Implementation of Higher Order Panel Methods Deployment Guide

Revision History

Author	Revision Number	Date
yedtoss	1.0	05/01/2015

Deployment Instructions

1. Organization of Submission

Nemoh/ Contains the modified source of Nemoh Fortran software
docs/ Contains this deployment guide
docs/GreenFunctionDerivativeODE.pdf Contains ODE for the derivative of the green function
NemohPython/ Contains the modified Nemoh python code
README.txt note about testing previous old Fortran code

2. Application Setup

Linux

- GCC with GFortran >= 4.8

- BLAS
- LAPACK
- OpenMP provided by GCC
- HDF5 >=1.8.11 http://www.hdfgroup.org/HDF5/ Optionally provided by Anaconda
- HDFView http://www.hdfgroup.org/products/java/release/download.html
- Python 2.7 Optionally provided by Anaconda
- H5py >= 2.3.1 Optionally provided by Anaconda
- Numpy Optionally provided by Anaconda
- Cmake >= 4.8 http://www.cmake.org/cmake/resources/software.html
- Anaconda (with Python 2.7) >= 2.1.0 http://continuum.io/downloads

Windows

- MinGW 4.8.1 http://sourceforge.net/projects/MinGWbuilds/files/host-windows/releases/4.8.1/
- BLAS http://icl.cs.utk.edu/lapack-for-windows/lapack/#libraries
- LAPACK http://icl.cs.utk.edu/lapack-for-windows/lapack/#libraries
- OpenMP provided by MinGW
- HDF5 >=1.8.11 http://www.hdfgroup.org/HDF5/ Optionally provided by Anaconda
- HDFView http://www.hdfgroup.org/products/java/release/download.html
- Python 2.7 Optionally provided by Anaconda
- H5py >= 2.3.1 Optionally provided by Anaconda
- Numpy Optionally provided by Anaconda
- Cmake >= 4.8 http://www.cmake.org/cmake/resources/software.html
- Anaconda (with Python 2.7) >= 2.1.0 http://continuum.io/downloads

MAC OS X

- Brew http://brew.sh/
- GCC >= 4.8 installed by brew
- XCode Command Line Tools installed by brew
- BLAS provided by Xcode commands

- LAPACK Provided by Xcode commands
- OpenMP provided by GCC
- HDF5 >=1.8.11 http://www.hdfgroup.org/HDF5/ Optionally provided by Anaconda
- HDFView http://www.hdfgroup.org/products/java/release/download.html
- Python 2.7 Optionally provided by Anaconda
- H5py >= 2.3.1 Optionally provided by Anaconda
- Numpy Optionally provided by Anaconda
- Cmake >= 4.8 installed by brew
- Anaconda (with Python 2.7) >= 2.1.0 http://continuum.io/downloads

Variables

Let's call:
$NEMOH_FORTRAN the directory Nemoh/ in the root of the submission directory
$NEMOH_PYTHON the directory NemohPython/
$FORTRAN_BUILD the build directory for the FORTRAN version of Nemoh
$MINGW_ROOT the directory where MINGW will be installed

3. Configuration

The python code can be configured using the file **$NEMOH_PYTHON/nemoh/settings.py**
Newly added properties are in red

Property	Definition	Example
USE_HIGHER_ORDER	Whether or not to use higher order panel method.	True
NUM_PANEL_HIGHER_ORDER	The number of panel per patch in the higher order method. Read section 4 and 7 before attempting to increase this	1
B_SPLINE_ORDER	The order of the B-Spline for the potential in the higher order panel method. Read section 4 and 7 before attempting to increase this	1
USE_ODE_INFLUENCE_COEFFICIENTS	Indicate whether or not to use the ODE method to compute the influence coefficients	True
GREEN_TABULATION_NUMX	Number of points in x direction of tabulated data	500
GREEN_TABULATION_NUMZ	Number of points in z direction of tabulated data	60
GREEN_TABULATION_SIMPSON_NPOINTS	Number of sub intervals used to approximate the green function	551

