

HydroCODE

制作者 Doxygen 1.9.3



<b>1 1D Godunov/GRP scheme for Lagrangian/Eulerian hydrodynamics</b>	<b>1</b>
1.1 File directory . . . . .	1
1.2 Program structure . . . . .	1
1.3 Program exit status code . . . . .	1
1.4 Compile environment . . . . .	2
1.5 Usage description . . . . .	2
<b>2 弃用列表</b>	<b>3</b>
<b>3 结构体索引</b>	<b>5</b>
3.1 结构体 . . . . .	5
<b>4 文件索引</b>	<b>7</b>
4.1 文件列表 . . . . .	7
<b>5 结构体说明</b>	<b>9</b>
5.1 cell_var结构体 参考 . . . . .	9
5.1.1 详细描述 . . . . .	9
5.1.2 结构体成员变量说明 . . . . .	9
5.1.2.1 E . . . . .	9
5.1.2.2 P . . . . .	10
5.1.2.3 RHO . . . . .	10
5.1.2.4 U . . . . .	10
5.1.2.5 V . . . . .	10
5.2 flu_var结构体 参考 . . . . .	10
5.2.1 详细描述 . . . . .	10
5.2.2 结构体成员变量说明 . . . . .	10
5.2.2.1 P . . . . .	11
5.2.2.2 RHO . . . . .	11
5.2.2.3 U . . . . .	11
<b>6 文件说明</b>	<b>13</b>
6.1 src/file_io/_1D.file_in.c 文件参考 . . . . .	13
6.1.1 详细描述 . . . . .	13
6.1.2 宏定义说明 . . . . .	13
6.1.2.1 STR_FLU_INI . . . . .	14
6.1.3 函数说明 . . . . .	14
6.1.3.1 _1D.flu_var_read() . . . . .	14
6.1.3.2 _1D.initialize() . . . . .	14
6.2 src/file_io/_1D.file_out.c 文件参考 . . . . .	15
6.2.1 详细描述 . . . . .	15
6.2.2 宏定义说明 . . . . .	15
6.2.2.1 PRINT_NC . . . . .	15
6.2.3 函数说明 . . . . .	16

6.2.3.1 _1D_file_write()	16
6.3 src/file_io/config_in.c 文件参考	16
6.3.1 详细描述	17
6.3.2 函数说明	17
6.3.2.1 config_check()	17
6.3.2.2 config_read()	17
6.3.2.3 configurate()	18
6.4 src/file_io/io_control.c 文件参考	18
6.4.1 详细描述	18
6.4.2 函数说明	18
6.4.2.1 example_io()	18
6.4.2.2 flu_var_count()	19
6.5 src/finite_volume/Godunov_solver_EUL_source.c 文件参考	19
6.5.1 详细描述	20
6.5.2 函数说明	20
6.5.2.1 Godunov_solver_EUL_source()	20
6.6 src/finite_volume/Godunov_solver_LAG_source.c 文件参考	20
6.6.1 详细描述	21
6.6.2 函数说明	21
6.6.2.1 Godunov_solver_LAG_source()	21
6.7 src/finite_volume/GRP_solver_EUL_source.c 文件参考	21
6.7.1 详细描述	21
6.7.2 函数说明	22
6.7.2.1 GRP_solver_EUL_source()	22
6.8 src/finite_volume/GRP_solver_LAG_source.c 文件参考	22
6.8.1 详细描述	22
6.8.2 函数说明	22
6.8.2.1 GRP_solver_LAG_source()	23
6.9 src/hydrocode/hydrocode.c 文件参考	23
6.9.1 详细描述	24
6.9.2 宏定义说明	24
6.9.2.1 CV_INIT_MEM	24
6.9.3 函数说明	24
6.9.3.1 main()	24
6.9.4 变量说明	25
6.9.4.1 config	25
6.10 src/include/file_io.h 文件参考	25
6.10.1 详细描述	25
6.10.2 函数说明	25
6.10.2.1 _1D_file_write()	26
6.10.2.2 _1D_initialize()	26
6.10.2.3 configurate()	26

