Gridding

How to setup spatial grids and control time steps

ParFlow Short Course

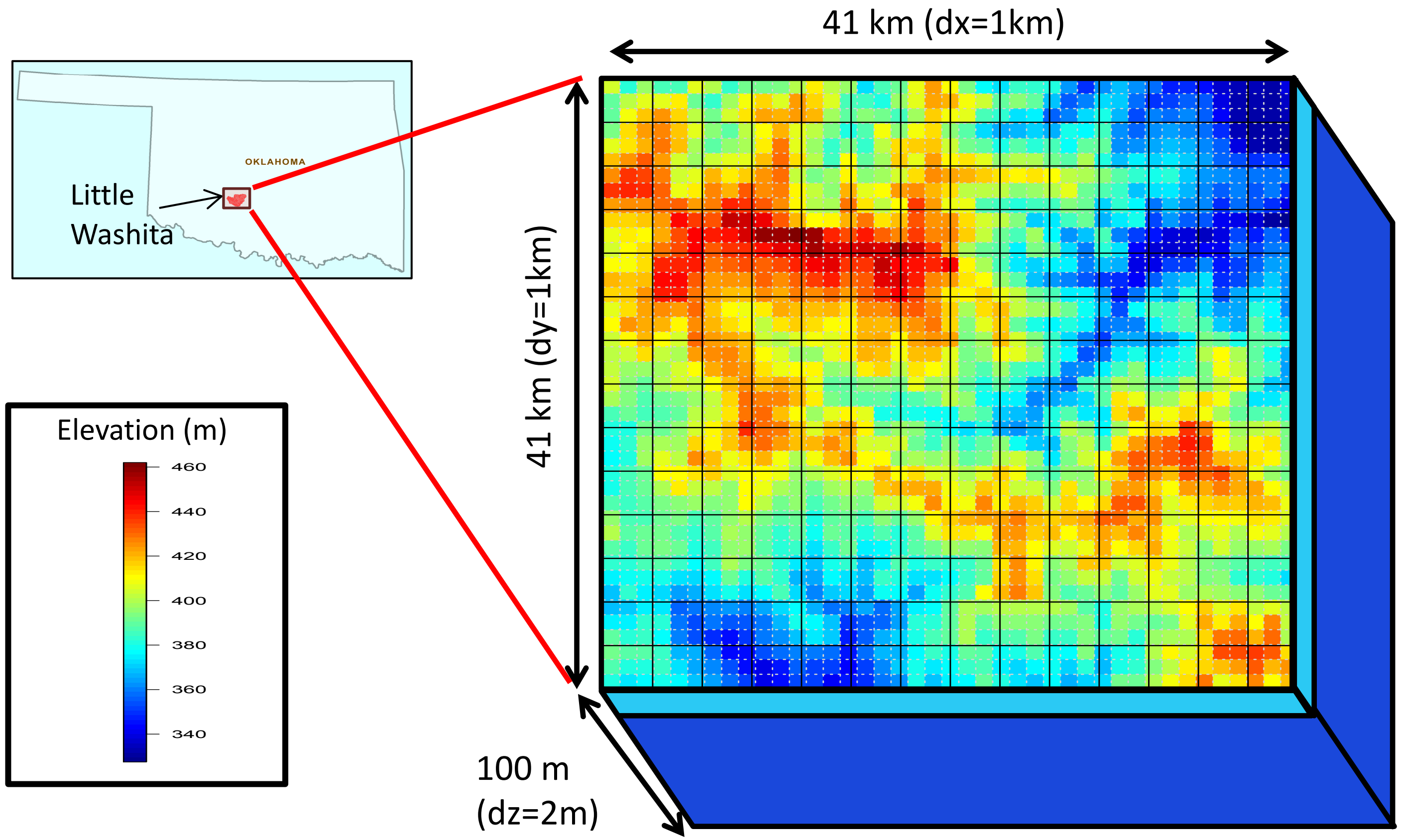
Module 1

Learning Objectives:

At the end of this module students will understand:

- The basics of using keys to setup a ParFlow Grid
- How active and inactive cells are handled
- Terrain following vs standard grid option
- Variable DZ
- How to set the stop and start times for a run
- How to setup the timesteps
- How to building a domain with a solid file and why you might want to do this
- How to build a box domain and why you might want to do this

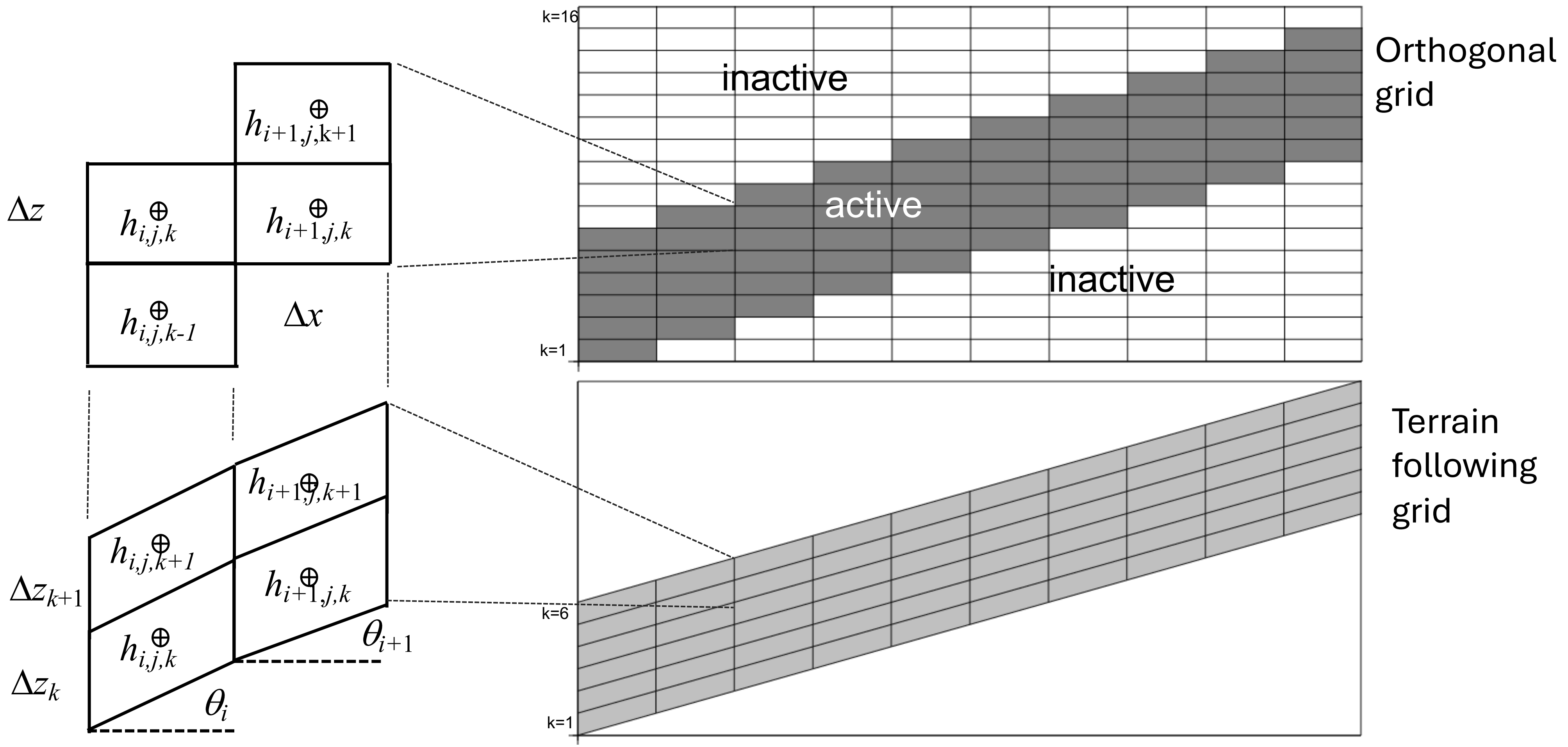
Lateral resolution and domain extent



```
# Define the number of grid blocks in the domain.
model.ComputationalGrid.NX = 64
model.ComputationalGrid.NY = 32
model.ComputationalGrid.NZ = 10

# Define the size of each grid cell. The length units are the
same as those on hydraulic conductivity, here that is meters.
model.ComputationalGrid.DX = 1000.0
model.ComputationalGrid.DY = 1000.0
model.ComputationalGrid.DZ = 200.0

#Locate the origin in the domain.
model.ComputationalGrid.Lower.X = 0.0
model.ComputationalGrid.Lower.Y = 0.0
model.ComputationalGrid.Lower.Z = 0.0

#Declare the geometries that you will use for the problem
model.GeomInput.Names = "solid_input"

#Define the solid_input geometry.
model.GeomInput.solid_input.InputType = "SolidFile"
model.GeomInput.solid_input.GeomNames = "domain"
model.GeomInput.solid_input.FileName = "LW.pfsol"

#First set the name for your `Domain` and setup the patches
for this domain
model.Domain.GeomName = "domain"
model.Geom.domain.Patches = "top bottom side"
```