Property	Definition	Example
	integral using simpson rule	
HDF5_FILE	The path to the hdf5 file where to save and load the results and input. Required	'db.hdf5' No need to change
NEMOH_CALCULATIONS_FILE	The old nemoh calculation file. Not required but it is needed to automatically convert the old Nemoh.cal file to hdf5 storage	'Nemoh.cal' No need to change but then make sure you have the nemoh calculation file Nemoh.cal in your working directory. Also make sure the path to the mesh file references in Nemoh.cal exists. For exemple if in the Nemoh.cal file you have 'Cylinder.dat', then you should have the Cylinder.dat file in your current directory
NEMOH_INPUT_FILE	Same as above but applied to the nemoh input file	'input.txt' No need to change but then make sure you have the nemoh input file input.txt in your working directory
NEMOH_INT	Represents the integer type to use when performing computations. It should be a valid numpy integer type. See http://docs.scipy.org/doc/numpy/reference/arrays.scalars.html#arrays-scalars-built-in for possible values	'i'
NEMOH_FLOAT	Represents the float type to use when performing computations. It should be a valid numpy float type. See http://docs.scipy.org/doc/numpy/reference/arrays.scalars.html#arrays-scalars-built-in for possible values	'f'
NEMOH_COMPLEX	Represents the complex type to use when performing computations. It should be a valid numpy complex type. See http://docs.scipy.org/doc/numpy/reference/arrays.scalars.html#arra	'F'

Property	Definition	Example
	ys-scalars-built-in for possible values	
MESH_TEC_FILE	The path to the file where to save the mesh tec file	No need to change. The default value is fine.
FK_FORCE_TEC_FILE	The path to the froudkrylov force data in tec format	No need to change. The default value is fine.
RADIATION_COEFFICIENTS_TEC_FILE	The path to the file where to save the added mass and damping forces for the radiation problems in tec format.	No need to change. The default value is fine.
DIFFRACTION_FORCE_TEC_FILE	The path to the file where to save the diffraction force for the diffraction problems in tec format.	No need to change. The default value is fine.
EXCITATION_FORCE_TEC_FILE	The path to the file where to save the the excitation force for the diffraction problems in tec format.	No need to change. The default value is fine.
IRF_TEC_FILE	The path to the file where to save the IRF tec file	No need to change. The default value is fine.
WAVE_FIELD_TEC_FILE	The path to the file where to save the wave field tec file	No need to change. The default value is fine.

4. Implementation details

The references are listed in section 9.

The main code for higher panel method is implemented in
"$NEMOH_FORTRAN/Solver/Core/SOLVE_BEM_HIGH.f90"

The body is assumed to be divided in patches. We will then transform the patches from a Cartesian coordinate (x, y, z) to a rectangular shape in a parametric space (u, v). The precise transformation from Cartesian to parametric coordinate is done according to equation 2.49 and 2.46 of reference R4.
The parametric space is such that it's domain is [-1, 1]x[-1, 1]

The higher panel method is performed by modeling the velocity potential as a product of B-Spline basis

functions U(u) and V(v) as shown in equation (7.4) of reference R2. The total number of B-spline basis

functions on each patch is Mu times Mv. A panel is assumed to be a non-zero space between the element of the knots vectors of the B-Spline U and V. This would lead to Nu and Nv panel respectively for each B-Spline U and V. The order of the B-Spline is Ku and Kv respectively. For a brief description of the method, see section 15.5 of reference R2. For a more in-depth review see reference R1.

The Garlekin method is used to discretize the radiation equation. This leads to equation (15.30) of reference R2. This equation is a linear equation. The number of unknowns is the sum of the product MuXMv over each patches or (Nu+Ku-1)*(Nv+Kv+1) over each patches. The system is solved using a

Singular Value Decomposition (SVD) method.

To compute the green function and derivative we use previous approach in Nemoh through series approximation and interpolation.