---

6.10.2.4 example_io()	27
6.10.2.5 flu_var_count()	27
6.11 file_io.h	28
6.12 src/include/finite_volume.h 文件参考	28
6.12.1 详细描述	28
6.12.2 函数说明	29
6.12.2.1 Godunov_solver_EUL_source()	29
6.12.2.2 Godunov_solver_LAG_source()	29
6.12.2.3 GRP_solver_EUL_source()	29
6.12.2.4 GRP_solver_LAG_source()	30
6.13 finite_volume.h	30
6.14 src/include/Riemann_solver.h 文件参考	31
6.14.1 详细描述	31
6.14.2 宏定义说明	31
6.14.2.1 Riemann_solver_exact	31
6.14.3 函数说明	31
6.14.3.1 linear_GRP_solver_Edir()	32
6.14.3.2 linear_GRP_solver_LAG()	32
6.14.3.3 Riemann_solver_exact_Ben()	33
6.14.3.4 Riemann_solver_exact_Toro()	34
6.15 Riemann_solver.h	36
6.16 src/include/tools.h 文件参考	36
6.16.1 详细描述	36
6.16.2 函数说明	37
6.16.2.1 CreateDir()	37
6.16.2.2 DispPro()	37
6.16.2.3 minmod2()	37
6.16.2.4 minmod3()	38
6.16.2.5 rinu()	38
6.17 tools.h	38
6.18 src/include/var_struct.h 文件参考	39
6.18.1 宏定义说明	39
6.18.1.1 EPS	39
6.18.1.2 N_CONF	40
6.18.2 变量说明	40
6.18.2.1 config	40
6.19 var_struct.h	40
6.20 src/Riemann_solver/linear_GRP_solver_Edir.c 文件参考	40
6.20.1 详细描述	41
6.20.2 函数说明	41
6.20.2.1 linear_GRP_solver_Edir()	41
6.21 src/Riemann_solver/linear_GRP_solver_LAG.c 文件参考	42

---

6.21.1 详细描述	42
6.21.2 函数说明	42
6.21.2.1 linear_GRP_solver_LAG()	42
6.22 src/Riemann_solver/Riemann_solver_exact_Ben.c 文件参考	43
6.22.1 详细描述	43
6.22.2 函数说明	44
6.22.2.1 Riemann_solver_exact_Ben()	44
6.23 src/Riemann_solver/Riemann_solver_exact_Toro.c 文件参考	45
6.23.1 详细描述	45
6.23.2 函数说明	45
6.23.2.1 Riemann_solver_exact_Toro()	45
6.24 src/tools/math_algo.c 文件参考	46
6.24.1 详细描述	47
6.24.2 函数说明	47
6.24.2.1 rinv()	47
6.25 src/tools/str_num_common.c 文件参考	47
6.25.1 详细描述	48
6.25.2 函数说明	48
6.25.2.1 format_string()	48
6.25.2.2 str2num()	48
6.26 src/tools/sys_pro.c 文件参考	49
6.26.1 详细描述	49
6.26.2 函数说明	49
6.26.2.1 CreateDir()	49
6.26.2.2 DispPro()	50
<b>Index</b>	<b>51</b>

# Chapter 1

## 1D Godunov/GRP scheme for Lagrangian/Eulerian hydrodynamics

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

版本

0.1

### 1.1 File directory

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

### 1.2 Program structure

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

### 1.3 Program exit status code

<b>exit(0)</b>	EXIT_SUCCESS
<b>exit(1)</b>	File directory error

<b>exit(2)</b>	Data reading error
<b>exit(3)</b>	Calculation error
<b>exit(4)</b>	Arguments error
<b>exit(5)</b>	Memory error

## 1.4 Compile environment

- Linux/Unix: gcc, glibc, MATLAB/Octave
  - Compile in 'src/hydrocode': Run './make.sh' command on the terminal.
- Windows: 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

<b>Command Arguments</b>	name_of_test_example order coordinate (e.g. 'GRP_Book.6_1 1 EUL')
<b>Working Directory</b>	\$(SolutionDir)\$(Platform)\\$(Configuration)\

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

<b>Subsystem</b>	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:

<a href="#">cell_var</a>	Pointer structural body of variables on computational cells . . . . .	9
<a href="#">flu_var</a>	Pointer structural body of fluid variables . . . . .	10