Setting Resolution and domain extent

ParFlow has terrain following or orthogonal grid options



http://parflow.blogspot.com/2015/08/domain-options-terrain-following-vs.html

Terrain Following Grid EQ

Modified Darcy's Law:

$$\mathbf{q} = -\mathbf{K}_s(\mathbf{x})k_r(h)[\nabla(h+z)\cos\theta_x + \sin\theta_x]$$

Slopes and fluxes:

$$\theta_x = \tan^{-1}(S_{0,x}) \text{ and } \theta_y = \tan^{-1}(S_{0,y})$$

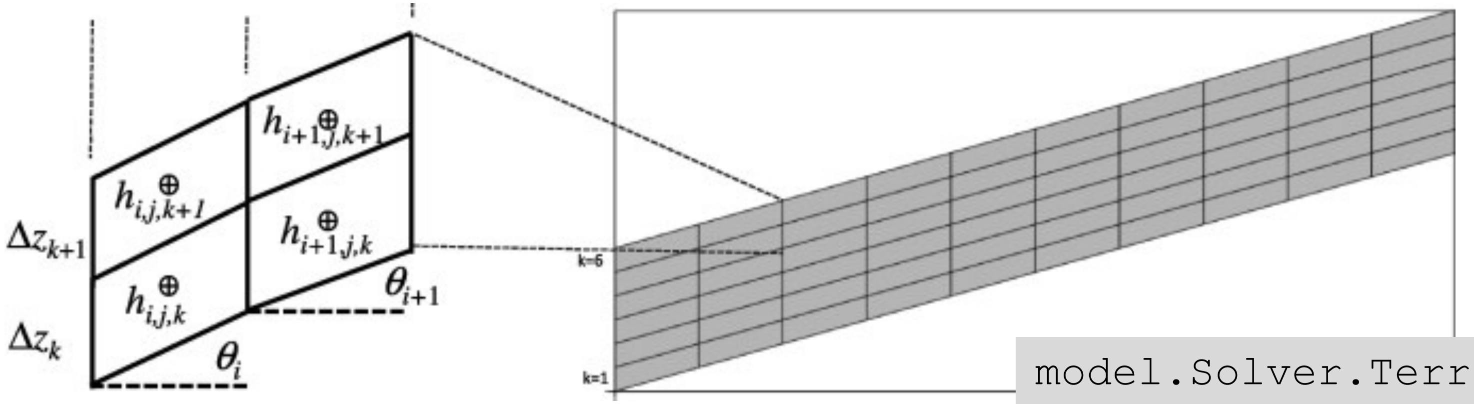
$$q_x = -K_{s,x}(\mathbf{x})k_r(h)\left[\frac{\partial(h)}{\partial x}\cos\theta_x + \sin\theta_x\right]$$

$$= -K_{s,x}(x)k_r(h)\frac{\partial(h)}{\partial x}\cos\theta_x - K_{s,x}(x)k_r(h)\sin\theta_x$$

Diffusive Pressure Term Topographic Term

Terrain Following Grid Configuration

– Transforms the grid to conform to topography using the slope files
– Generates a grid with a uniform thickness everywhere
– Can only be used with Solver Richards and not available with Solver Impes

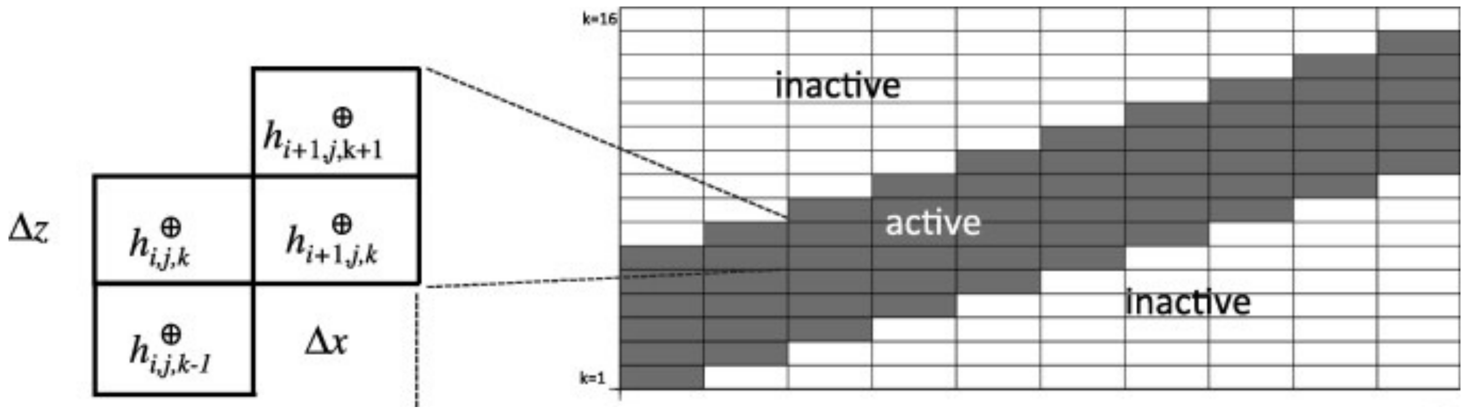



```
model.Solver.TerrainFollowingGrid = True
```

Maxwell, 2013

Orthogonal Grid configuration:

- Traditional grid
- Allows for irregular geologic layers and non-uniform watershed depths
- Requires a pfsol file to define active and inactive cells

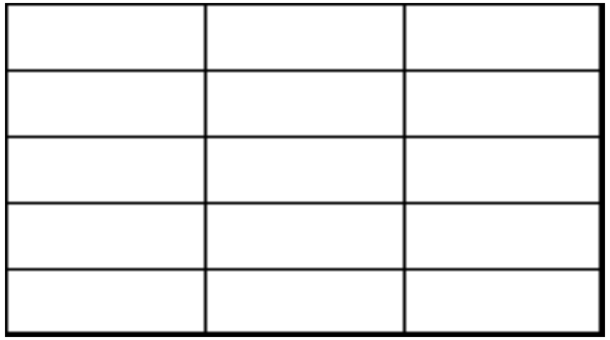


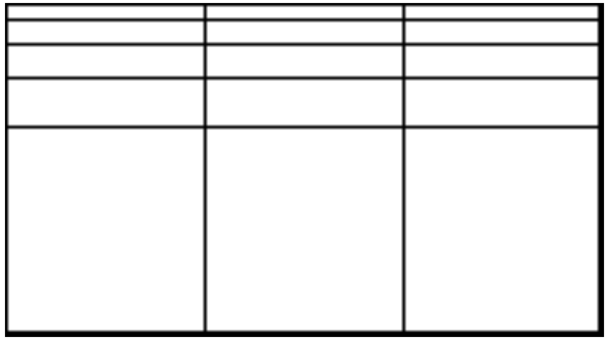

```
model.GeomInput.solid_input.InputType = "SolidFile"
model.GeomInput.solid_input.GeomNames = "domain"
model.GeomInput.solid_input.FileName = "LW.pfsol"
```

http://parflow.blogspot.com/2008/08/patch-order-and-solid-files.html

Constant or variable layer thickness options

- Constant or variable layer thickness (dz)





