U_star, double * P_star, const double gamma,
16                  const double U_l, const double U_r, const double P_l, const double P_r,
17                  const double c_l, const double c_r, _Bool * CRW,
18                  const double eps, const double tol, const int N);
19
20 void linear_GRP_solver_LAG
21 (double * dire, double * mid,
22  const double rho_L, const double rho_R, const double s_rho_L, const double s_rho_R,
23  const double   u_L, const double   u_R, const double   s_u_L, const double   s_u_R,
24  const double   p_L, const double   p_R, const double   s_p_L, const double   s_p_R,
25  const double gamma, const double eps, const double  atc);
26
27 void linear_GRP_solver_Edir
28 (double * direvative, double * mid,
29  const double rho_L, const double rho_R, const double s_rho_L, const double s_rho_R,
30  const double   u_L, const double   u_R, const double   s_u_L, const double   s_u_R,
31  const double   p_L, const double   p_R, const double   s_p_L, const double   s_p_R,
32  const double gamma, const double eps);
33
34 #ifndef Riemann_solver_exact
35 #define Riemann_solver_exact Riemann_solver_exact_Ben
36 #endif
37
38 #endif
```

6.16 src/include/tools.h 文件参考

This file is the header file of several independent tool functions.

函数

- void DispPro (const double pro, const int step)

  This function print a progress bar on one line of standard output.
- int CreateDir (const char *pPath)

  This is a function that recursively creates folders.
- int rinv (double a[ ], const int n)

  A function to caculate the inverse of the input square matrix.
- double minmod2 (double s_L, double s_R)

  Minmod limiter function of two variables.
- double minmod3 (double s_L, double s_R, double s_m)

  Minmod limiter function of three variables.

6.16.1 详细描述

This file is the header file of several independent tool functions.

This header file declares functions in the folder 'tools',

6.16.2 函数说明

6.16.2.1 CreateDir()

```
int CreateDir (
            const char * pPath )
```

This is a function that recursively creates folders.

参数

in	*pPath*	Pointer to the folder Path.

返回

Folder Creation Status.

返回值

-1	The path folder already exists and is readable.
0	Readable path folders are created recursively.
1	The path folder is not created properly.

6.16.2.2 DispPro()

```
void DispPro (
            const double pro,
            const int step )
```

This function print a progress bar on one line of standard output.

参数

in	*pro*	Numerator of percent that the process has completed.
in	*step*	Number of time steps.

6.16.2.3 minmod2()

```
double minmod2 (
            double s_L,
            double s_R )  [inline]
```

Minmod limiter function of two variables.

6.16.2.4 minmod3()

```
double minmod3 (
            double s_L,
            double s_R,
            double s_m )  [inline]
```

Minmod limiter function of three variables.

6.16.2.5 rinv()

```
int rinv (
            double a[],
            const int n )
```

A function to caculate the inverse of the input square matrix.

参数

in,out	*a*	The pointer of the input/output square matrix.
in	*n*	The order of the input/output square matrix.

返回

Matrix is invertible or not.

返回值

0	No inverse matrix
1	Invertible matrix

6.17 tools.h

浏览该文件的文档.

```
1
7 #ifndef TOOLS_H
8 #define TOOLS_H
9
10 // sys_pro.c
11 void DispPro(const double pro, const int step);
12
13 int CreateDir(const char* pPath);
14
15
16 // math_algo.c
```

```
17 int rinv(double a[], const int n);
18
19
23 inline double minmod2(double s_L, double s_R)
24 {
25     if(s_L * s_R < 0.0)
26     return 0.0;
27     else if(fabs(s_R) < fabs(s_L))
28     return s_R;
29     else
30     return s_L;
31 }
32
36 inline double minmod3(double s_L, double s_R, double s_m)
37 {
38     if(s_L * s_m < 0.0 || s_R * s_m < 0.0)
39     return 0.0;
40     else if(fabs(s_m) < fabs(s_L) && fabs(s_m) < fabs(s_R))
41     return s_m;
42     else if(fabs(s_R) < fabs(s_L))
43     return s_R;
44     else
45     return s_L;
46 }
47
48 #endif
```

6.18 src/include/var_struc.h 文件参考

结构体

- struct flu_var

  Pointer structural body of fluid variables.
- struct cell_var

  Pointer structural body of variables on computational cells.

宏定义

- #define EPS 1e-9

  If the system does not set, the default largest value can be seen as zero is EPS.
- #define N_CONF 400

  Define the number of configuration parameters.

变量

- double config [ ]

  Initial configuration data array.

6.18.1 宏定义说明

6.18.1.1 EPS

