

# conti2d

## Description

Conti2d is a free source and cross-platform C++ program to calculate upward and downward continuation of potential field data. The algorithm is implemented in spatial domain. The [source](#) file, precompiled [Windows](#) and [macOS](#) version are available on [Github](#). The doxygen document of source files can be accessed on [github-pages](#).

### Conti2D 2.1

Main Page

Data Types List ▾

Files ▾

src

Conti2D.cpp File Reference

#include "Conti2D.h"  
Include dependency graph for Conti2D.cpp:

```
graph TD
    C2Dcpp[Conti2D.cpp] --> C2Dh[Conti2D.h]
    C2Dh --> sstream
    C2Dh --> fstream
    C2Dh --> iomanip
    C2Dh --> FFTN_h[FTTN.h]
    C2Dh --> string_h[string.h]
    C2Dh --> omp_h[omp.h]
    C2Dh --> MultiProgressBar_h[MultiProgressBar.h]
    C2Dh --> netcdf_h[netcdf.h]
    C2Dh --> unistd_h[unistd.h]
    C2Dh --> math_h[math.h]
    C2Dh --> fftw3_h[fftw3.h]
    C2Dh --> vector
    C2Dh --> iostream
    C2Dh --> stdio_h[stdio.h]
    C2Dh --> sys_roctf_h[sys/roctf.h]
    C2Dcpp --> C2Dcpp
```

Go to the source code of this file.

Functions

string Path\_GetFileName (string filepath)  
Get file name from a full path. More...

string Path\_GetPath (string filepath)  
Get file path from a full path. More...

string Path\_GetExtName (string filepath)  
Get file extension name from a full file path. More...

string Path\_GetBaseName (string filepath)  
Get file name from a full path, e.g. More...

void GetKernelMatrix (GrdHead grdhead, double \*G, const double rph, int num\_thread)  
Get the Kernel Matrix object. More...

void GetKernelMatrix\_new (GrdHead grdhead, double \*G, const double rph, int num\_thread)  
Get the kernel matrix using the new developed formula. More...

# Download

The [windows version](#) and [macOS version](#) can be download and executed directly.

The executable file has been tested successfully on several windows computers.

# Build

If you can't run conti2d successfully. Then you can try to build it from source file.

- For the windows system, the visual studio 2017 project file and source file are available on [Github](#). Just download and open the vs2017 project file windows/conti2d/conti2d.sln in the main folder. Then rebuild the project and you will get the executable file in the folder of windows/conti2d/x64/release.
- For macOS and Linux users, you can clone or download the [source file](#) and change directory to the main folder for the source file in the terminal. Then you just type command of **make**, the executable file will be generated in the bin folder. Just like the following steps in snapshot.

```

→ conti2d git:(master) x pwd
/MyData/Research/3-CodeProject/conti2d
→ conti2d git:(master) x ls
Doxyfile  bin      docs      example  fftw-3.3.7  figures  makefile  netcdf     package.json  readme.md  readme.pdf  src
→ conti2d git:(master) x make
g++-9 -fopenmp -c src/MultiProgressBar.cpp -o src/MultiProgressBar.o -I./fftw-3.3.7/build/include -I./netcdf/include
g++-9 -fopenmp -c src/Conti2D.cpp -o src/Conti2D.o -I./fftw-3.3.7/build/include -I./netcdf/include
g++-9 -fopenmp -c src/FFTN.cpp -o src/FFTN.o -I./fftw-3.3.7/build/include -I./netcdf/include
g++-9 -fopenmp -c src/main.cpp -o src/main.o -I./fftw-3.3.7/build/include -I./netcdf/include
g++-9 -fopenmp -o bin/conti2d src/MultiProgressBar.o src/Conti2D.o src/FFTN.o src/main.o -I./fftw-3.3.7/build/include -I./netcdf/include -L/MyData/Research/3-CodeProject/conti2d/fftw-3.3.7/build/lib -lfftw3 -L./netcdf/lib -lnetcdf
→ conti2d git:(master) x ls bin
conti2d
→ conti2d git:(master) x conti2d

===== conti2d =====
Analytical continuation of potential field data
Author      Zhikui Guo
Locus       SIO, Hangzhou
Date        Aug 8, 2016
Version     2.0
/ y
/ 0 _____ x
|
|
| z(-)
Synopsis:   conti2d input.grd -Goutput.grd -EextBoundNum -Hlevel_outputdata|grdfile -
ads -f[FrequencyDomain]
=====

```

Figure 1. Steps of building conti2d from source on macOS and Linux.

## Synopsis

**conti2d** Inputfile -Goutputfile -EextBoundNum -Hlevel\_outputdata|grdfile -  
Tlevel\_inputdata|grdfile -D[+Llandweber\_par] -tthreads -f

**Note:** No space is allowed between the option flag and the associated arguments.

## Required Arguments

### 1. Upward continuation from plane to plane

*Inputfile*

Names of data file to be continued. The supported input file format is ASCII Golden Software Surfer format (see example data file in the attachement)

### **-G**outputfile

Output file name. The supported file format are ASCII Golden Software Surfer format, netCDF, vtk, which are identified by the extension name .grd, .nc and .vtk respectively. For example, -Gout.vtk means the output grid data file in vtk format, which can be viewed by [Paraview](#). The .nc file can also be view by Paraview. The .grd and .nc file can be plotted by [Golden Software Surfer](#) and [GMT](#) command of [grdimage](#).