Variable dz allows for thin layers at the surface and thicker layers at the bottom of the domain to maintain high resolution in the upper layers and total domain thickness without too many layers

(See Manual [6.14 dZMultipliers])

Example Grid Configuration:

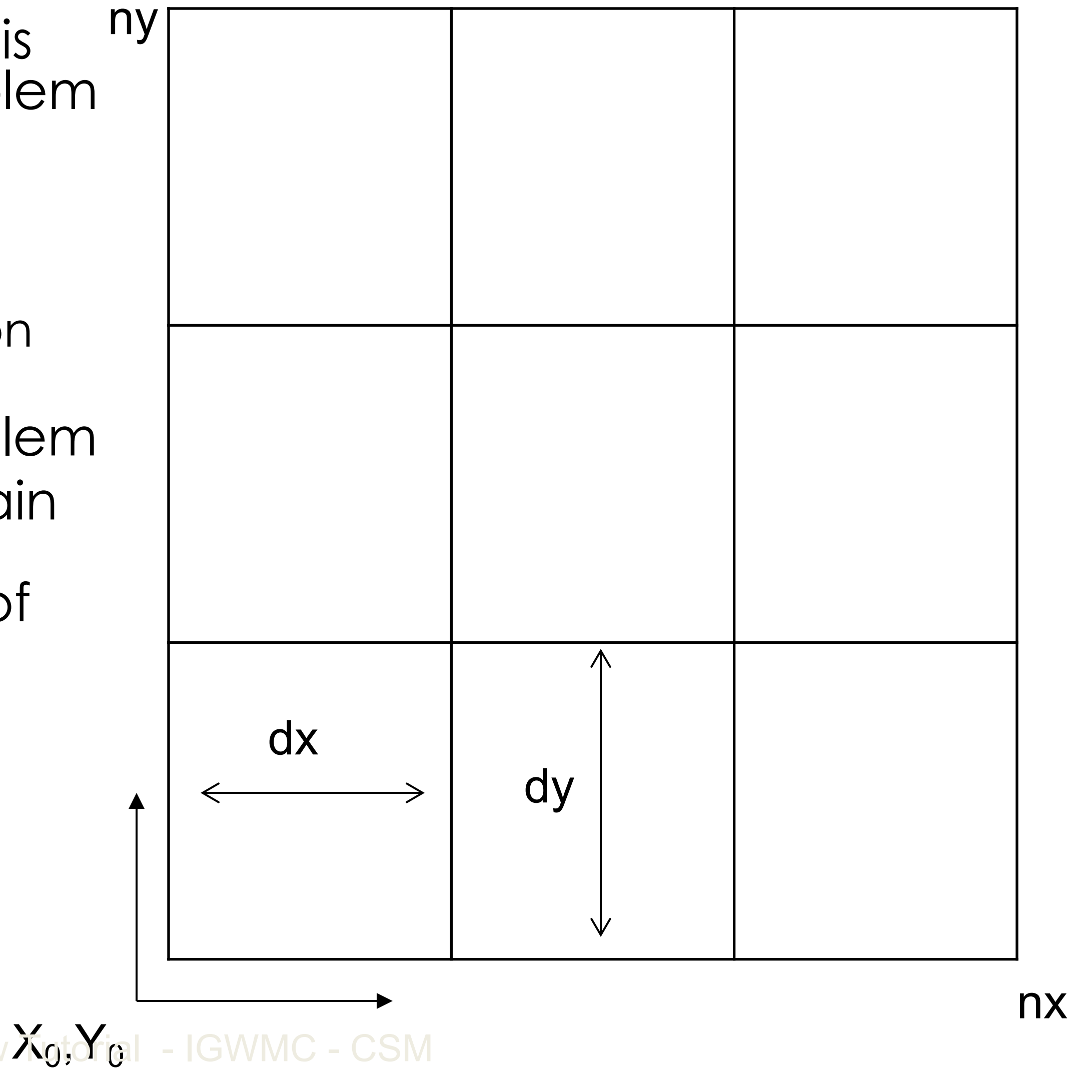
Variable dz example for a terrain following grid with solid file geometry

```
model.ComputationalGrid.NZ = 10
model.ComputationalGrid.DZ = 200.0

model.Solver.Nonlinear.VariableDz = True
model.dzScale.GeomNames = "domain"
model.dzScale.Type = "nzList"
model.dzScale.nzListNumber = 10

# 10 layers, starts at 0 for the bottom to 9 at the top
model.Cell._0.dzScale.Value = 5
model.Cell._1.dzScale.Value = 0.5
model.Cell._2.dzScale.Value = 0.25
model.Cell._3.dzScale.Value = 0.125
model.Cell._4.dzScale.Value = 0.05
model.Cell._5.dzScale.Value = 0.025
model.Cell._6.dzScale.Value = 0.005
model.Cell._7.dzScale.Value = 0.003
model.Cell._8.dzScale.Value = 0.0015
model.Cell._9.dzScale.Value = 0.0005
```

Note the z-upper is synched to computational grid, and is not linked with the Z multipliers

Computational Grid (§ 6.1.3)

- computational grid is a box that is the global outer shell of the problem
- it is defined by:
  - a lower x,y,z coordinate
  - cell diminsions (dx,dy,dz)
  - number of cells in each dimension (nx,ny,nz)
- grid spacing is uniform over problem
- though cubic the problem domain which defines the actual, active computational domain can be of any shape
- Code is cell-centered



Computational Grid (Input File)

Comment character for tcl/tk

```
#-----------------------------------------------------------------
#  Computational Grid
#-----------------------------------------------------------------
model.ComputationalGrid.Lower.X    =        0.0
Model.ComputationalGrid.Lower.Y    =        0.0
model.ComputationalGrid.Lower.Z    =        0.0

model.ComputationalGrid.NX      =          30
model.ComputationalGrid.NY      =          30
model.ComputationalGrid.NZ      =          30

model.ComputationalGrid.DX    =     10.0
model.ComputationalGrid.DY    =     10.0
model.ComputationalGrid.DZ    =     0.05
```

Coordinates (length units)

Grid dimensions (integer)

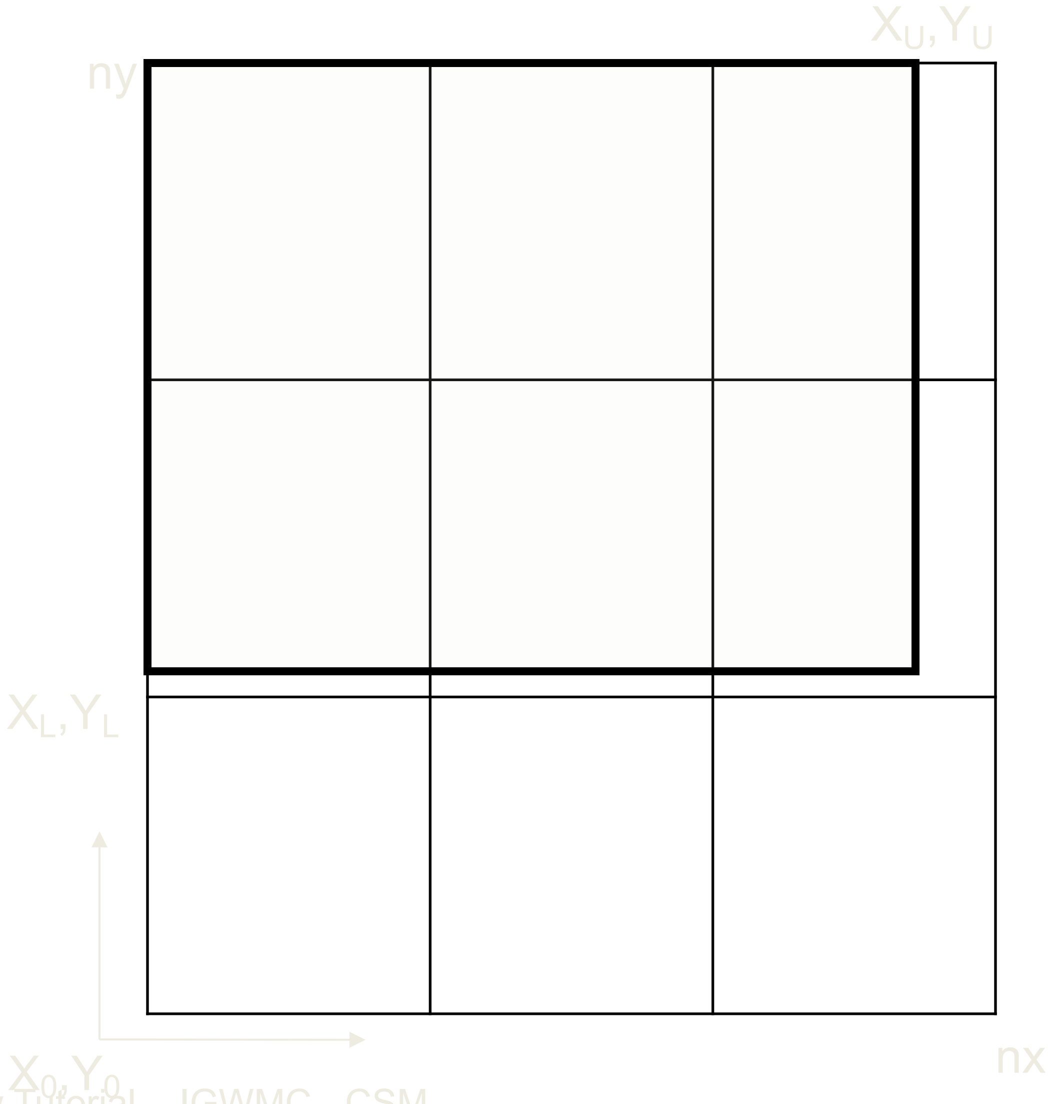
Cell size (length units)