# 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:

<a href="#">src/file_io/_1D_file.in.c</a>	This is a set of functions which control the read-in of one-dimensional data . . . . .	13
<a href="#">src/file_io/_1D_file.out.c</a>	This is a set of functions which control the readout of one-dimensional data . . . . .	15
<a href="#">src/file_io/config.in.c</a>	This is a set of functions which control the read-in of configuration data . . . . .	16
<a href="#">src/file_io/io_control.c</a>	This is a set of common functions which control the input/output data . . . . .	18
<a href="#">src/finite_volume/Godunov_solver_EUL.source.c</a>	This is an Eulerian Godunov scheme to solve 1-D Euler equations . . . . .	19
<a href="#">src/finite_volume/Godunov_solver_LAG.source.c</a>	This is a Lagrangian Godunov scheme to solve 1-D Euler equations . . . . .	20
<a href="#">src/finite_volume/GRP_solver_EUL.source.c</a>	This is an Eulerian GRP scheme to solve 1-D Euler equations . . . . .	21
<a href="#">src/finite_volume/GRP_solver_LAG.source.c</a>	This is a Lagrangian GRP scheme to solve 1-D Euler equations . . . . .	22
<a href="#">src/hydrocode/hydrocode.c</a>	This is a C file of the main function . . . . .	23
<a href="#">src/include/file_io.h</a>	This file is the header file that controls data input and output . . . . .	25
<a href="#">src/include/finite_volume.h</a>	This file is the header file of Lagrangian/Eulerian hydrocode in finite volume framework . . . . .	28
<a href="#">src/include/Riemann_solver.h</a>	This file is the header file of several Riemann solvers and GRP solvers . . . . .	31
<a href="#">src/include/tools.h</a>	This file is the header file of several independent tool functions . . . . .	36
<a href="#">src/include/var_struct.h</a>	. . . . .	39
<a href="#">src/Riemann_solver/linear_GRP_solver_Edir.c</a>	This is a direct Eulerian GRP solver for compressible inviscid flow in Li's paper . . . . .	40
<a href="#">src/Riemann_solver/linear_GRP_solver_LAG.c</a>	This is a Lagrangian GRP solver for compressible inviscid flow in Ben-Artzi's book . . . . .	42
<a href="#">src/Riemann_solver/Riemann_solver_exact_Ben.c</a>	This is an exact Riemann solver in Ben-Artzi's book . . . . .	43
<a href="#">src/Riemann_solver/Riemann_solver_exact_Toro.c</a>	This is an exact Riemann solver in Toro's book . . . . .	45

<a href="#">src/tools/math_algo.c</a>	
There are some mathematical algorithms . . . . .	46
<a href="#">src/tools/str_num_common.c</a>	
This is a set of common functions for string and number processing . . . . .	47
<a href="#">src/tools/sys_pro.c</a>	
There are some system processing programs . . . . .	49

## 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_struct.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\\_struct.h](#)