Note that the implementation is not dependent on the method used to describe the geometry. As far as a transformation from Cartesian coordinate in the body to parametric space is provided, together with it's Jacobian any geometry description can be used.

Once the linear system of equation is solved, the velocity potential is determined using equation (15.29) of reference R2.

The user is expected to provide the number of patches(P), the number of panel(N) per patches and the order of the B-Spline (K). For an optimal solution the number of patches should be low (10 at a maximum). The order of the B-Spline should be 3 or 4 usually and the number of panels can be increased for better accuracy. Keep in mind that a system of CxC with C=P*(N+K-1)*(N+K-1) will be solved by the implementation and memory needed is in the order of CxCxC.
It is important that C is expected to be much lower than the order of the linear equations in the lower order panel method. Even with a lower C, the higher panel method will give a better accuracy if the description of the body is complex and not a simple flat quadrilateral (See reference R1).

5. Deployment Instructions

5.1. Generic Instructions

5.1.1. Install a Fortran and C/C++ compiler which support OpenMP (Currently gcc and ifort are supported)
5.1.2. Install Python 2.7
5.1.3. Install pip
5.1.4. Install CMake version greater or equal to 2.8
5.1.5. Install Blas and Lapack and make sure there are in the library search paths
5.1.6. Install hdf5 libraries version greater or equal to 1.8.11 and make sure there are in the library search path

5.1.7. Compile the Fortran version of Nemoh Solver:
5.1.7.1. Create a build directory $FORTRAN_BUILD different from $NEMOH_FORTRAN
5.1.7.2. Go to $FORTRAN_BUILD folder.
5.1.7.3. Delete the file CMakeCache.txt and the directory CMakeFiles if it exists
5.1.7.4. Run the command cmake
5.1.7.5. Run make. The Nemoh library will be created

5.1.8. Compile the Nemoh python against the Nemoh Fortran
5.1.8.1. Go to $NEMOH_PYTHON
5.1.8.2. Install the python module prerequisites pip install -r requirements.txt

5.1.8.3. Make sure the $FORTRAN_BUILD is in the library search paths for compilation and for linking
5.1.8.4. Run python setup.py build_ext --inplace

5.2. Linux Instructions with Ubuntu commands

(If you are not using Ubuntu, you should be able to use your distribution package manager to install equivalent commands)

5.2.1. Install GFortran and gcc by running: sudo apt-get install build-essential gfortran gcc

5.2.2. Install cmake by running sudo apt-get install cmake

5.2.3. Install Blas and Lapack and make sure there are in the library search paths by running sudo apt-get install liblapack-dev libblas-dev

5.2.4. Install python 2.7, hdf5 libraries, and the nemoh python module requirements. To do so, we just need to install Anaconda:
5.2.4.1. Download Anaconda >= 2.1.0 from http://continuum.io/downloads .
For linux 64 bits the direct link is (with no space) http://09c8d0b2229f813c1b93-c95ac804525aac4b6dba79b00b39d1d3.r79.cf1.rackcdn.com/Anaconda-2.1.0-Linux-x86_64.sh
5.2.4.2. Install Anaconda by running bash Anaconda-2.1.0-Linux-x86_64.sh
5.2.4.3. When prompted, accept to add it's path to your ~/.bashrc.
5.2.4.4. Make sure the python version you are using is the one from Anaconda by logout then login or by running source ~/.bashrc
5.2.4.5. If successful, when you run python --version you should see something like **Python 2.7.8 :: Anaconda 2.1.0 (64-bit)**

5.2.5. Compile the Nemoh Fortran
5.2.5.1. Create a build directory $FORTRAN_BUILD different from $NEMOH_FORTRAN
5.2.5.2. Go to $FORTRAN_BUILD folder by running cd $FORTRAN_BUILD .
5.2.5.3. Delete the file CMakeCache.txt and the directory CMakeFiles if it exists by running rm -rf CMakeCache.txt CMakeFiles/
5.2.5.4. Run cmake -DCMAKE_Fortran_COMPILER="gfortran" $NEMOH_FORTRAN to generate the Makefiles
5.2.5.5. Run make to build the library
5.2.5.6. The library libnemoh.so will be created

