Session 3: Compilation and Installation

Eric Nielsen
(Bill Jones)
Learning Goals

• What this will teach you
  – How to configure and compile the FUN3D suite
  – Configuration options
    • Enable/Disable capabilities
    • Specify the location of 3rd party libraries and tools
  – How we do it
• What you will not learn
  – How to build/install 3rd party libraries and tools
  – How to configure your system to compile Fortran 90/MPI code
• What should you already know
  – How to navigate through a *NIX shell
    • mkdir
    • cd
    • Absolute/relative paths
Setting

• Background
  – FUN3D uses the *de facto* industry standard build environment provided by GNU Autotools
  – Build of the FUN3D distribution does **not** require Autotools on your system
  – Provides localization through options to a configuration script

• Compatibility
  – Requires a Bourne Shell derivative (*NIX, OS X, MinGW, etc.)
  – Requires GNU `make`
  – Requires a functioning Fortran 95 compliant compiler (some optional capabilities rely on Fortran 2003 additions)
  – May not work with *non-standard* installation of 3rd party libraries
  – DiRTLib and SUGGAR++ assumptions
    • Required library names: `libp3d.a`, `libdirt.a`, `libdirt_mpich.a`, `libsuggar.a`, and `libsuggar_mpi.a`
  – Developers will need GNU Autotools installed
Nuts and Bolts (1 of 4)

- Two step process
  - `configure` selects capabilities and localizes to system
  - `make` creates executables

- Distribution contains a `configure` script
  - Familiar to Linux users/administrators who have built open source packages
  - Must **NOT** be edited by hand
  - Custom localization through command line options

- The `configure` script creates **Makefiles**
  - **Makefiles** are customized/localized for a specific *configuration*
  - Not practical for human consumption
  - Must **NOT** be edited by hand
    - All localization is managed through the `configure` script
  - Checks various details required by compilation
    - **Fails fast** (prior to compilation of FUN3D) if problems are detected with the configuration options (no compiler, missing libraries, etc.)
Nuts and Bolts (2 of 4)

• `configure --help` will show a list of all options
  - Command line options
  - Environment variables
  - Order independent (uses last value if specified multiple times)

• FUN3D optional Features of general interest

  --disable-FEATURE do not include FEATURE
  (same as --enable-FEATURE=no)
  --enable-FEATURE[=ARG] include FEATURE [ARG=yes]

  --enable-design build Adjoint design tools [no]
  --enable-hefss build with High Energy Physics [no]
  --enable-ftune tailor Fortran compiler options for FUN3D [yes]
Nuts and Bolts (3 of 4)

- **FUN3D optional Packages of general interest**

  --with-PACKAGE[=ARG]  use PACKAGE [ARG=yes]
  --without-PACKAGE  do not use PACKAGE (same as --with-PACKAGE=no)

  --with-mpi[=ARG]  Path to MPI library (installation root)
  --with-mpibin[=ARG]  MPI binary directory (relative, absolute, without)
  --with-mpif90[=ARG]  MPI compiler wrapper (relative, absolute, without)
  --with-mpiexec[=ARG]  MPI execution startup script (relative, absolute, without)
  --with-mpiinc[=ARG]  Path to “mpif.h” (relative, absolute, without)
  --with-metis[=ARG]  Metis library install path (contains libmetis.a)
  --with-ParMetis[=ARG]  ParMetis library install path (contains libparmetis.a)
  --with-dirtlib[=ARG]  use DiRTlib overset library (contains libdirt.a)
  --with-suggar[=ARG]  use SUGGAR overset library (contains libsuggar.a)
  --with-tecio[=ARG]  Tecplot I/O library install path (contains tecio.a)
  --with-refine[=ARG]  use refine adaptation package (installation root)
  --with-refineFAKEGeom[=ARG]  to specify refine FAKEGeom libs [-lFAUXGeom]
  --with-knife[=ARG]  use Knife cut cell package (installation root)
  --with-CGNS[=ARG]  CGNS library path (installation root)
  --with-PORT[=ARG]  use PORT optimization library (contains libport.a)
  --with-NPSOL[=ARG]  use NPSOL optimization library (contains libnpsol.a)
  --with-KSOPT[=ARG]  use KSOPT optimization library (contains libksopt.a)
Nuts and Bolts (4 of 4)

- FUN3D environment variables of general interest

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FC</td>
<td>Fortran compiler command (overridden by <code>--with-mpif90</code>)</td>
</tr>
<tr>
<td>FCFLAGS</td>
<td>Fortran compiler flags (adds to default unless <code>--disable-ftune</code>)</td>
</tr>
<tr>
<td>LDFLAGS</td>
<td>linker flags, e.g. <code>-L&lt;lib dir&gt;</code> if you have libraries in a nonstandard directory <code>&lt;lib dir&gt;</code></td>
</tr>
<tr>
<td>CC</td>
<td>C compiler command</td>
</tr>
<tr>
<td>CFLAGS</td>
<td>C compiler flags</td>
</tr>
<tr>
<td>CPPFLAGS C/C++</td>
<td>preprocessor flags, e.g. <code>-I&lt;include dir&gt;</code> if you have headers in a nonstandard directory <code>&lt;include dir&gt;</code></td>
</tr>
<tr>
<td>CPP</td>
<td>C preprocessor</td>
</tr>
</tbody>
</table>