Geometries (§ 6.1.4)

- Geometries are shapes that define aspects of the problem
- Any number is possible
- Combinations are fine
- Three types
  - Box
  - SolidFile
  - IndicatorField

Box Geometry

- a rectangular shape, specified within ParFlow input as upper and lower corner coordinates



Box Geometry (Input File)

```
#-----------------------------------------------------------
# The Names of the GeomInputs
#-----------------------------------------------------------
pfset GeomInput.Names                          "channelinput"

pfset GeomInput.channelinput.GeomName   channel

pfset GeomInput.channelinput.InputType   Box
#-----------------------------------------------------------
# Channel Geometry
#-----------------------------------------------------------
pfset Geom.channel.Lower.X                     140.0
pfset Geom.channel.Lower.Y                       0.0
pfset Geom.channel.Lower.Z                       0.0

pfset Geom.channel.Upper.X                      160.0
pfset Geom.channel.Upper.Y                      300.0
pfset Geom.channel.Upper.Z                        1.5
```

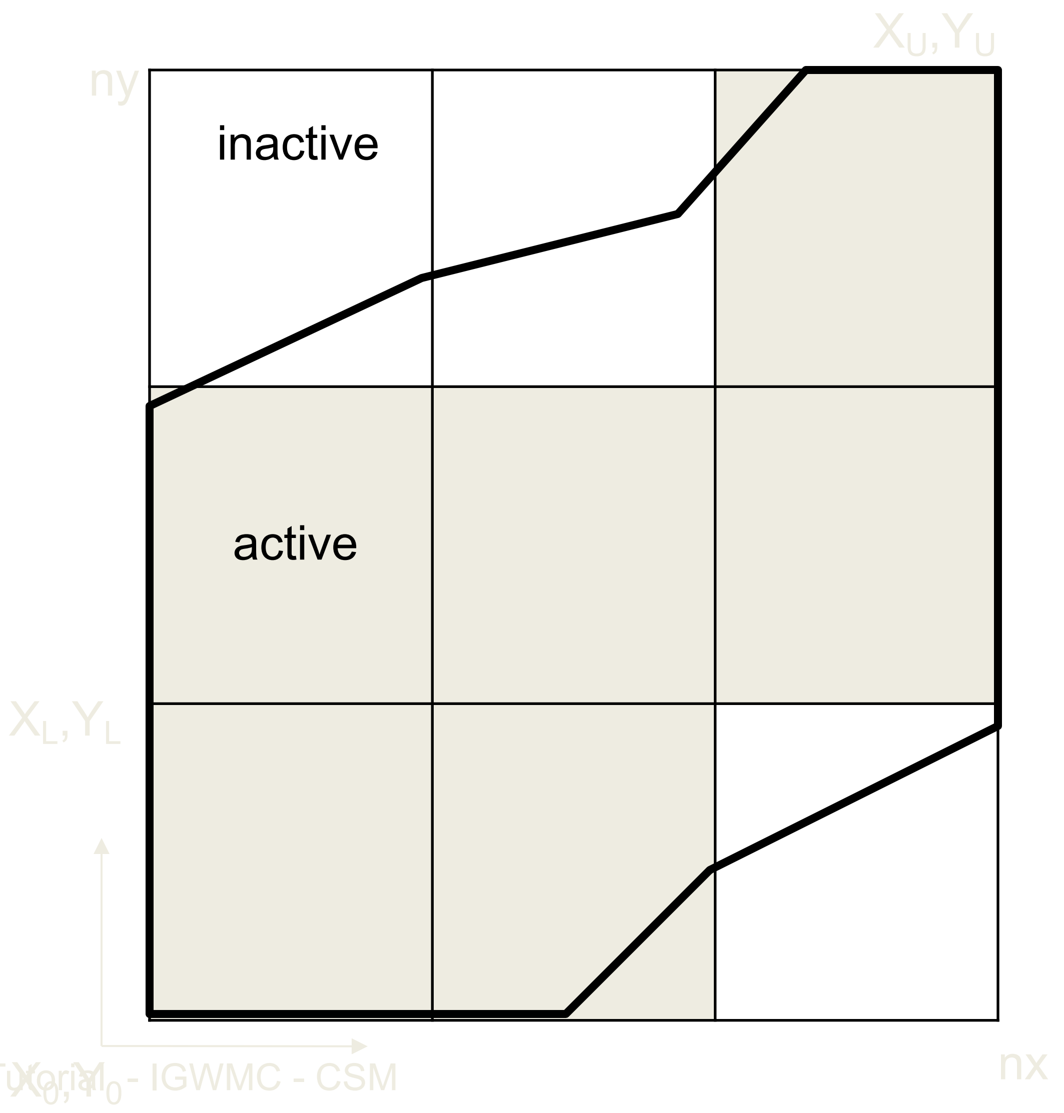
First define names for geometry inputs

Lower Coordinates (length units)

Upper Coordinates (length units)