5.2.6. Compile the nemoh python against the nemoh Fortran
5.2.6.1. Go to $NEMOH_PYTHON/nemoh
5.2.6.2. Make sure the $FORTRAN_BUILD is in the library search paths for compilation and for linking by running:
export LD_LIBRARY_PATH=$FORTRAN_BUILD
export LDFLAGS="-L$FORTRAN_BUILD"
5.2.6.3. Run python setup.py build_ext --inplace to build the python module in place

5.3. Windows Instructions

5.3.1. *Install MinGW 4.8.1*
5.3.1.1. You should install posix threads MinGW from

http://sourceforge.net/projects/MinGWbuilds/files/host-windows/releases

The 64 bits is located at http://sourceforge.net/projects/MinGWbuilds/files/host-windows/releases/4.8.1/64-bit/threads-posix/sjlj/x64-4.8.1-release-posix-sjlj-rev5.7z/download

The 32 bits is located at http://sourceforge.net/projects/MinGWbuilds/files/host-windows/releases/4.8.1/32-bit/threads-posix/sjlj/x32-4.8.1-release-posix-sjlj-rev5.7z/download

5.3.1.2. Download the binaries for your platform and extract it somewhere. By default the 64 bits get extracted in a directory named "MinGW64". Let's call this directory **$MINGW_ROOT**

5.3.1.3. Now add the full path to the folder **$MINGW_ROOT\bin** and **$MINGW_ROOT\lib** to your Path. Make sure those directories are at the left most (the beginning) of the Path. See **Setting Path on Windows** sub section for more information

5.3.2. Install CMake 2.8

5.3.2.1. Download and install cmake from http://www.cmake.org/files/v2.8/cmake-2.8.12.2-win32-x86.exe

5.3.2.2. Choose to add Cmake to the path for all users. By default Cmake doesn't put the path at the beginning.
So, you need to make sure the cmake path is at the beginning of your path. It is by default " C:\Program Files (x86)\CMake 2.8\bin". See **Setting Path on Windows** sub section for more information

5.3.3. Install Lapack and Blas

Download and Install lapack and blas from http://icl.cs.utk.edu/lapack-for-windows/lapack/#libraries Choose the dll libraries for MinGW

The 64 bits blas is http://icl.cs.utk.edu/lapack-for-windows/libraries/VisualStudio/3.5.0/Dynamic-MINGW/Win64/libblas.dll
And the 64 bits lapack is
http://icl.cs.utk.edu/lapack-for-windows/libraries/VisualStudio/3.5.0/Dynamic-MINGW/Win64/liblapack.dll

5.3.3.1. Copy them to **$MINGW_ROOT\lib** (the lib directory inside MinGW installation root)

5.3.4. Install Anaconda >= 2.1.0

5.3.4.1. Download and install Anaconda Windows version greater or equal to 2.1.0 from http://continuum.io/downloads . *The 64 bits Windows version is located at* http://09c8d0b2229f813c1b93-c95ac804525aac4b6dba79b00b39d1d3.r79.cf1.rackcdn.com/Anaconda-2.1.0-Windows-x86_64.exe

5.3.4.2. When installing accept adding anaconda to the path and using it's python version as the default python.
By default Anaconda would not put it's path to the beginning of the Windows Path.

You need to move the anaconda paths to the beginning of the Path list.