```
#define EPS 1e-9
```

If the system does not set, the default largest value can be seen as zero is EPS.

6.18.1.2 N_CONF

```
#define N_CONF 400
```

Define the number of configuration parameters.

6.18.2 变量说明

6.18.2.1 config

```
double config[]  [extern]
```

Initial configuration data array.

6.19 var_struc.h

浏览该文件的文档.

```
1 #ifndef VARSTRUC_H
2 #define VARSTRUC_H
3
5 #ifndef EPS
6 #define EPS 1e-9
7 #endif
8
10 #ifndef N_CONF
11 #define N_CONF 400
12 #endif
13
14 extern double config[];
15
17 struct flu_var {
18     double *RHO, *U, *P;
19 };
20
22 struct cell_var {
23     double **RHO, **U, **V, **P, **E;
24 };
25
26 #endif
```

6.20 src/Riemann_solver/linear_GRP_solver_Edir.c 文件参考

This is a direct Eulerian GRP solver for compressible inviscid flow in Li's paper.

```
#include <math.h>
#include <stdio.h>
#include "../include/Riemann_solver.h"
```

函数

- void linear_GRP_solver_Edir (double *direvative, double *mid, const double rho_L, const double rho_R, const double s_rho_L, const double s_rho_R, const double u_L, const double u_R, const double s_u_L, const double s_u_R, const double p_L, const double p_R, const double s_p_L, const double s_p_R, const double gamma, const double eps)

  A direct Eulerian GRP solver for unsteady compressible inviscid flow in one space dimension.

6.20.1 详细描述

This is a direct Eulerian GRP solver for compressible inviscid flow in Li's paper.

6.20.2 函数说明

6.20.2.1 linear_GRP_solver_Edir()

```
void linear_GRP_solver_Edir (
            double * direvative,
            double * mid,
            const double rho_L,
            const double rho_R,
            const double s_rho_L,
            const double s_rho_R,
            const double u_L,
            const double u_R,
            const double s_u_L,
            const double s_u_R,
            const double p_L,
            const double p_R,
            const double s_p_L,
            const double s_p_R,
            const double gamma,
            const double eps )
```

A direct Eulerian GRP solver for unsteady compressible inviscid flow in one space dimension.

参数

out	*direvative*	the temporal derivative of fluid variables. [rho, u, p]_t
out	*mid*	the Riemann solutions. [rho_star, u_star, p_star]
in	*rho_L,u_L,p_L*	Left States.
in	*rho_R,u_R,p_R*	Right States.
in	*s_rho_L,s_u_L,s_p_L*	Left x-spatial derivatives.
in	*s_rho_R,s_u_R,s_p_R*	Right x-spatial derivatives.
in	*gamma*	the constant of the perfect gas.
in	*eps*	the largest value could be seen as zero.

Reference

Theory is found in Reference [1].
[1] M. Ben-Artzi, J. Li & G. Warnecke, A direct Eulerian GRP scheme for compressible fluid flows, Journal of Computational Physics, 218.1: 19-43, 2006.

6.21 src/Riemann_solver/linear_GRP_solver_LAG.c 文件参考

This is a Lagrangian GRP solver for compressible inviscid flow in Ben-Artzi's book.

```
#include <math.h>
#include <stdio.h>
#include "../include/Riemann_solver.h"
```

函数

- void linear_GRP_solver_LAG (double *dire, double *mid, const double rho_L, const double rho_R, const double s_rho_L, const double s_rho_R, const double u_L, const double u_R, const double s_u_L, const double s_u_R, const double p_L, const double p_R, const double s_p_L, const double s_p_R, const double gamma, const double eps, const double atc)

  A Lagrangian GRP solver for unsteady compressible inviscid flow in one space dimension.

6.21.1 详细描述

This is a Lagrangian GRP solver for compressible inviscid flow in Ben-Artzi's book.

6.21.2 函数说明

6.21.2.1 linear_GRP_solver_LAG()