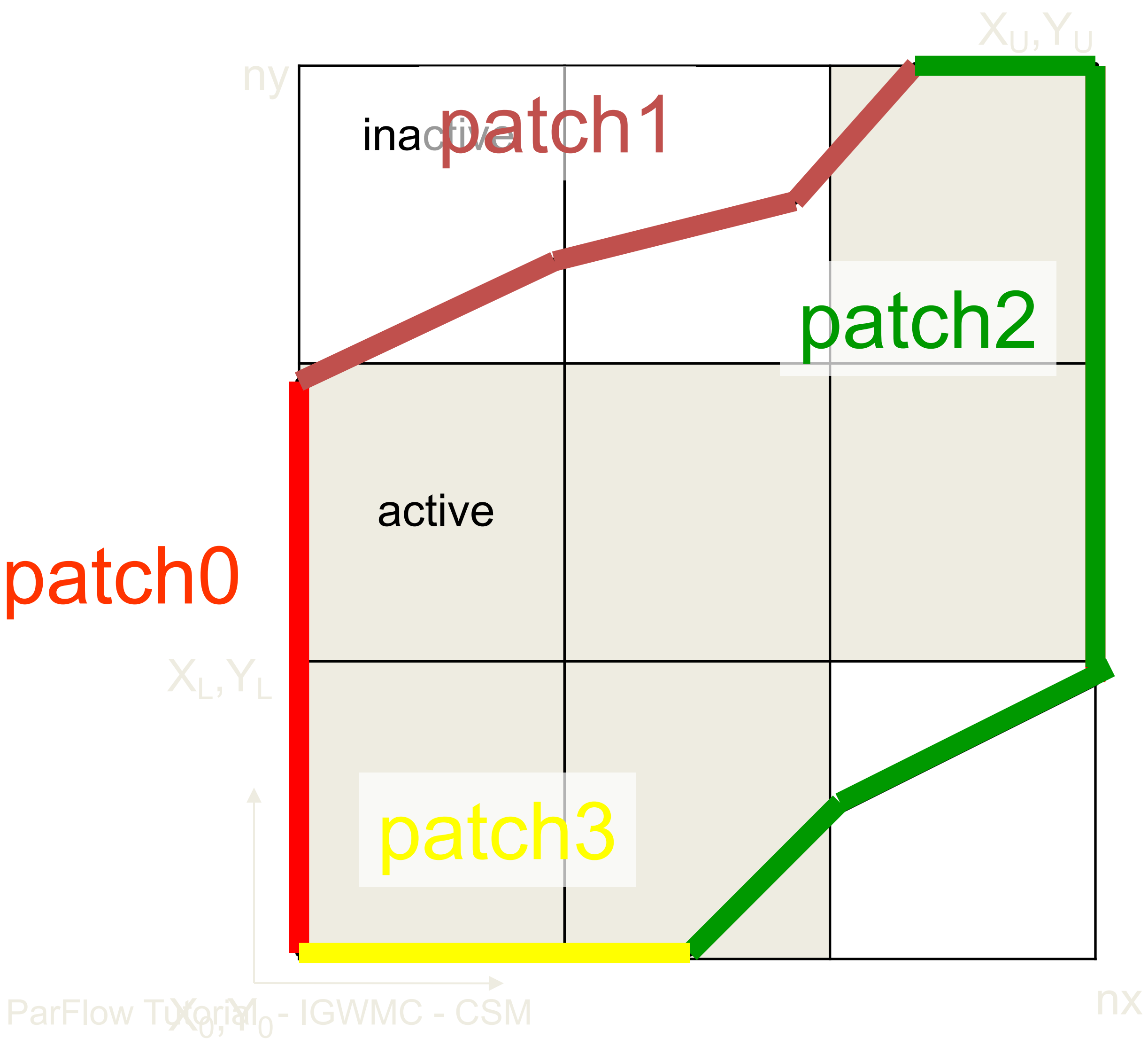
SolidFile Geometry

- A triangulated information network file that can delineate geometries of any shape
- Read in as a .pfsol file
- Geometries and patches are defined from within the file
- May be used to delineate active and inactive cells