See **Setting Path on Windows** sub section for more information

For me the paths were C:\Users\yedtoss\Anaconda (the most important) and C:\Users\yedtoss\Anaconda\Scripts. You should replace C:\Users\yedtoss\Anaconda by the location where you install Anaconda

5.3.5. *Compile the Nemoh Fortran*

5.3.5.1. Start Powershell (Or windows cmd if you prefer it)
Make sure that all paths were correctly set. If not then you should close Powershell/Cmd, set the path and reopen it. Basically run python --version, cmake --version, gfortran --version and verify that they come from the one you just installed

5.3.5.2. Create a build directory $FORTRAN_BUILD different from $NEMOH_FORTRAN
5.3.5.3. Go to $FORTRAN_BUILD folder.
5.3.5.4. Delete the file CMakeCache.txt and the directory CMakeFiles if it exists *or better make sure* $FORTRAN_BUILD is empty
5.3.5.5. Run *cmake -DCMAKE_Fortran_COMPILER="gfortran" "*$NEMOH_FORTRAN**"** -G "MinGW Makefiles"

Please note the "" surrounding $NEMOH_FORTRAN. You need it even if the directory does not contain space. Make sure **"**$NEMOH_FORTRAN" is a full path to avoid any cmake bug

5.3.5.6. Run mingw32-make and you will get libnemoh.dll and libnemoh.dll.a
5.3.5.7. Copy both generated file to **$MINGW_ROOT\lib**

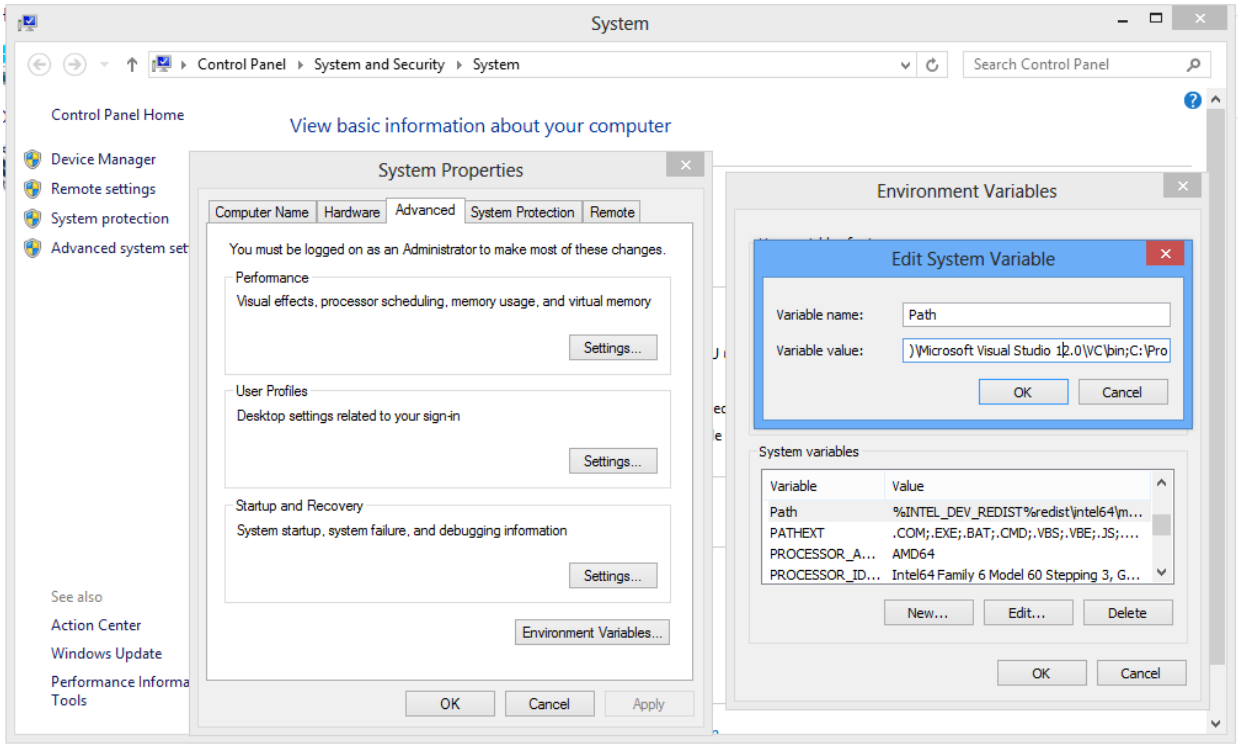
5.3.6. Compile the nemoh python against the nemoh Fortran

5.3.6.1. Go to $NEMOH_PYTHON\nemoh
5.3.6.2. Run python setup.py build_ext --inplace

5.3.7. **Setting Path on Windows**

You need to go to Computer → Right click and choose Properties → Advanced System Settings → Advanced Tabs → Environment Variables → Look for path.