```
void linear_GRP_solver_LAG (
            double * dire,
            double * mid,
            const double rho_L,
            const double rho_R,
            const double s_rho_L,
            const double s_rho_R,
            const double u_L,
            const double u_R,
            const double s_u_L,
            const double s_u_R,
            const double p_L,
            const double p_R,
            const double s_p_L,
            const double s_p_R,
            const double gamma,
            const double eps,
            const double atc )
```

A Lagrangian GRP solver for unsteady compressible inviscid flow in one space dimension.

参数

out	*dire*	the temporal derivative of fluid variables. [rho_L, u, p, rho_R]_t
out	*mid*	the Riemann solutions. [rho_star_L, u_star, p_star, rho_star_R]
in	*rho_L,u_L,p_L*	Left States.
in	*rho_R,u_R,p_R*	Right States.
in	*s_rho_L,s_u_L,s_p_L*	Left ξ -Lagrangian spatial derivatives.
in	*s_rho_R,s_u_R,s_p_R*	Right ξ -Lagrangian spatial derivatives.
in	*gamma*	the constant of the perfect gas.
in	*eps*	the largest value could be seen as zero.
in	*atc*	Parameter that determines the solver type. • INFINITY: acoustic approximation • eps: GRP solver(nonlinear + acoustic case) • -0.0: GRP solver(only nonlinear case)

Reference

Theory is found in Reference [1].
[1] M. Ben-Artzi & J. Falcovitz, A second-order Godunov-type scheme for compressible fluid dynamics, Journal of Computational Physics, 55.1: 1-32, 1984

6.22 src/Riemann_solver/Riemann_solver_exact_Ben.c 文件参考

This is an exact Riemann solver in Ben-Artzi's book.

```
#include <math.h>
#include <stdio.h>
#include <stdbool.h>
```

函数

- double Riemann_solver_exact_Ben (double *U_star, double *P_star, const double gamma, const double u_L, const double u_R, const double p_L, const double p_R, const double c_L, const double c_R, _Bool *CRW, const double eps, const double tol, const int N)

  EXACT RIEMANN SOLVER FOR A γ -Law Gas

6.22.1 详细描述

This is an exact Riemann solver in Ben-Artzi's book.

6.22.2 函数说明

6.22.2.1 Riemann_solver_exact_Ben()

```
double Riemann_solver_exact_Ben (
            double * U_star,
            double * P_star,
            const double gamma,
            const double u_L,
            const double u_R,
            const double p_L,
            const double p_R,
            const double c_L,
            const double c_R,
            _Bool * CRW,
            const double eps,
            const double tol,
            const int N )
```

EXACT RIEMANN SOLVER FOR A γ -Law Gas

The purpose of this function is to solve the Riemann problem exactly, for the time dependent one dimensional Euler equations for a γ -law gas.

参数

out	*U_star,P_star*	Velocity/Pressure in star region.
in	*u_L,p_L,c_L*	Initial Velocity/Pressure/sound_speed on left state.
in	*u_R,p_R,c_R*	Initial Velocity/Pressure/sound_speed on right state.
in	*gamma*	Ratio of specific heats.
out	*CRW*	Centred Rarefaction Wave (CRW) Indicator of left and right waves. • true: CRW • false: Shock wave
in	*eps*	The largest value can be seen as zero.
in	*tol*	Condition value of 'gap' at the end of the iteration.
in	*N*	Maximum iteration step.

返回

gap: Relative pressure change after the last iteration.

Reference

Theory is found in Appendix C of Reference [1].
[1] M. Ben-Artzi & J. Falcovitz, ”Generalized Riemann problems in computational fluid dynamics”, Cambridge University Press, 2003

6.23 src/Riemann_solver/Riemann_solver_exact_Toro.c 文件参考

This is an exact Riemann solver in Toro's book.