## 5.2 flu\_var结构体 参考

Pointer structural body of fluid variables.

```
#include <var_struct.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\\_struct.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_struct.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	<i>fp</i>	The pointer to the input file.
out	<i>U</i>	The pointer to the data array of fluid variables.
in	<i>num</i>	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	<i>name</i>	Name of the test example.
out	<i>FV0</i>	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_struct.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	<i>m</i>	The number of spatial points in the output data.
in	<i>N</i>	The number of time steps in the output data.
in	<i>CV</i>	Structural body of grid variable data.
in	<i>X[]</i>	Array of the coordinate data.
in	<i>cpu_time</i>	Array of the CPU time recording.
in	<i>name</i>	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_struct.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 `configure` (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

<code>in</code>	<code>fp</code>	The pointer to the configuration data file.
-----------------	-----------------	---

返回

Configuration data file read status.

返回值

<code>1</code>	Success to read in configuration data file.
<code>0</code>	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	<i>add</i> ↔ <i>_in</i>	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_struct.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	<i>example</i>	Name of the test example/numerical results.
out	<i>add_mkdir</i>	Folder path for data input or output.
in	<i>i_or_o</i>	Conversion parameters for data input/output. <ul style="list-style-type: none"> <li>• 0: data output.</li> <li>• else (e.g. 1): data input.</li> </ul>

#### 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	<i>fp</i>	The pointer to the input file.
in	<i>add</i>	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_struct.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	<i>m</i>	Number of the grids.
in, out	<i>CV</i>	Structural body of cell variable data.
in, out	<i>X[]</i>	Array of the coordinate data.
out	<i>cpu_time</i>	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_struct.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	<i>m</i>	Number of the grids.
in, out	<i>CV</i>	Structural body of cell variable data.
in, out	<i>X[]</i>	Array of the coordinate data.
out	<i>cpu_time</i>	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_struct.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	<i>m</i>	Number of the grids.
in, out	<i>CV</i>	Structural body of cell variable data.
in, out	<i>X[]</i>	Array of the coordinate data.
out	<i>cpu_time</i>	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_struct.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	<i>m</i>	Number of the grids.
in, out	<i>CV</i>	Structural body of cell variable data.
in, out	<i>X[]</i>	Array of the coordinate data.
out	<i>cpu_time</i>	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_struct.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	<i>argc</i>	ARGument counter.
in	<i>argv</i>	ARGument values. <ul style="list-style-type: none"> <li>• <i>argv</i>[1]: Folder name of test example (input path).</li> <li>• <i>argv</i>[2]: Folder name of numerical results (output path).</li> <li>• <i>argv</i>[3]: Dimensionality (= 1).</li> <li>• <i>argv</i>[4]: Order of numerical scheme[_scheme name] (= 1[_Riemann_exact] or 2[_GRP]).</li> <li>• <i>argv</i>[5]: Lagrangian/Eulerian coordinate framework (= LAG or EUL).</li> <li>• <i>argv</i>[6,7,...]: Configuration supplement <i>config</i>[n]=(double)C (= n=C).</li> </ul>

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 `configure` (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	<i>m</i>	The number of spatial points in the output data.
in	<i>N</i>	The number of time steps in the output data.
in	<i>CV</i>	Structural body of grid variable data.
in	<i>X[]</i>	Array of the coordinate data.
in	<i>cpu_time</i>	Array of the CPU time recording.
in	<i>name</i>	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	<i>name</i>	Name of the test example.
out	<i>FV0</i>	Structural body pointer of initial data array pointer.

### 6.10.2.3 `configure()`

```
void configure (
    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	<i>add</i> ↔ <i>_in</i>	Address 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	<i>example</i>	Name of the test example/numerical results.
out	<i>add_mkdir</i>	Folder path for data input or output.
in	<i>i_or_o</i>	Conversion parameters for data input/output. <ul style="list-style-type: none"> <li>• 0: data output.</li> <li>• else (e.g. 1): data input.</li> </ul>

#### 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	<i>fp</i>	The pointer to the input file.
in	<i>add</i>	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 * addmkdir, 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_struct.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	<i>m</i>	Number of the grids.
in, out	<i>CV</i>	Structural body of cell variable data.
in, out	<i>X[]</i>	Array of the coordinate data.
out	<i>cpu_time</i>	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	<i>m</i>	Number of the grids.
in, out	<i>CV</i>	Structural body of cell variable data.
in, out	<i>X[]</i>	Array of the coordinate data.
out	<i>cpu_time</i>	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	<i>m</i>	Number of the grids.
in, out	<i>CV</i>	Structural body of cell variable data.
in, out	<i>X[]</i>	Array of the coordinate data.
out	<i>cpu_time</i>	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	<i>m</i>	Number of the grids.
in, out	<i>CV</i>	Structural body of cell variable data.
in, out	<i>X[]</i>	Array of the coordinate data.
out	<i>cpu_time</i>	Array of the CPU time recording.