- `make` is used to build the executables
  - Will reside in respective directories (e.g. `nodet` is in `FUN3D_90`)
Basic Operation

- Construct the vanilla **serial** executable
- Unpack your FUN3D distribution
  - Creates a directory “fun3d-11.3-48116”
- Enter the FUN3D distribution directory
- Run the `configure` script and build executables with `make`
  
  ```
  $ ./configure
  $ make
  ```

- Note that this will search for a supported compiler in your path
  - Chooses the first one found based on pre-defined order
  - Override this with the **FC=mycompiler** option
    
    - Serial version or when using `--without-mpif90`
    - MPI configurations will use the `--with-mpif90` wrapper if given
Did It Work? Expected Output

Flow solver executable created as “FUN3D_90/nodet”
Extended Operation
(How we do it)

- Create a parallel version of the code with design capability enabled
- Build in a separate configuration subdirectory
  - Stores object code and executables only
  - Does not pollute the source tree with object code
  - Multiple configurations utilize the same source

```bash
$ mkdir mpi
$ cd mpi
$ ../configure --enable-design --with-mpi=/path/to/mpi
  --with-metis=/path/to/metis
  --with-ParMetis=/path/to/parmetis
$ make
```
Did It Work? Expected Output

... Configuration (FUN3D):
  Source code location: ..
  Version: 11.1-46128
  Compiler: /path/to/mpi/bin/mpif90
  Compiler flags: -O2 -ip -align
                -fno-alias -vec-report0
  Linker flags: -Vaxlib -lm
  Dependencies: Normal

build:
  Design modules: yes
  High Energy Physics: no
  Cmplx Variable Tools: no
  Dynamic Partitioning:

bindings:
  refine: no
  CAPRI support: no
  knife: no
  MPI support: yes
  OpenMP support: yes
  MPI: /path/to/mpi
  Metis: /path/to/metis
  ParMetis: /path/to/parmetis
  ParMGridGen: no
  Tecplot I/O: no
  6DOF libraries: no
  DiRTlib support: no
  SUGGAR support: no
  CGNS support: no
  PORT support: no
  NPSOL support: no
  KSOPT support: no
  SMEMRD support: version 1.3.1

- Executables created under the configuration directory
  - FUN3D_90/nodet_mpi, Adjoint/dual_mpi, Design/opt_driver
Troubleshooting/FAQ (1 of 3)

- Problems
  - “checking for Fortran compiler default output file name... configure: error: Fortran compiler cannot create executables
  See `config.log` for more details.”
  - Make sure that Fortran compiler works in your environment
    - Adjust PATH, load appropriate GNU modules, MPI installation, etc.
  - Limited check of `configure` options
    - Bad “--enable-*" and “--with-*" options silently ignored
  - Option values containing spaces must be quoted from shell
    - e.g. FCFLAGS="-g -O2 -m32 -fno-common"
  - Do **NOT** configure in top level distribution directory and then try to make individual configuration directories
    - `make distclean` to clean a previous configuration of the source
  - Look/send “config.log” file
    - Also includes configuration options at the top (less quoted values w/ spaces)
Troubleshooting/FAQ (2 of 3)

- Can I...
  - Override the default compiler options?
    - Yes, `--disable-ftune FCFLAGS="--what-ever-you-want"
    - Remember some compilers always need certain options (`-Vaxlib`)
  - Explicitly specify my compiler?
    - You can, with `FC=compiler`, but this will be overridden if using `--with-mpif90`
  - Fix anything by manually editing the `configure` script or `Makefiles`?
    - **NO**! and we cannot support any such action
    - Anything that you can safely change is governed by a configure option
  - Install the executables in a central location?
    - Yes, `make install` will install executables, etc. under the location given by the `--prefix=/your/path` option to `configure`
Troubleshooting/FAQ (3 of 3)

• What if I…
  – Have a proprietary MPI installation?
    • Some HPC resources have proprietary MPI installations using non-standard paths and names
    • Use “--with-mpibin”, “--with-mpiinc”, “--with-mpif90”, and “--with-mpiexec” along with their “—without-*” counterparts as needed to specify the binary and include paths as well as the name for the `mpif90` compiler wrapper and, if needed, the `mpiexec` script
    • Paths can be absolute or relative to the --with-mpi and --with-mpibin values
      $ ./configure --with-mpi=/path/to/mpi
      --with-mpif90=my_mpif90 --without-mpiexec …
  – My MPI executables will not run
    • Check the consistency of your MPI compilation/runtime installations
    • The MPI installation used for compilation is available as MPI Prefix: from
      $ /path/to/nodet/nodet_mpi --version
What We Learned

• How to configure and compile the FUN3D suite
  – Execute `configure` to localize a configuration
  – Build the executables with `make`

• Configuration options
  – Enable/Disable Features
  – With/Without Packages (3rd party libraries and tools)
  – Custom environment variables

• How we do it
  – Use separate configuration subdirectories
    • Keeps source and object code separate
    • Allows multiple configurations under one source
    • Invoke as `../configure`