
Chombo BootCamp

Lecture 3: MultiDim Chombo

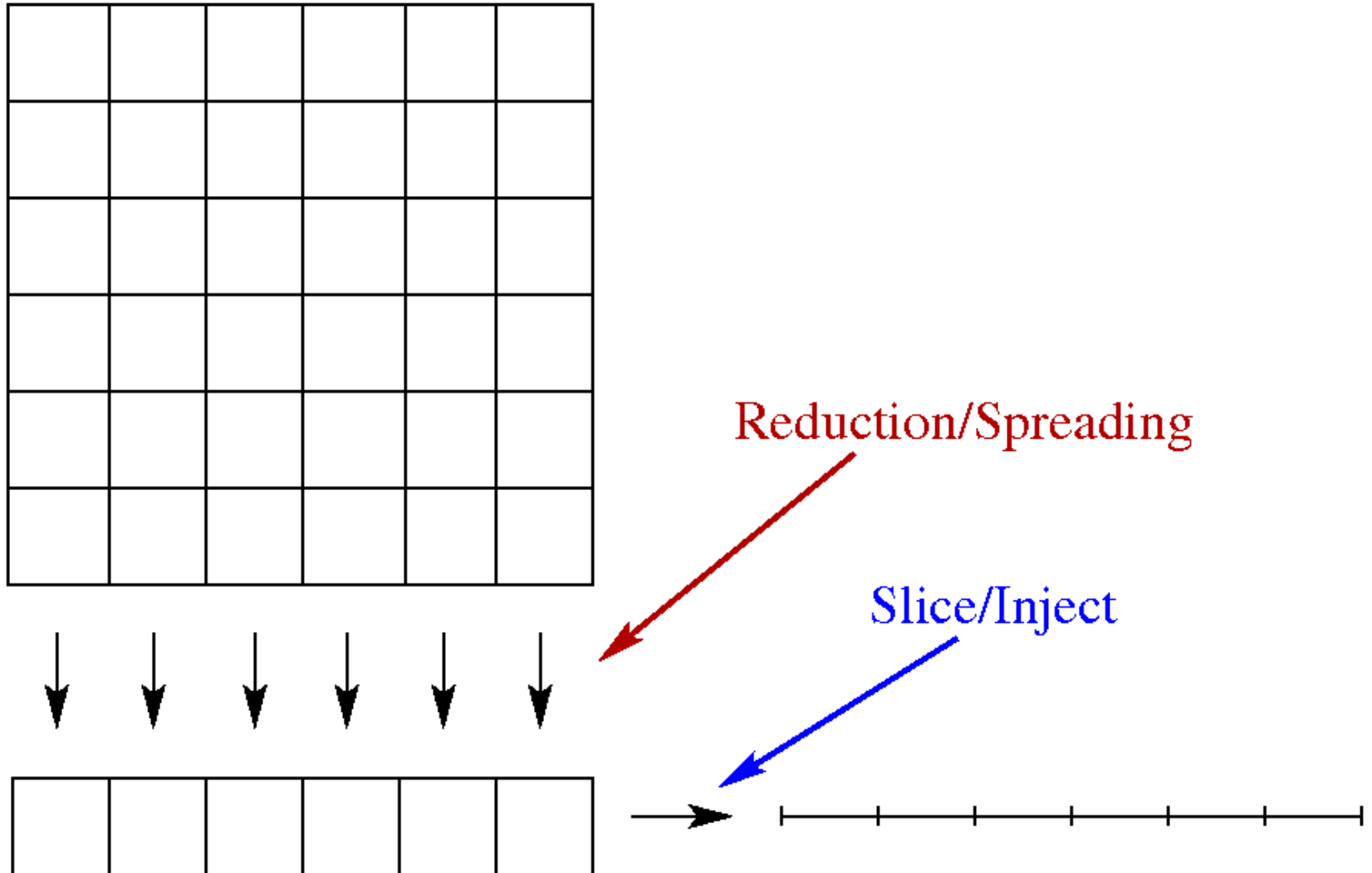
Example: Phase-space

- In some contexts, can be useful to think in terms of a phase space made up of physical and velocity dimensions
 - Example: 2+2D (2 physical, 2 velocity dimensions: (x,y,u,v))
 - Often want to work in both the full 4D and the reduced 2D spaces
- Working example in Chombo MultiDimPhase example
 - `Chombo/example/MultiDimPhase`
 - Document in `ChomboDoc/newDocs/multiDim.*`
 - 1+1D example (x, v) :
 - Hyperbolic advection in 2D: $\frac{\partial f}{\partial t} + \frac{\partial v f}{\partial x} + \frac{\partial a f}{\partial v} = 0$
 - $a = -\frac{\partial \varphi}{\partial x}$
 - 1D Elliptic equation for φ : $\frac{\partial^2 \varphi}{\partial x^2}(x, t) = 4\pi \int f(x, v, t) dv$

