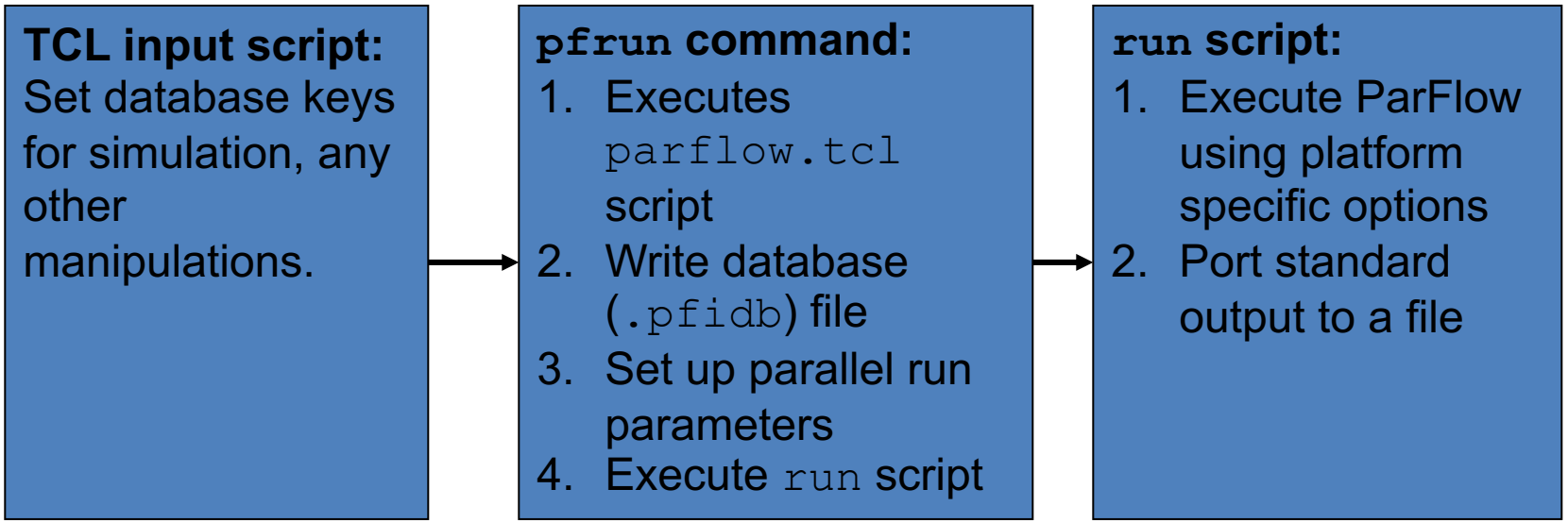
Overview: Interacting with ParFlow

ParFlow Short Course

1. *How do I tell it what I want it to do?*
2. *How do I get my inputs into the model?*
3. *How do I press go?*
4. *What comes out and how do I look at it?*

1. Model input file:
(How you tell ParFlow what to do)



Input Scripts

- TCL/TK scripting language
- All parameters input as keys using `pfset` command
- Keys used to build a database that ParFlow uses
- ParFlow executed by `pfrun` command
- Since input file is a script may be run like a program

Example: Setting up the input grid

Comment character for tcl/tk

```
#-----------------------------------------------------------
# Computational Grid
#-----------------------------------------------------------
pfset ComputationalGrid.Lower.X                0.0
pfset ComputationalGrid.Lower.Y                0.0
pfset ComputationalGrid.Lower.Z                0.0

pfset ComputationalGrid.NX                     30
pfset ComputationalGrid.NY                     30
pfset ComputationalGrid.NZ                     30

pfset ComputationalGrid.DX    10.0
pfset ComputationalGrid.DY              10.0
pfset ComputationalGrid.DZ    .05
```

Coordinates (length units)

Grid dimensions (integer)

Cell size (length units)

Example: Setting up the timing

```
#---------------------------------------------------------
# Setup timing info
#---------------------------------------------------------

pfset TimingInfo.BaseUnit      1.0

pfset TimingInfo.StartCount    0

pfset TimingInfo.StartTime 0.0
pfset TimingInfo.StopTime      300.0

pfset TimingInfo.DumpInterval  30.0

pfset TimeStep.Type     Constant

pfset TimeStep.Value    10.0
```

Sets time units for time cycles (T)

Initial output file number

Start and finish time for simulation (T)

Interval to write output (T)
-1 outputs at every timestep

Timestep type

ΔT (T)

Best practices for building an input file:

- Start from an existing script:
  - Look at the annotated input scripts in the manual (Section 3.6)
  - Look at the test problems that come with ParFlow (See list in section 3.5)
- Get the details on every input key from the manual (Section 6)