A screenshot



Note that when setting the path, you need to separate the different directories by **;** Also make sure your new directory is not at the end of the list but at the beginning because Windows read the environment from left to right.

5.4. MAC OS X Instructions

We will use homebrew to install most software

5.4.1. Install brew: ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
(If you did not have XCode Command Line Tools, it will request it, install it and when done press enter on the terminal)

5.4.2. Run brew doctor

5.4.3. Install gcc and gfortran brew install gcc You can ignore any warning about multilib. It won't affect us

5.4.4. Install cmake brew install cmake

5.4.5. Download and Install anaconda from http://continuum.io/downloads
You can get the 64 bits version from http://09c8d0b2229f813c1b93-c95ac804525aac4b6dba79b00b39d1d3.r79.cf1.rackcdn.com/Anaconda-2.1.0-MacOSX-x86_64.pkg
We will export anaconda bin to the Path before compiling and using the nemoh python module

5.4.6. *Compile the Nemoh Fortran*

5.4.6.1. Create a build directory $FORTRAN_BUILD different from $NEMOH_FORTRAN
5.4.6.2. Go to $FORTRAN_BUILD folder cd $FORTRAN_BUILD .
5.4.6.3. Delete the file CMakeCache.txt and the directory CMakeFiles if it exists *rm -rf CMakeCache.txt CMakeFiles/*
5.4.6.4. Run *cmake -DCMAKE_Fortran_COMPILER="gfortran"* $NEMOH_FORTRAN
5.4.6.5. *Generate nemoh Fortran library by running make*
The library libnemoh.dylib will be created
(If you get a warning about cmake policy ignore it)

5.4.7. Compile the nemoh python against the nemoh Fortran
5.4.7.1. Go to $NEMOH_PYTHON/nemoh
5.4.7.2. Make sure you are using python from Anaconda
By running export PATH=/Users/tcs/anaconda/bin:$PATH
Replace /Users/tcs/anaconda/bin according to the location where anaconda was installed
5.4.7.3. Make sure the $FORTRAN_BUILD is in the library search paths for compilation and for linking by running
export LD_LIBRARY_PATH=$FORTRAN_BUILD
export LDFLAGS="-L$FORTRAN_BUILD*"*
5.4.7.4. Run python setup.py build_ext --inplace

Note that in the above process we did not explicitly install lapack or blas libraries. This is because it is implicitly installed with brew (XCode Command Line Tools). If for some reason you receive an error when linking against lapack or blas you can install a custom version by running brew install https://raw.githubusercontent.com/Homebrew/homebrew-dupes/master/lapack.rb

Also note that the Intel Fortran Compiler (ifort) is fully supported on all three platforms. It comes bundled with lapack and blas so If you choose it, you won't need them.

Installing VMTK (Optional, you can skip)

VMTK is used by $NEMOH_PYTHON/nemoh/export_tec.py to export the generated .tec files to other format. To install it for Mac, Windows or Linux follow instructions at
http://www.vmtk.org/documentation/installation.html
Note that unless you are using the .egg for Anaconda (Windows only), you should not use it with the python from Anaconda. More specifically, on Linux and Mac you have to use it with the built-in python when installing or using it.

6. Starting

You need to setup all environment variables as described in the deployment instructions.
You should also configure the application as described in the configuration section
Enter the directory $NEMOH_PYTHON/nemoh/

Run *python preprocessor.py* to run the preprocessor
Then run the solver with *python solver.py*
Finally run the post processor with *python postprocessor.py*

7. Verification

Setup your environment using the deployment instructions.
Then run the tools by following the Starting section

By default, the cylinder example has been configured. You can run other example by modifying the configuration. You can find additional cases files in $NEMOH_FORTRAN/Verification folder
The hdf5 file db.hdf5 will be generated.