## 6.13 finite\_volume.h

[浏览该文件的文档.](#)

```
1
2 #ifndef FINITEVOLUME_H
3 #define FINITEVOLUME_H
4
5 #include "../include/var_struct.h"
6
7 void Godunov_solver_LAG_source
8 (const int m, struct cell_var CV, double * X[], double * cpu_time);
9
10 void GRP_solver_LAG_source
11 (const int m, struct cell_var CV, double * X[], double * cpu_time);
12
13 void Godunov_solver_EUL_source
14 (const int m, struct cell_var CV, double * X[], double * cpu_time);
15
16 void GRP_solver_EUL_source
17 (const int m, struct cell_var CV, double * X[], double * cpu_time);
18
19 #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  $\gamma$ -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	<i>direvative</i>	the temporal derivative of fluid variables. [rho, u, p].t
out	<i>mid</i>	the Riemann solutions. [rho_star, u_star, p_star]
in	<i>rho_L,u_L,p_L</i>	Left States.
in	<i>rho_R,u_R,p_R</i>	Right States.
in	<i>s_rho_L,s_u_L,s_p_L</i>	Left x-spatial derivatives.
in	<i>s_rho_R,s_u_R,s_p_R</i>	Right x-spatial derivatives.
in	<i>gamma</i>	the constant of the perfect gas.
in	<i>eps</i>	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	<i>dire</i>	the temporal derivative of fluid variables. [rho_L, u, p, rho_R]_t
out	<i>mid</i>	the Riemann solutions. [rho_star_L, u_star, p_star, rho_star_R]
in	<i>rho_L,u_L,p_L</i>	Left States.
in	<i>rho_R,u_R,p_R</i>	Right States.
in	<i>s_rho_L,s_u_L,s_p_L</i>	Left $\xi$ -Lagrangian spatial derivatives.
in	<i>s_rho_R,s_u_R,s_p_R</i>	Right $\xi$ -Lagrangian spatial derivatives.
in	<i>gamma</i>	the constant of the perfect gas.
in	<i>eps</i>	the largest value could be seen as zero.
in	<i>atc</i>	Parameter that determines the solver type. <ul style="list-style-type: none"> <li>• INFINITY: acoustic approximation</li> <li>• eps: GRP solver(nonlinear + acoustic case)</li> <li>• -0.0: GRP solver(only nonlinear case)</li> </ul>

#### 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 $\gamma$ -Law Gas

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

#### 参数

out	<i>U_star, P_star</i>	Velocity/Pressure in star region.
in	<i>u_L, p_L, c_L</i>	Initial Velocity/Pressure/sound speed on left state.
in	<i>u_R, p_R, c_R</i>	Initial Velocity/Pressure/sound speed on right state.
in	<i>gamma</i>	Ratio of specific heats.
out	<i>CRW</i>	Centred Rarefaction Wave (CRW) Indicator of left and right waves. <ul style="list-style-type: none"> <li>• true: CRW</li> <li>• false: Shock wave</li> </ul>
in	<i>eps</i>	The largest value can be seen as zero.
in	<i>tol</i>	Condition value of 'gap' at the end of the iteration.
in	<i>N</i>	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	<i>U_star, P_star</i>	Velocity/Pressure in star region.
in	<i>U_l, P_l, c_l</i>	Initial Velocity/Pressure/sound_speed on left state.
in	<i>U_r, P_r, c_r</i>	Initial Velocity/Pressure/sound_speed on right state.
in	<i>gamma</i>	Ratio of specific heats.
out	<i>CRW</i>	Centred Rarefaction Wave (CRW) Indicator of left and right waves. <ul style="list-style-type: none"> <li>• true: CRW</li> <li>• false: Shock wave</li> </ul>
in	<i>eps</i>	The largest value can be seen as zero.
in	<i>tol</i>	Condition value of 'gap' at the end of the iteration.
in	<i>N</i>	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

### 版权所有

This program is part of NUMERICA —

A Library of Source Codes for Teaching, Research and Applications, by E. F. Toro

Published by NUMERITEK LTD

## 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	<i>pPath</i>	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	<i>pro</i>	Numerator of percent that the process has completed.
in	<i>step</i>	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\_struct.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\_struct.h

[浏览该文件的文档.](#)