2. Reading gridded files *(How you get your inputs into ParFlow)*

- Some keys allow you to specify a file as your input. Like this:

```
pfset GeomInput.indi_input.InputType      IndicatorField
pfset GeomInput.indi_input.GeomNames      "s1 s2 s3 g1 g2 g3”
pfset Geom.indi_input.FileName            "IndicatorFile.pfb"
```

- After getting correct gridded inputs you will need to convert to PFB before they can be read into the model

ParFlow File Types

- PFB: **P**ar**F**low **B**inary. ParFlow's native file type, can be written, read into ParFlow, read into and written by PFTools.
- SILO. VisIt's native file type, can be written by ParFlow, read into and written by PFTools

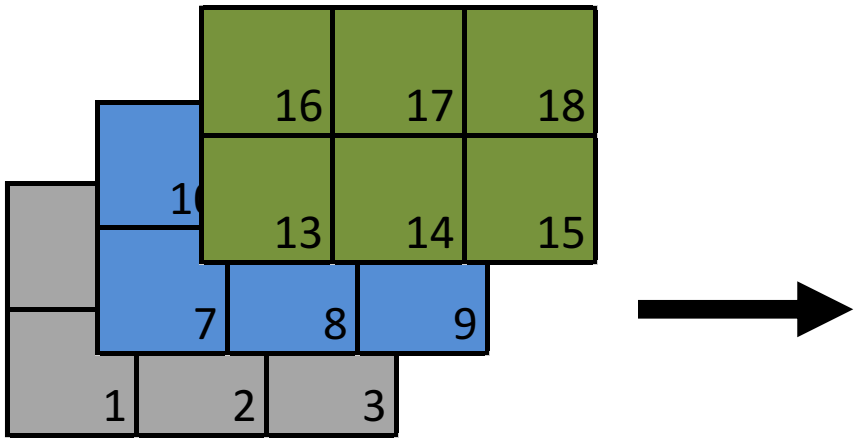
Getting from raster files to ParFlow

Start with a raster which will be formatted as a matrix and convert to a vector in the correct ParFlow order with the grid dimensions at the top

9	10	11	12
5	6	7	8
1	2	3	4

4 3 1
1
2
3
4
5
6
7
8
9
10
11
12

3D ParFlow Inputs



3 2 3
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

Getting from raster files to ParFlow

- Refer to *Manual 6.2 – 6.7* for details on file format
- Start in the lower left corner of the bottom layer and work your way to the upper right corner of the top layer looping over x, y and z in that order
- Header is nx ny nz

```
<integer : NX> <integer : NY> <integer : NZ>
FOR k=0 TO  <nz>-1
BEGIN
    FOR j=0  TO   <ny>-1
    BEGIN
        FOR i= 0 TO  <nx>-1
        BEGIN
            <double : data_ijk>
        END
    END
END
```

PFTools Commands (§4.2)

- Many commands load and write files
- `pfload` reads files that are parflow binary, simple binary and ascii
- `pfsave` writes files that are parflow binary, simple binary and ascii
- One a dataset is loaded (from a file) it may be manipulated with many different tools commands (e.g. convert pressure head to head potential)