You can visualize it with HDFView from http://www.hdfgroup.org/products/java/release/download.html
It has a version for Windows, MAC and Linux

For example, to install the Ubuntu 64 bits:

- Download http://www.hdfgroup.org/ftp/HDF5/hdf-java/current/bin/HDFView-2.10.1-centos5-static64.tar.gz
- Extract it to /tmp so that you have /tmp/HDFView-2.10.1-Linux, then locate the file hdfview.sh inside it. It should be located at /tmp/HDFView-2.10.1-Linux/HDF_Group/HDFView/2.10.1/bin/hdfview.sh
- Open the file hdfview.sh and set INSTALLDIR to ■■
- Enter the bin directory (/tmp/HDFView-2.10.1-Linux/HDF_Group/HDFView/2.10.1/bin)
- Make sure you have java from Sun, JRE is enough. You can use JRE 6 or JRE 7
- Run bash hdfview.sh

Then open the hdf5 file (Click File → Open)

Enter the directory $NEMOH_PYTHON/nemoh/
Run *python preprocessor.py* to run the preprocessor
Then run the solver with *python solver.py*
Finally run the post processor with *python postprocessor.py*

Ignore the warning of the type "Warning: normal vector of panel 14 points towards the x axis". It is caused by the input mesh used.
The default example for test is $NEMOH_PYTHON/test_files/higher_order_panel

To run the higher order panel method, set USE_HIGHER_ORDER settings to True (the default). To run previous lower order panel method set it to False.

The lower order results are located in *$NEMOH_PYTHON/test_files/higher_order_panel/db_old.hdf5* and the higher order panel in $NEMOH_PYTHON/test_files/higher_order_panel/db.hdf5

SPEED VERIFICATION

The current implementation of the higher panel method is mathematically faster than the lower order panel method for the same accuracy as described in reference R1. Actually it should be faster than the lower order panel method and still give a better accuracy.

This cannot be observed currently because these analysis are based on the fact that the geometrical

description of the bodies leads to a small number of complex sub-divisions. Currently, the Mesh software generating the input mesh to Nemoh and the preprocessor only support flat quadrilateral which leads to a huge number of sub-divisions. See first point of section 8.

We have decreased the number of problems to solve to 7 and we have set the number of panels per patch and the order of B-Spline to 1 so that running time is not too huge. As far as a flat quadrilateral is used for the geometry description these values should only be changed with precaution

ACCURACY VERIFICATION

The previous code is implementing the lower order panel method. In this method, the velocity potential on a panel is assumed to be constant (equal to the value at the centroid of the panel). This approximation is clearly not good enough and will lead to less accurate result. The current implementation of the higher order panel method described the velocity potential at each point by a product of two B-Splines. This will lead to better accuracy.
Mathematically speaking, the current implementation is more accurate than the previous one according to the analysis of reference R1.

This means that we can not expect the current values to match the values of the previous implementation.

A plan to check the accuracy of this implementation is to derive a problem for which we can compute exactly (mathematically speaking) the solutions (potential, forces ...). Then, we will run the lower order panel method with a flat quadrilateral description of the body and then the higher order panel method with a suitable geometric description of the body (For example B-Spline).
The higher order panel method error is expected to be lower than the lower order panel one's.

This plan can not be executed currently as such problem with known theoretical result is not provided and the current Mesh software generating input meshes to Nemoh only support flat quadrilateral.

Currently, we are computing values at the centroid of patches to make verification easier.
It can be manually verified that many values in forces, fk_forces and potential match the one from the lower order panel method. (See hdf5 file and path results/).