### **-H**height

A value represents height of upward continuation.

**Example:** upward continuation from plane of  $z=0$  to plane of  $z=8$

**conti2d** mag\_plane\_0.grd **-G**mag\_plane\_uwc\_8.vtk **-H8**

## 2. Upward continuation from plane to surface

### *Inputfile*

Names of data file to be continued. The supported input file format is ASCII Golden Software Surfer format (see example data file in the attachement)

### **-G**outputfile

Output file name. The supported file format are ASCII Golden Software Surfer format, netCDF, vtk, which are identified by the extension name .grd, .nc and .vtk respectively. For example, -Gout.vtk means the output grid data file in vtk format, which can be viewed by [Paraview](#). The .nc file can also be view by Paraview. The .grd and .nc file can be plotted by [Golden Software Surfer](#) and [GMT](#) command of [grdimage](#).

**-T***level\_inputdata*

A value represents z level of the input grid data.

**-H***topography\_outputdata*

A grid data file represents the upper surface (topography of the output field data).

**Example:** upward continuation from plane of  $z=0$  to a surface of topo.grd

**conti2d** mag\_plane\_0.grd **-G**mag\_plane\_uwc\_p2s.vtk **-T0 -H**topo.grd

### 3. Downward continuation from plane to plane

*Inputfile*

Names of data file to be continued. The supported input file format is ASCII Golden Software Surfer format (see example data file in the attachement)

**-G***outputfile*

Output file name. The supported file format are ASCII Golden Software

Surfer format, netCDF, vtk, which are identified by the extension name .grd, .nc and .vtk respectively. For example, -Gout.vtk means the output grid data file in vtk format, which can be viewed by [Paraview](#). The .nc file can also be view by Paraview. The .grd and .nc file can be plotted by [Golden Software Surfer](#) and [GMT](#) command of [grdimage](#).

#### **-Hzlevel\_output**

A value represents z level of the input field data.

#### **-Tzlevel\_input**

A value represents z level of the continued field data.

#### **-D+Literation\_number**

D means downward continuation. L means Landweber method.

Iteration\_number is the maximum iteration number.

### **Optional Arguments**

#### **-tthreads**

A integer (>1) represents how much threads do you want to use. The default value is the maximum threads of your computer.

#### **-Eextension\_number**

A integer value(>0) represents how much points do you want to extend the boundary. The default value is zero.

### **Temporary results of iteration**

The results of each iteration are saved in an automatically created folder, which name is like *inputfile\_dwc\_p2p\_Landwebber*. The file name of the temporary results are formatted as **result\_n.vtk**, which are like a time-series data sets and can be played by Paraview to see the iteration and convergence process. And the *fitting-smooth* curve data is also save in this folder, the data file contains four columns, they are **iteration number**, **fitting**, **2<sup>th</sup> norm of result**, **smooth**, respectively. Then plot the fitting-smooth curve to find the optimal iteration number and the optimal result.

**Example:** downward continuation from plane of z=8 m to a plane of z=0 m

**conti2d** mag\_plane\_8.grd -Gmag\_plane\_dwc\_p2p.vtk -T8 -H0 -E5 -D+L500

#### 4. Downward continuation from surface to plane

##### *Inputfile*

Names of data file to be continued. The supported input file format is ASCII Golden Software Surfer format (see example data file in the attachement)

##### **-G***outputfile*

Output file name. The supported file format are ASCII Golden Software Surfer format, netCDF, vtk, which are identified by the extension name .grd, .nc and .vtk respectively. For example, -Gout.vtk means the output grid data file in vtk format, which can be viewed by [Paraview](#). The .nc

file can also be view by Paraview. The .grd and .nc file can be plotted by [Golden Software Surfer](#) and [GMT](#) command of [grdimage](#).

#### **-Hzlevel\_output**

A value represents z level of the input field data.

#### **-Tzlevel\_input**

A value represents z level of the continued field data.

#### **-D+Literation\_number**

D means downward continuation. L means Landweber method.

Iteration\_number is the maximum iteration number.

### **Optional Arguments**

#### **-tthreads**

A integer (>1) represents how much threads do you want to use. The default value is the maximum threads of your computer.

#### **-Extension\_number**

A integer value(>0) represents how much points do you want to extend the boundary. The default value is zero.

### **Temporary results of iteration**

The results of each iteration are saved in an automatically created folder, which name is like *inputfile\_dwc\_s2p\_Landwebber*. The file name of the temporary results are formatted as **result\_n.vtk**, which are like a time-series



data sets and can be played by Paraview to see the iteration and convergence process. And the *fitting-smooth* curve data is also save in this folder, the data file contains four columns, they are **iteration number**, **fitting**, **2<sup>th</sup> norm of result**, **smooth**, respectively. Then plot the fitting-smooth curve to find the optimal iteration number and the optimal result.

**Example:** downward continuation from surface of topo.grd to a plane of z=0 m  
**conti2d** mag\_plane\_8.grd -Gmag\_plane\_dwc\_p2p.vtk -Ttopo.grd -H0 -E5 -  
D+L500 -t8