```
1 #ifndef VARSTRUC_H
2 #define VARSTRUC_H
3
4 #ifndef EPS
5 #define EPS 1e-9
6 #endif
7
8 #ifndef N_CONF
9 #define N_CONF 400
10 #endif
11
12 extern double config[];
13
14 struct flu_var {
15     double *RHO, *U, *P;
16 };
17
18 struct cell_var {
19     double **RHO, **U, **V, **P, **E;
20 };
21
22 #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	<i>direvative</i>	the temporal derivative of fluid variables. [rho, u, p].t
out	<i>mid</i>	the Riemann solutions. [rho_star, u_star, p_star]
in	<i>rho_L, u_L, p_L</i>	Left States.
in	<i>rho_R, u_R, p_R</i>	Right States.
in	<i>s_rho_L, s_u_L, s_p_L</i>	Left x-spatial derivatives.
in	<i>s_rho_R, s_u_R, s_p_R</i>	Right x-spatial derivatives.
in	<i>gamma</i>	the constant of the perfect gas.
in	<i>eps</i>	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	<i>dire</i>	the temporal derivative of fluid variables. [rho_L, u, p, rho_R]_t
out	<i>mid</i>	the Riemann solutions. [rho_star_L, u_star, p_star, rho_star_R]
in	<i>rho_L,u_L,p_L</i>	Left States.
in	<i>rho_R,u_R,p_R</i>	Right States.
in	<i>s_rho_L,s_u_L,s_p_L</i>	Left $\xi$ -Lagrangian spatial derivatives.
in	<i>s_rho_R,s_u_R,s_p_R</i>	Right $\xi$ -Lagrangian spatial derivatives.
in	<i>gamma</i>	the constant of the perfect gas.
in	<i>eps</i>	the largest value could be seen as zero.
in	<i>atc</i>	Parameter that determines the solver type. <ul style="list-style-type: none"> <li>• INFINITY: acoustic approximation</li> <li>• eps: GRP solver(nonlinear + acoustic case)</li> <li>• -0.0: GRP solver(only nonlinear case)</li> </ul>

## 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  $\gamma$ -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 $\gamma$ -Law Gas

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

#### 参数

out	<i>U_star, P_star</i>	Velocity/Pressure in star region.
in	<i>u_L, p_L, c_L</i>	Initial Velocity/Pressure/sound speed on left state.
in	<i>u_R, p_R, c_R</i>	Initial Velocity/Pressure/sound speed on right state.
in	<i>gamma</i>	Ratio of specific heats.
out	<i>CRW</i>	Centred Rarefaction Wave (CRW) Indicator of left and right waves. <ul style="list-style-type: none"> <li>• true: CRW</li> <li>• false: Shock wave</li> </ul>
in	<i>eps</i>	The largest value can be seen as zero.
in	<i>tol</i>	Condition value of 'gap' at the end of the iteration.
in	<i>N</i>	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	<i>U_star, P_star</i>	Velocity/Pressure in star region.
in	<i>U_l, P_l, c_l</i>	Initial Velocity/Pressure/sound_speed on left state.

## 参数

in	<i>U_r, P_r, c_r</i>	Initial Velocity/Pressure/sound speed on right state.
in	<i>gamma</i>	Ratio of specific heats.
out	<i>CRW</i>	Centred Rarefaction Wave (CRW) Indicator of left and right waves. <ul style="list-style-type: none"> <li>• true: CRW</li> <li>• false: Shock wave</li> </ul>
in	<i>eps</i>	The largest value can be seen as zero.
in	<i>tol</i>	Condition value of 'gap' at the end of the iteration.
in	<i>N</i>	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

## 版权所有

This program is part of NUMERICA —  
A Library of Source Codes for Teaching, Research and Applications, by E. F. Toro  
Published by NUMERITEK LTD

## 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	<i>a</i>	The pointer of the input/output square matrix.
in	<i>n</i>	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	<i>str</i>	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	<i>number</i>	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	<i>pPath</i>	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	<i>pro</i>	Numerator of percent that the process has completed.
in	<i>step</i>	Number of time steps.