8. Limitations of the current higher panel method

- The current implementation can work with any geometry description (the parametric space transformation routine and it's Jacobian needs to be modified according to each geometry). Currently though, we are using the geometry description of the Mesh software generating input mesh for Nemoh. This tool will generate flat quadrilaterals. Each quadrilateral is assumed to be a patch for the current implementation. This will lead to low accuracy of the higher panel method and to a huge running time. This is so, because to the number of panels has to be high to approximate the true geometry of the Mesh.

To solve this issue, the Mesh software used by Nemoh needs to be modified to provide a more complex

description of the mesh by the usage of B-Spline for example or by using exact equations of the body if available. See section 7.6 and 7.8 of reference R2 for possibilities.
Once the Mesh software is modified it is also expected to modify the preprocessor to match the changes. Typically the number of subdivisions generated by this mesh should not be too huge. A value of 10 is fine. Each subdivision can be complex and accurately described exactly or by a B-Spline.
These changes should lead to a huge speed-up and better accuracy. It should be a top priority.

- The current implementation does not compute the Kochin function and thus does not compute the drift forces, yaw moment and wave elevation (in postprocessing module). The Kochin function is not computed because the previous method to compute it according to references R5 and R6 use the source strength which is not available in the implementation of the higher panel method. Other technique to compute the Kochin function should be performed.

- In the higher panel method, it is possible to compute the velocity potential at the boundary by using equation (7.4) of reference R2. This equation is precise and does not needs an integration as equation (15.29) of reference R2. For a given point in Cartesian coordinate the current implementation is using equation (15.29) instead of (7.4). This is because the routine to convert a Cartesian coordinate to parametric space (transform_cartesian_to_uv) is mocked currently so the method COMPUTE_POTENTIAL_BOUNDARY_PATCH cannot be used. If speed is needed the conversion should be implemented.

- The higher panel method does not support the technique for symmetry in the body as described in R5. This is mainly because the analysis done in R5 is only valid for the lower order panel method. It should be however possible to find a similar analysis for the higher order panel method

To run your problem containing a symmetry in the mesh, disable the symmetry switch by setting the second number in the mesh file to 0 before running the preprocessor. Alternatively you can also set the second number in input/bodies/body1/mesh of the input hdf5 file to 0. You should do this for all bodies of your problem.

An error message will be given and program will stop if an attempt to use the new method in such case.

- The current implementation of the higher panel method does not support computing the green function using an ODE as supported in the lower order panel method. If such an attempt is made, a warning will be showed and the method to compute the green function will fall back to the default (tabulation, series approximation and interpolation).

- There are opportunities to speed-up the current implementation. When the frequency is not changed, the influence coefficients should not be re-computed and other similar techniques used in the lower panel method could be used. This changes are not as important as the changes to the Mesh software presented earlier.

9. References

R1 **A three dimensional higher order panel method based on B-splines.**
http://dspace.mit.edu/bitstream/handle/1721.1/11127/34279481.pdf?sequence=1

R2. **Wamit User Manual** http://www.wamit.com/manualupdate/V70_manual.pdf

R3. **A higher order panel method for large-amplitude simulation of bodies in waves.**
http://dspace.mit.edu/bitstream/handle/1721.1/9708/42663926.pdf?sequence=1

R4 **Fast Multipole Boundary Element Method: Theory and Applications in Engineering**
https://books.google.bj/books?id=1p4SCnM5UIYC http://www.amazon.com/Fast-Multipole-Boundary-Element-Method-ebook/dp/B002TRJ076

R5. **Amélioration des performances des codes de calcul de diffraction-radiation au premier ordre**
http://lheea.ec-nantes.fr/lib/exe/fetch.php/emo/nemoh/delhommeau_papier052jh.pdf

R6. **Les problemes de diffraction-radiation et de resistance de vagues : etude theorique et resolution numerique par la methode des singularites**
http://lheea.ec-nantes.fr/lib/exe/fetch.php/emo/nemoh/thesedsgd010787.pdf

10. Resource Contact List

Name	Resource Email
yedtoss	