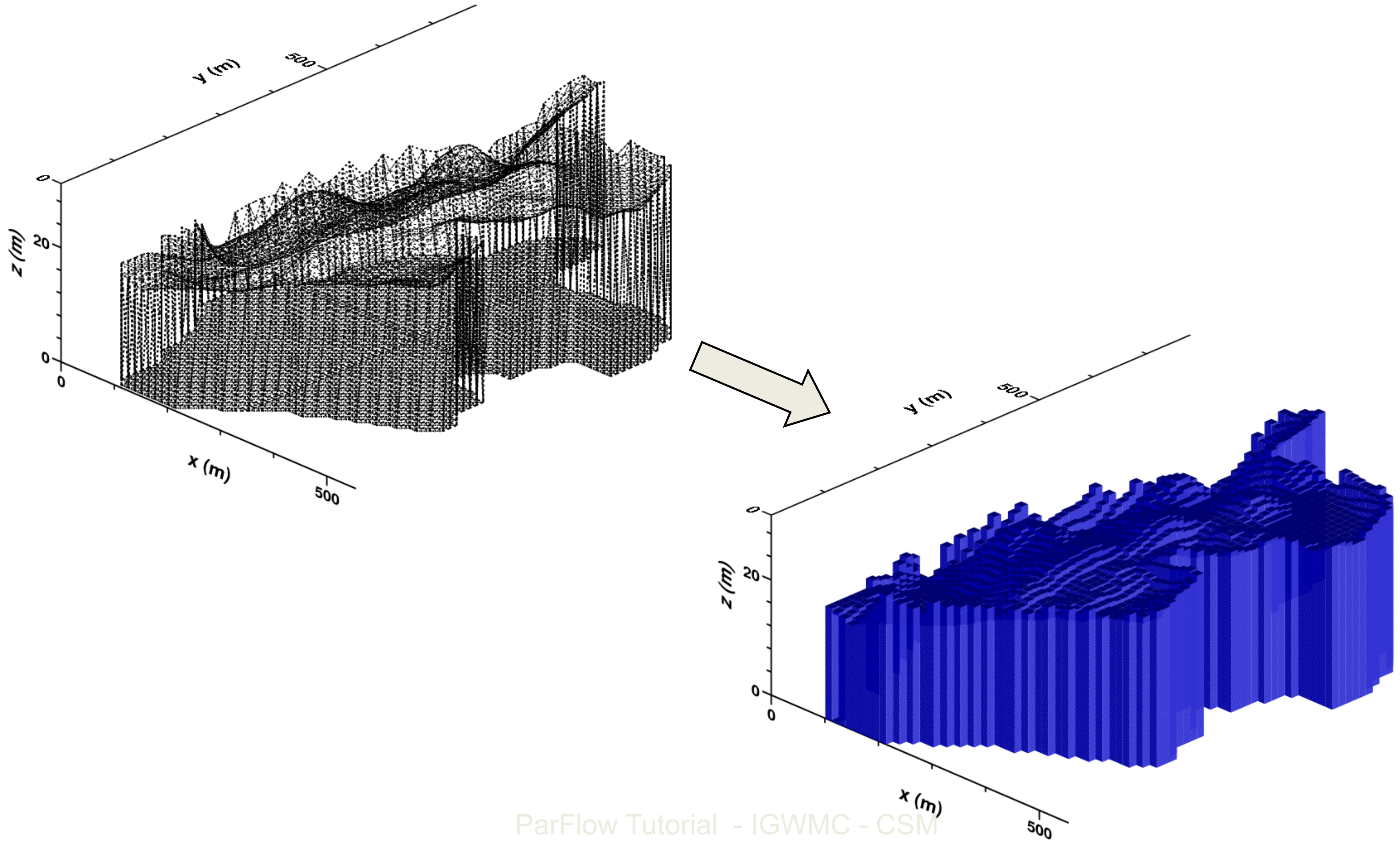


SolidFile Geometry- Patches

- patches can be any number or combination
- Must completely enclose geometry



SolidFile Geometry: Orthogonal grid

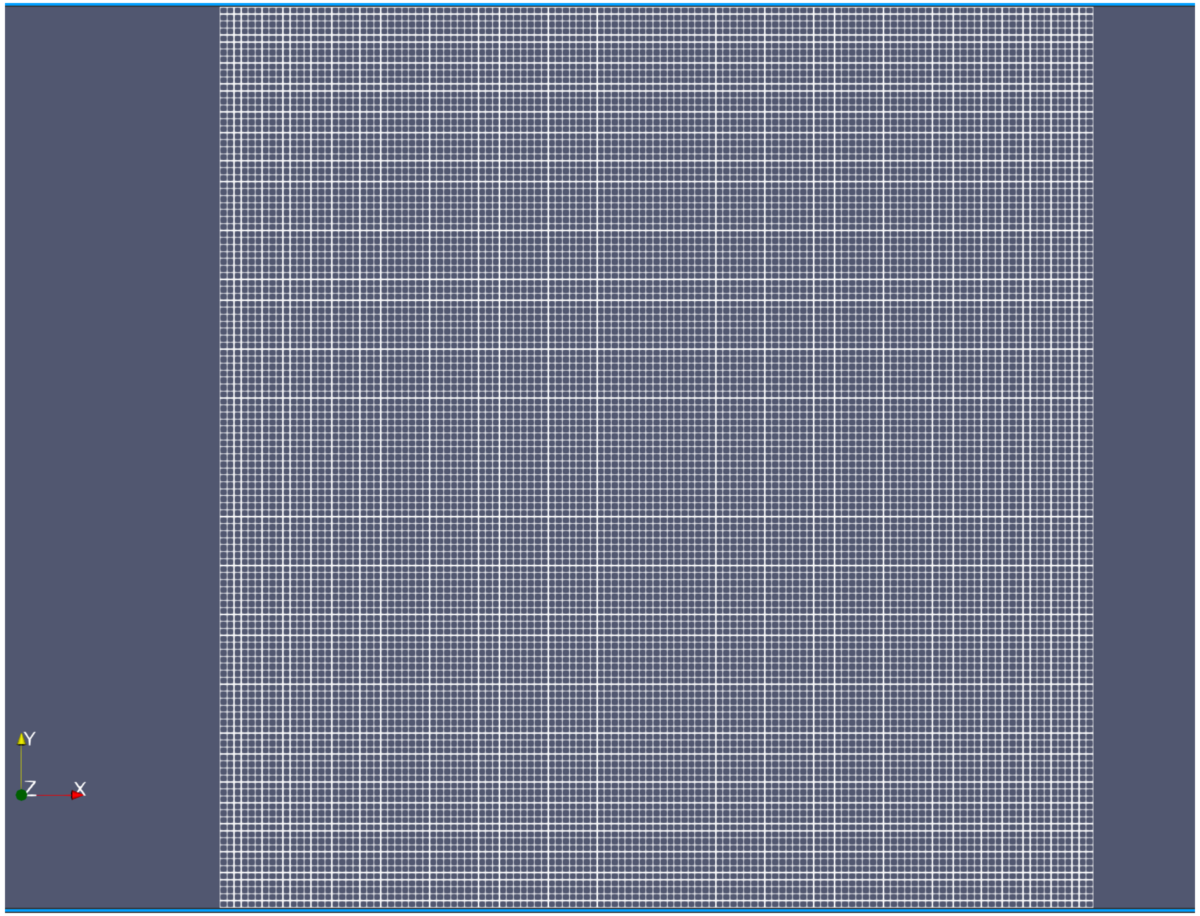


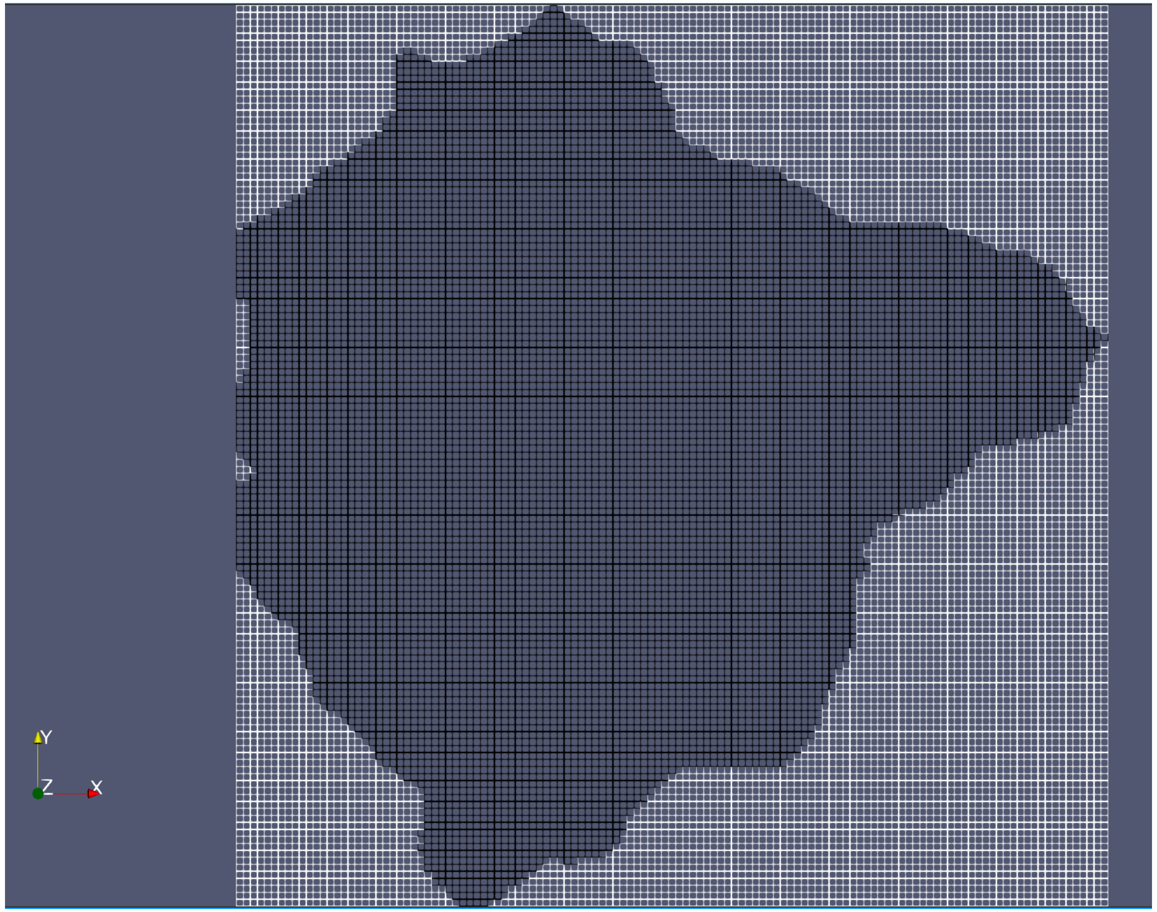
SolidFile Geometry (input file)

```
pfset GeomInput.Names                          "solidinput"

pfset GeomInput.solidinput.InputType  SolidFile
pfset GeomInput.solidinput.GeomNames  domain
pfset GeomInput.solidinput.FileName   fors2_hf.pfsol

pfset Geom.domain.Patches                      "infiltration z-upper
     x-lower y-lower x-upper y-upper z-lower"
```

Solid file: Real domain

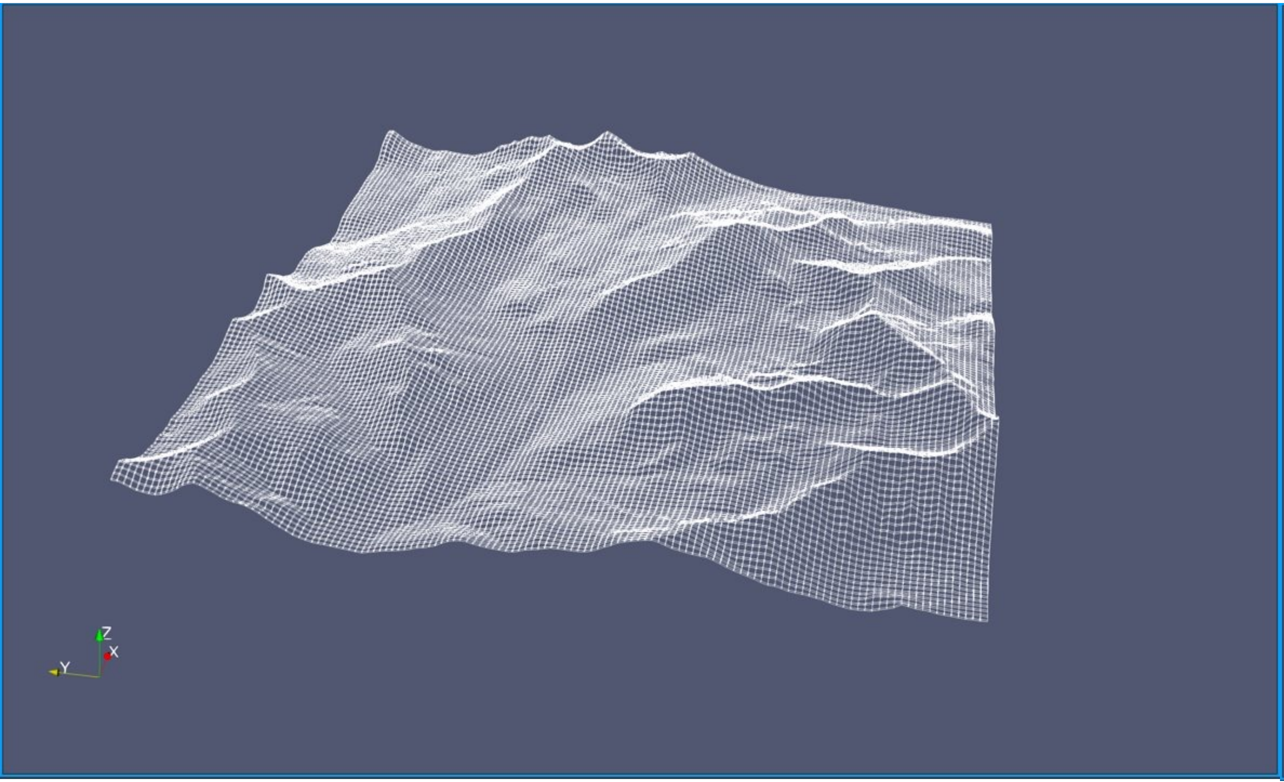
- Overhead view (grid only)