MultiDim Chombo

- “normal Chombo -- compile-time dimensionality
 - Dimensionality is pervasive (`SpaceDim`, `D_DECL`, etc macros, `ChF`)
- Solution – namespaces
 - Each dimensionality lives in its own distinct namespace: (`Chombo::D1`)
 - Transdimensional utilities for moving between dimensions
 - (`inject` from `D` to `D+1`, `slice` from `D` to `D-1`)
 - Reduction and Spreading utilities within a single dimension
 - `ReductionCopier`, `SumOp`, `SpreadingOp`
 - Spreading done by defining `ReductionCopier`, then calling `reverse()` function
 - Most of the time, work in a single dimensional namespace
 - Only worry about dimensionality in the interface points

MultiDim, cont



Example : Reduction by summing in y-direction

```
// function to reduce a 2D LevelData<FArrayBox> to 1D by
// summing over the y-direction

void reduce2DTo1D(D1::LevelData<D1::FArrayBox> & destData,
                  D2::LevelData<D2::FArrayBox> & srcData)
{
// create 2D version of 1D boxes
D2::LevelData<D2::FArrayBox> sliceGrids;
D2::SliceSpec slice(1,0);
injectDisjointBoxLayout(sliceGrids, srcData.getBoxes(), slice);

// define sliced data holder
D2::LevelData<D2::FArrayBox> reducedData(sliceGrids,
                                          srcData.nComp);
```

```
// define ReductionCopier to compute intersections
//(sum in the y-direction)

int transverseDir = 1;

ReductionCopier reduceCopier(srcData.getBoxes(), sliceGrids,
1);

SumOp op(transverseDir);

op.scale = 1.0;

/* do summing operation -- sums data in srcData along lines in
the y-direction and places sum in reducedData */

srcData.copyTo(srcData.interval(),
                reducedData, reducedData.interval(),
                reduceCopier, op);

/*finally, take the data in reducedData (which is a 2D object)
and slice to 1D */

sliceLevelData(destData, reducedData, slice);
}
```


MultiDim Builds

- Special build system
- `BaseTools` library has no dimensionality
- Other Chombo libraries can be used with any DIM
 - Special headers placed in `Chombo/lib/include/multidim`
- `Chombo/lib/MultiDim` contains transdim support
 - `Slicing.H.transdim`, `Injection.H.transdim`
- Use 2 GNUmakefiles in local build directory
 - GNUmakefile: specifies `ebase`, `MINDIM`, `MAXDIM`, `MULTIDIM_MAKEFILE`
 - `GNUmakefile.multidim`: specifies libraries and src directories:
 - `1dLibNames`, `2dLibNames`, `3dLibNames`, ...
 - `1dsrc_dirs`, `2dsrc_dirs`, `3dsrc_dirs`, ...
 - `mdsrc_dirs` (files without a compile-time dimension)

GNUmakefile:

```
#the location of Chombo lib dir
CHOMBO_HOME = ../../../../lib
ebase = phase
MINDIM = 1
MAXDIM = 2
# this is the local GNUmakefile which contains this example's
# multidim build info -- libraries and src directory information
MULTIDIM_MAKEFILE = GNUmakefile.multidim

# this is the default target, which just invokes the all-multidim target
all: all-multidim

# this file contains the basic rules used to build multidim codes (using
# the GNUmakefile.multidim in this directory), including the shell script
# which orchestrates the make process
include $(CHOMBO_HOME)/mk/Make.multidim.basic
```

GNUmakefile.multidim

```
## Define the variables needed by Make.example
# the location of Chombo lib dir
CHOMBO_HOME = ../../../../lib

# names of Chombo libraries needed by this program, in order of search.
1dLibNames = OldAMRElliptic AMRTools BoxTools BaseTools
2dLibNames = AMRTools BoxTools
#3dLibNames = BoxTools

# relative paths to source code directories
base_dir = .
1dsrc_dirs = ../util
2dsrc_dirs = ../src ../util ../advectSrc
3dsrc_dirs =
mdsrc_dirs = ../mdsrc

# shared code for building example programs
include $(CHOMBO_HOME)/mk/Make.example.multidim
```