```
#include <math.h>
#include <stdio.h>
#include <stdbool.h>
```

函数

- double Riemann_solver_exact_Toro (double *U_star, double *P_star, const double gamma, const double U_l, const double U_r, const double P_l, const double P_r, const double c_l, const double c_r, _Bool *CRW, const double eps, const double tol, const int N)

  EXACT RIEMANN SOLVER FOR THE EULER EQUATIONS

6.23.1 详细描述

This is an exact Riemann solver in Toro's book.

6.23.2 函数说明

6.23.2.1 Riemann_solver_exact_Toro()

```
double Riemann_solver_exact_Toro (
            double * U_star,
            double * P_star,
            const double gamma,
            const double U_l,
            const double U_r,
            const double P_l,
            const double P_r,
            const double c_l,
            const double c_r,
            _Bool * CRW,
            const double eps,
            const double tol,
            const int N )
```

EXACT RIEMANN SOLVER FOR THE EULER EQUATIONS

The purpose of this function is to solve the Riemann problem exactly, for the time dependent one dimensional Euler equations for an ideal gas.

参数

out	*U_star,P_star*	Velocity/Pressure in star region.
in	*U_l,P_l,c_l*	Initial Velocity/Pressure/sound_speed on left state.

参数

in	*U_r,P_r,c_r*	Initial Velocity/Pressure/sound_speed on right state.
in	*gamma*	Ratio of specific heats.
out	*CRW*	Centred Rarefaction Wave (CRW) Indicator of left and right waves. • true: CRW • false: Shock wave
in	*eps*	The largest value can be seen as zero.
in	*tol*	Condition value of 'gap' at the end of the iteration.
in	*N*	Maximum iteration step.

返回

gap: Relative pressure change after the last iteration.

作者

E. F. Toro

日期

February 1st 1999

Reference

Theory is found in Chapter 4 of Reference [1].
[1] Toro, E. F., "Riemann Solvers and Numerical Methods for Fluid Dynamics", Springer-Verlag, Second Edition, 1999

6.24 src/tools/math_algo.c 文件参考

There are some mathematical algorithms.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
```

函数

- int rinv (double a[ ], const int n)

  A function to caculate the inverse of the input square matrix.

6.24.1 详细描述

There are some mathematical algorithms.

6.24.2 函数说明

6.24.2.1 rinv()

```
int rinv (
            double a[],
            const int n )
```

A function to caculate the inverse of the input square matrix.

参数

in,out	*a*	The pointer of the input/output square matrix.
in	*n*	The order of the input/output square matrix.

返回

Matrix is invertible or not.

返回值

0	No inverse matrix
1	Invertible matrix

6.25 src/tools/str_num_common.c 文件参考

This is a set of common functions for string and number processing.

```
#include <math.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
```

函数

- static int format_string (char *str)

  This function examine whether a string represents a real number.
- static double str2num (char *number)

  This function transform a string consisting '1', '2', ..., and '.' into the real number that it represents.

6.25.1 详细描述

This is a set of common functions for string and number processing.

6.25.2 函数说明

6.25.2.1 format_string()

```
static int format_string (
            char * str )  [static]
```

This function examine whether a string represents a real number.

Transform the string represents a negtive number into a string represents a positive one and return its' sign. It returns 0 if the string do not represents a real number. After calling this function, there will be only one 'e' in the string, and the only position for '-' is behind 'e', and there can be only one dot in the string and the only position for it in before 'e'.

参数

in	*str*	String to be examined.

返回

The sign of the number represented by the string.

返回值

1	Positive number.
-1	Negative number.
0	Not a number.

弃用 This function has been replaced by the variable 'errno' in the standard Library <errno.h>.

6.25.2.2 str2num()

```
static double str2num (
            char * number )  [static]
```

This function transform a string consisting '1', '2', ..., and '.' into the real number that it represents.

参数

in	*number*	String of the real number.

返回

The real number that the string represents.

弃用 This function has been replaced by the 'strtod()' function in the standard Library <stdio.h>.

6.26 src/tools/sys_pro.c 文件参考

There are some system processing programs.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <limits.h>
```

函数

- void DispPro (const double pro, const int step)

  This function print a progress bar on one line of standard output.
- int CreateDir (const char *pPath)

  This is a function that recursively creates folders.

6.26.1 详细描述

There are some system processing programs.

6.26.2 函数说明

6.26.2.1 CreateDir()

```
int CreateDir (
            const char * pPath )
```

This is a function that recursively creates folders.

参数

in	*pPath*	Pointer to the folder Path.

返回

Folder Creation Status.

返回值

-1	The path folder already exists and is readable.
0	Readable path folders are created recursively.
1	The path folder is not created properly.

6.26.2.2 DispPro()

```
void DispPro (
            const double pro,
            const int step )
```

This function print a progress bar on one line of standard output.

参数

in	*pro*	Numerator of percent that the process has completed.
in	*step*	Number of time steps.

Index