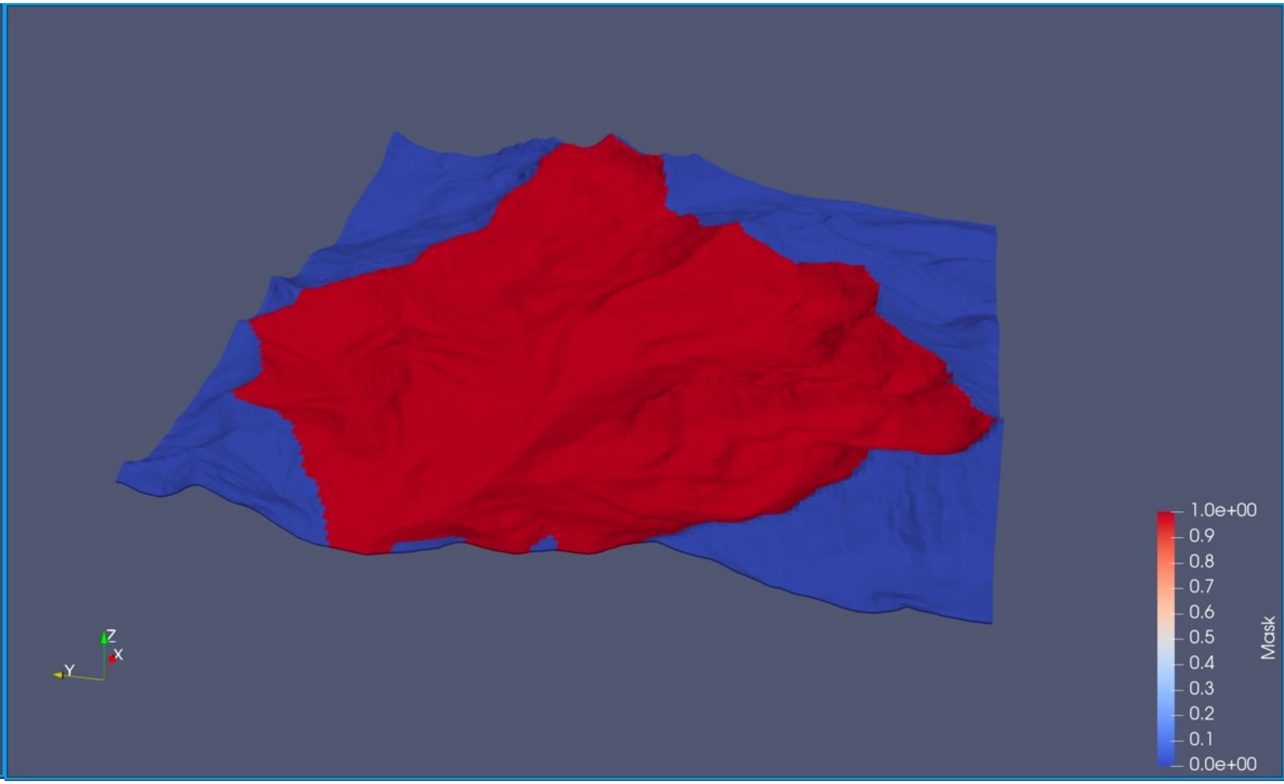


Solid file: Terrain following

- Oblique view

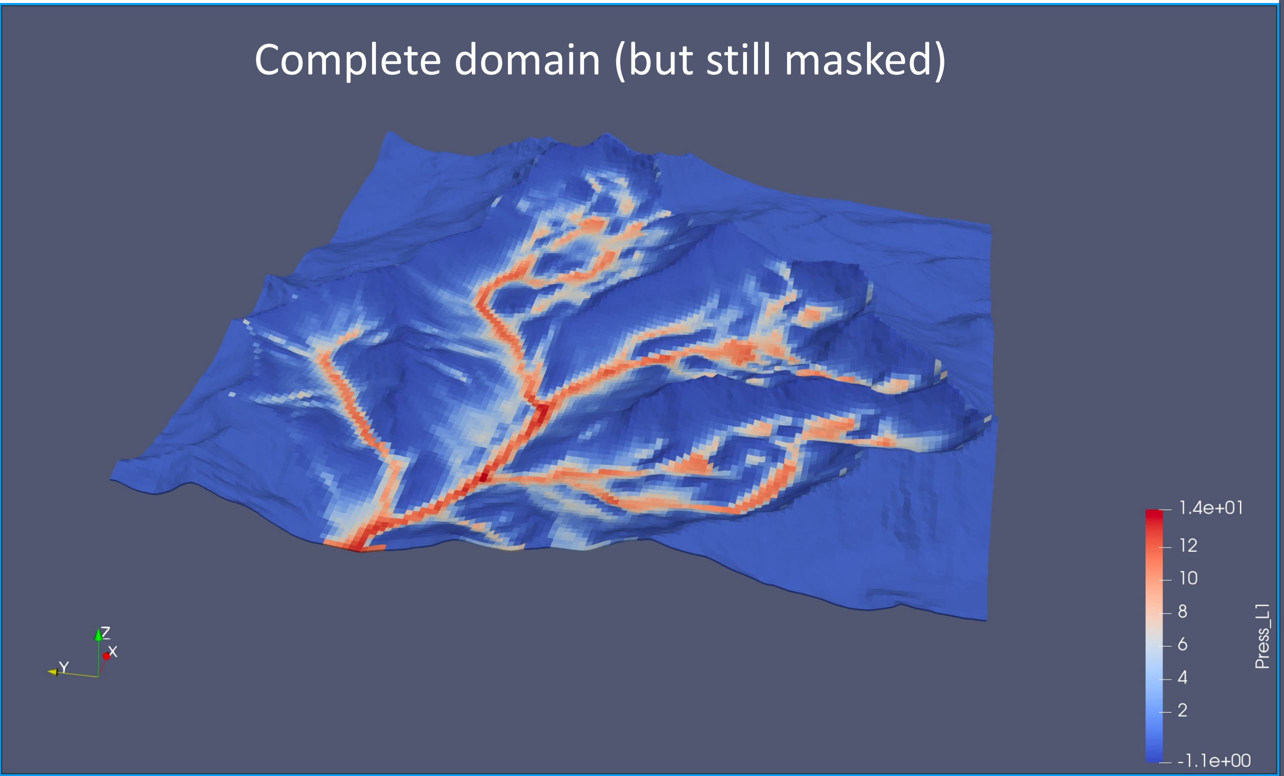


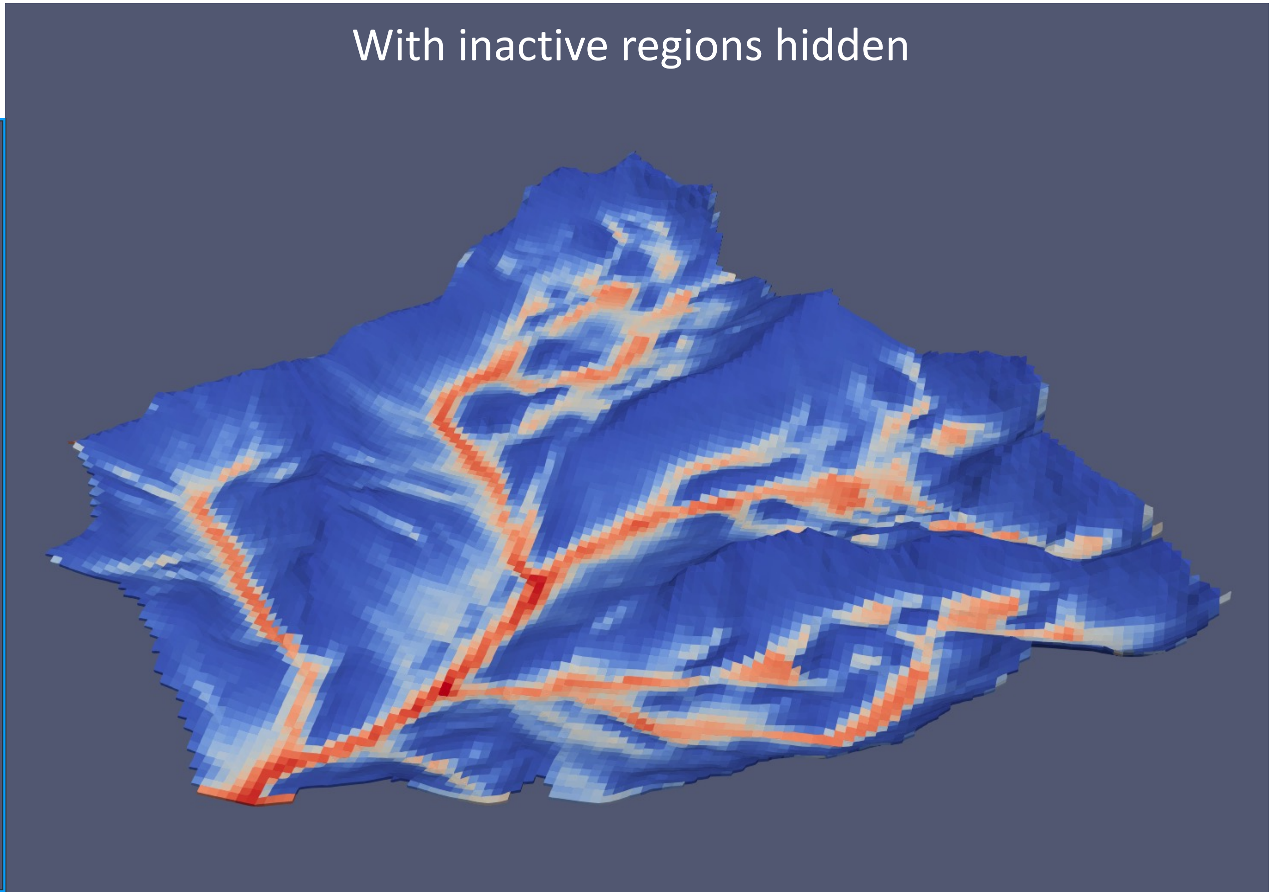
Grid only



Active cells – Red, Inactive - Blue

- Reduces number of active cells
- Example pressure field
  - Details in Engdahl (2024) https://doi.org/10.1029/2023WR035975





Temporal gridding: Time stepping

- A few preliminaries:
  - Convergence of the solver
    - Globally implicit means unconditional stability not accuracy
  - Time step can impact solution accuracy and runtime
  - Every model will be different
    - Some testing is necessary to understand how it will behave

Timing (§ 6.1.5)

- Timing only used for solver Richards
- Time steps may be constant or variable
- Timing section provides link between time steps and time cycles (next)
- Time units set by k, K units as described previously

Timing (§ 6.1.5)

```
#-----------------------------------------------------------
# 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)

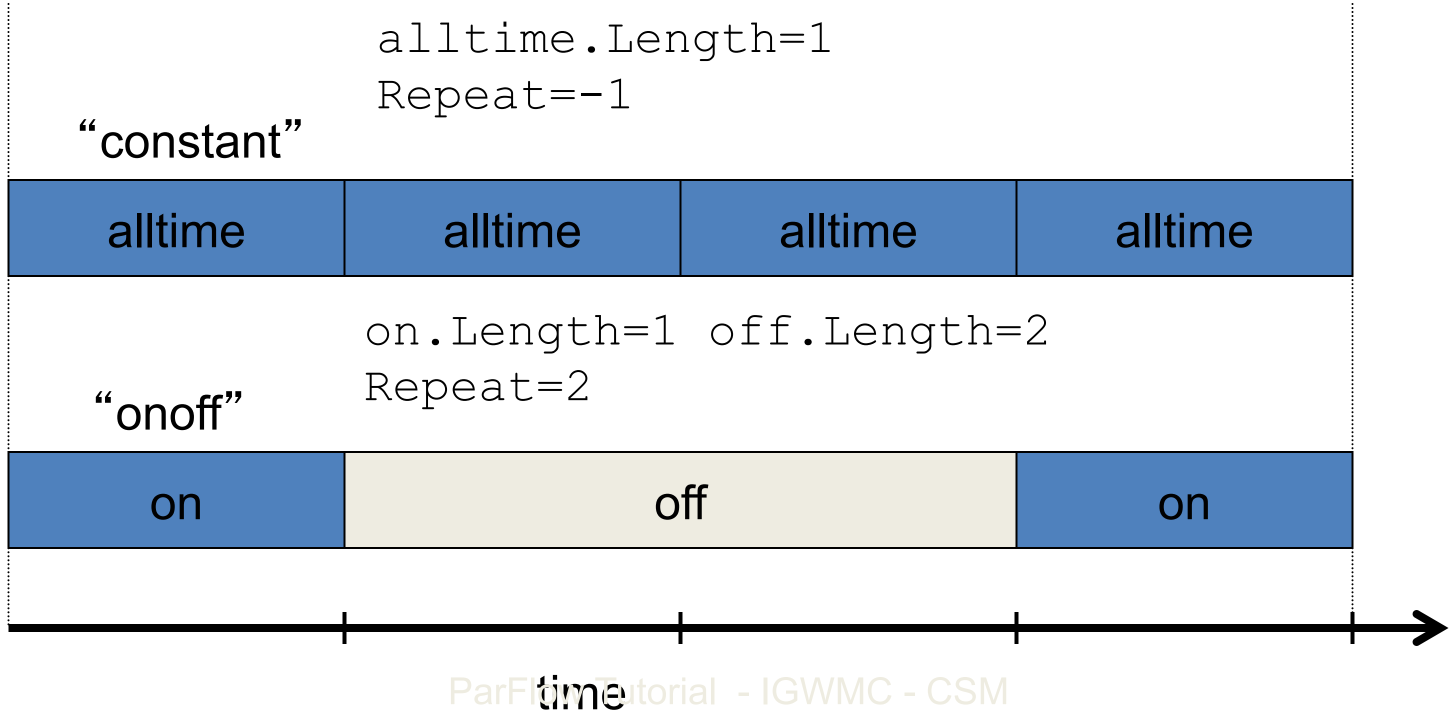
Interval to write output (T)
-1 outputs at every timestep

Timestep type

ΔT (T)

Time Cycles (§ 6.1.6)

- Time cycles are named lists
- All cycles are *integer multipliers* of `BaseUnit` value defined previously
- May be used for BC's and wells.



Time Cycles (§ 6.1.6)

```
#-----------------------------------------------------------
# Time Cycles
#-----------------------------------------------------------
pfset Cycle.Names                    "constant onoff"

pfset Cycle.constant.Names "alltime"
pfset Cycle.constant.alltime.Length          1
pfset Cycle.constant.Repeat                 -1


pfset Cycle.onoff.Names              "on off"
pfset Cycle.onoff.on.Length                 1
pfset Cycle.onoff.off.Length                2
pfset Cycle.onoff.Repeat                    2
```

Length of time cycle and repeat value

Length of each time cycle and repeat value

Tradeoffs to consider

- "Big time" steps will take longer to converge
  - Big is relative to the "complexity" of the model
    - i.e., runoff generation is slower than subsurface

- Small ones = You'll need more of them
  - But will also be more accurate

- Takes some experience and testing to find the balance
  - Watch the kinsol log