# Index

- [\\_1D\\_file.in.c](#)
    - [\\_1D\\_flu\\_var\\_read](#), [14](#)
    - [\\_1D\\_initialize](#), [14](#)
    - [STR\\_FLU\\_INI](#), [13](#)
  - [\\_1D\\_file.out.c](#)
    - [\\_1D\\_file\\_write](#), [16](#)
    - [PRINT\\_NC](#), [15](#)
  - [\\_1D\\_file\\_write](#)
    - [\\_1D\\_file\\_out.c](#), [16](#)
    - [file\\_io.h](#), [25](#)
  - [\\_1D\\_flu\\_var\\_read](#)
    - [\\_1D\\_file.in.c](#), [14](#)
  - [\\_1D\\_initialize](#)
    - [\\_1D\\_file.in.c](#), [14](#)
    - [file\\_io.h](#), [26](#)
- [cell\\_var](#), [9](#)
  - [E](#), [9](#)
  - [P](#), [9](#)
  - [RHO](#), [10](#)
  - [U](#), [10](#)
  - [V](#), [10](#)
- [config](#)
  - [hydrocode.c](#), [25](#)
  - [var\\_struct.h](#), [40](#)
- [config\\_check](#)
  - [config\\_in.c](#), [17](#)
- [config\\_in.c](#)
  - [config\\_check](#), [17](#)
  - [config\\_read](#), [17](#)
  - [configure](#), [17](#)
- [config\\_read](#)
  - [config\\_in.c](#), [17](#)
- [configure](#)
  - [config\\_in.c](#), [17](#)
  - [file\\_io.h](#), [26](#)
- [CreateDir](#)
  - [sys\\_pro.c](#), [49](#)
  - [tools.h](#), [37](#)
- [CV\\_INIT\\_MEM](#)
  - [hydrocode.c](#), [24](#)
- [DispPro](#)
  - [sys\\_pro.c](#), [50](#)
  - [tools.h](#), [37](#)
- [E](#)
  - [cell\\_var](#), [9](#)
- [EPS](#)
  - [var\\_struct.h](#), [39](#)
- [example\\_io](#)
  - [file\\_io.h](#), [27](#)
  - [io\\_control.c](#), [18](#)
- [file\\_io.h](#)
  - [\\_1D\\_file\\_write](#), [25](#)
  - [\\_1D\\_initialize](#), [26](#)
  - [configure](#), [26](#)
  - [example\\_io](#), [27](#)
  - [flu\\_var\\_count](#), [27](#)
- [finite\\_volume.h](#)
  - [Godunov\\_solver\\_EUL\\_source](#), [29](#)
  - [Godunov\\_solver\\_LAG\\_source](#), [29](#)
  - [GRP\\_solver\\_EUL\\_source](#), [29](#)
  - [GRP\\_solver\\_LAG\\_source](#), [30](#)
- [flu\\_var](#), [10](#)
  - [P](#), [10](#)
  - [RHO](#), [11](#)
  - [U](#), [11](#)
- [flu\\_var\\_count](#)
  - [file\\_io.h](#), [27](#)
  - [io\\_control.c](#), [19](#)
- [format\\_string](#)
  - [str\\_num\\_common.c](#), [48](#)
- [Godunov\\_solver\\_EUL\\_source](#)
  - [finite\\_volume.h](#), [29](#)
  - [Godunov\\_solver\\_EUL\\_source.c](#), [20](#)
- [Godunov\\_solver\\_EUL\\_source.c](#)
  - [Godunov\\_solver\\_EUL\\_source](#), [20](#)
- [Godunov\\_solver\\_LAG\\_source](#)
  - [finite\\_volume.h](#), [29](#)
  - [Godunov\\_solver\\_LAG\\_source.c](#), [21](#)
- [Godunov\\_solver\\_LAG\\_source.c](#)
  - [Godunov\\_solver\\_LAG\\_source](#), [21](#)
- [GRP\\_solver\\_EUL\\_source](#)
  - [finite\\_volume.h](#), [29](#)
  - [GRP\\_solver\\_EUL\\_source.c](#), [22](#)
- [GRP\\_solver\\_EUL\\_source.c](#)
  - [GRP\\_solver\\_EUL\\_source](#), [22](#)
- [GRP\\_solver\\_LAG\\_source](#)
  - [finite\\_volume.h](#), [30](#)
  - [GRP\\_solver\\_LAG\\_source.c](#), [22](#)
- [GRP\\_solver\\_LAG\\_source.c](#)
  - [GRP\\_solver\\_LAG\\_source](#), [22](#)
- [hydrocode.c](#)
  - [config](#), [25](#)
  - [CV\\_INIT\\_MEM](#), [24](#)
  - [main](#), [24](#)