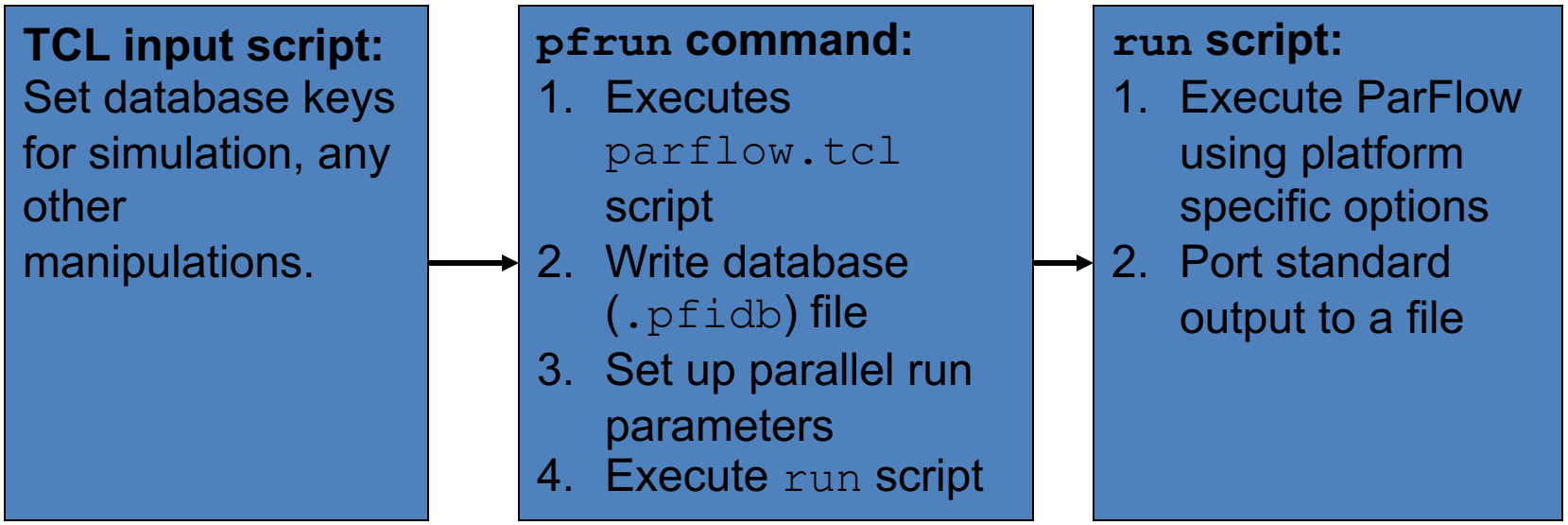
Getting from raster files to ParFlow

- If you generate your ParFLow inputs as text files you will need to convert to PFB before they can be read into the model

- You can do this using PFTools. For example:

```
set       input          [pfload -sa input.sa]
pfsetgrid {nz ny nz} {x0 y0 z0} {dx dy dz} $input
pfsave $input -silo input.silo
pfsave $input -pfb  input.pfb
```

3. Running ParFlow simulations (*How to press go*)



Running ParFlow

- `pfrun` command
  - Builds database of keys
  - Executes program
- Some error checking of keys
- Actual command line runs executable or mpirun's executable
- May run parflow code more than once in single script
- Need parflow package/header information

Running ParFlow (input file)

Mandatory Content at the top of your TCL script:

```
# Import the ParFlow TCL package

lappend auto_path $env(PARFLOW_DIR)/bin
package require parflow
namespace import Parflow::*

…
pfrun myrun
```

Load the parflow tcl package, no pf commands work without this

Run parflow, project name is myrun, this dictates all output names

To run the model:

tclsh runscript.tcl

Parallelization: splitting your problem up across multiple processors

- Domain parallelized by specifying number of processor divisions in x,y,z
- Parallelization done on computational domain
- Done using P,Q,R values
  - Total processors=P*Q*R
  - Domain divided by nx/P, ny/Q, nz/R
- Load balancing issues
- Usually keep R as 1 and split in the x and y directions only

Parallelization (input file)

```
pfset Process.Topology.P 1
pfset Process.Topology.Q 1
pfset Process.Topology.R 1
```

Single processor simulation,
P,Q,R are integer values

```
pfset Process.Topology.P 4
pfset Process.Topology.Q 2
pfset Process.Topology.R 1
```

Eight processor simulation,
P*Q*R=4*2*1=8

Distributing Files (§4.2)

- ParFlow reads and writes parallel files
- One portion of the file per processor (except for sequential/shared memory build)
- ParFlow binary files (.pfb) must be distributed (split up) before being read in
- ParFlow binary files (.pfb) must be undistributed at the end of the simulation
- Two tools to do this, `pfdist` and `pfundist`, may be run directly in tcl input script.

File Parallelism

- ParFlow has several options for parallel io
  - PFB may be distributed as *n* files or as a single file with companion file (.dist)
  - SILO has two options, PMPIO where *n* processors write to *m* files and regular where *n* files are written
- The best file type depends upon application

Distributing Files (input file)

```
pfdist my.input.file.pfb
```

Distribute an input file
Must have specified processor topology, can happen anywhere in script before pfrun command

```
pfundist default_over

pfundist my.input.file.pfb
```

First line undistributes an entire run

Second line undistributes a particular file

*** You can dist and undist files using separate tcl scripts outside your main model run or you can do it all in one step**

4. Handling outputs
(*What comes out and how to look at it*)

Running ParFlow (file structure)

- Project name is the base for all output
- Most output is *project*.out.*var*.*time*.ext

For a project called 'myrun'

Log files:

```
myrun.out.log
myrun.out.kinsol.log
```

Perm/porosity files:

```
myrun.out.perm_x.pfb
myrun.out.porosity.pfb
```

Pressure/Saturation files:

```
myrun.out.press.00001.pfb
myrun.out.satur.00001.pfb
```

Output time step, 00000 is initial, integer values depending on output times

Mask file:

```
myrun.out.mask.00000.pfb
```

The mask is a file of zero's and ones, 0=inactive cell, 1=active cell

Other/diagnostic files:

```
myrun.pfidb
myrun.out.pftcl
myrun.out.txt
```

Parflow database

Line output

Visualizing Outputs

ParaView:

- Free, developed by Kitware Inc
- https://paraview.org
- VTK and pfb formats supported

Visit:

- Free, developed at LLNL
- (http://www.llnl.gov/visit/)
- VTK and SILO format, which has many options within ParFlow (converting or IO), fully-supported

PFTools

- TCL keys that you can use to manipulate ParFLow inputs and outputs:
  - Extract parts of your domain to look at
  - Calculate water balance components
  - Convert pfb outputs to other file types