- io\_control.c
  - example\_io, 18
  - flu\_var\_count, 19
- linear\_GRP\_solver\_Edir
  - linear\_GRP\_solver\_Edir.c, 41
  - Riemann\_solver.h, 31
- linear\_GRP\_solver\_Edir.c
  - linear\_GRP\_solver\_Edir, 41
- linear\_GRP\_solver\_LAG
  - linear\_GRP\_solver\_LAG.c, 42
  - Riemann\_solver.h, 32
- linear\_GRP\_solver\_LAG.c
  - linear\_GRP\_solver\_LAG, 42
- main
  - hydrocode.c, 24
- math\_algo.c
  - rinv, 47
- minmod2
  - tools.h, 37
- minmod3
  - tools.h, 38
- N\_CONF
  - var\_struct.h, 39
- P
  - cell\_var, 9
  - flu\_var, 10
- PRINT\_NC
  - \_1D\_file\_out.c, 15
- RHO
  - cell\_var, 10
  - flu\_var, 11
- Riemann\_solver.h
  - linear\_GRP\_solver\_Edir, 31
  - linear\_GRP\_solver\_LAG, 32
  - Riemann\_solver\_exact, 31
  - Riemann\_solver\_exact\_Ben, 33
  - Riemann\_solver\_exact\_Toro, 34
- Riemann\_solver\_exact
  - Riemann\_solver.h, 31
- Riemann\_solver\_exact\_Ben
  - Riemann\_solver.h, 33
  - Riemann\_solver\_exact\_Ben.c, 44
- Riemann\_solver\_exact\_Ben.c
  - Riemann\_solver\_exact\_Ben, 44
- Riemann\_solver\_exact\_Toro
  - Riemann\_solver.h, 34
  - Riemann\_solver\_exact\_Toro.c, 45
- Riemann\_solver\_exact\_Toro.c
  - Riemann\_solver\_exact\_Toro, 45
- rinv
  - math\_algo.c, 47
  - tools.h, 38
- src/file\_io/\_1D\_file\_in.c, 13
- src/file\_io/\_1D\_file\_out.c, 15
- src/file\_io/config\_in.c, 16
- src/file\_io/io\_control.c, 18
- src/finite\_volume/Godunov\_solver\_EUL\_source.c, 19
- src/finite\_volume/Godunov\_solver\_LAG\_source.c, 20
- src/finite\_volume/GRP\_solver\_EUL\_source.c, 21
- src/finite\_volume/GRP\_solver\_LAG\_source.c, 22
- src/hydrocode/hydrocode.c, 23
- src/include/file\_io.h, 25, 28
- src/include/finite\_volume.h, 28, 30
- src/include/Riemann\_solver.h, 31, 36
- src/include/tools.h, 36, 38
- src/include/var\_struct.h, 39, 40
- src/Riemann\_solver/linear\_GRP\_solver\_Edir.c, 40
- src/Riemann\_solver/linear\_GRP\_solver\_LAG.c, 42
- src/Riemann\_solver/Riemann\_solver\_exact\_Ben.c, 43
- src/Riemann\_solver/Riemann\_solver\_exact\_Toro.c, 45
- src/tools/math\_algo.c, 46
- src/tools/str\_num\_common.c, 47
- src/tools/sys\_pro.c, 49
- str2num
  - str\_num\_common.c, 48
- STR\_FLU\_INI
  - \_1D\_file\_in.c, 13
- str\_num\_common.c
  - format\_string, 48
  - str2num, 48
- sys\_pro.c
  - CreateDir, 49
  - DispPro, 50
- tools.h
  - CreateDir, 37
  - DispPro, 37
  - minmod2, 37
  - minmod3, 38
  - rinv, 38
- U
  - cell\_var, 10
  - flu\_var, 11
- V
  - cell\_var, 10
- var\_struct.h
  - config, 40
  - EPS, 39
  - N\_CONF, 39