

Watershed Simulations with ParFlow CLM

Workflow overview, topographic processing, spin up and trouble shooting

ParFlow Short Course

Module 8

Learning Objectives:

At the end of this module students will understand:

- The inputs that are required to setup a watershed model
- How to set keys for a ParFlow model
- Where to find example workflows to start from
- A typical modeling workflow - parkinglot test, spinup, transient spinup, transient run
- The outputs that are generated and how to view them
- What is meant by 'spinup'
- Why it is an important to do
- What is the difference between a steady state solution and dynamic equilibrium.
- How to tell when a model spinup is complete.
- ParFlow Keys that can be used during spinup to speed up the process
- Some example approaches to model spinup

Steps for building a watershed model in ParFlow

1. Gridded Inputs

Meteorology*

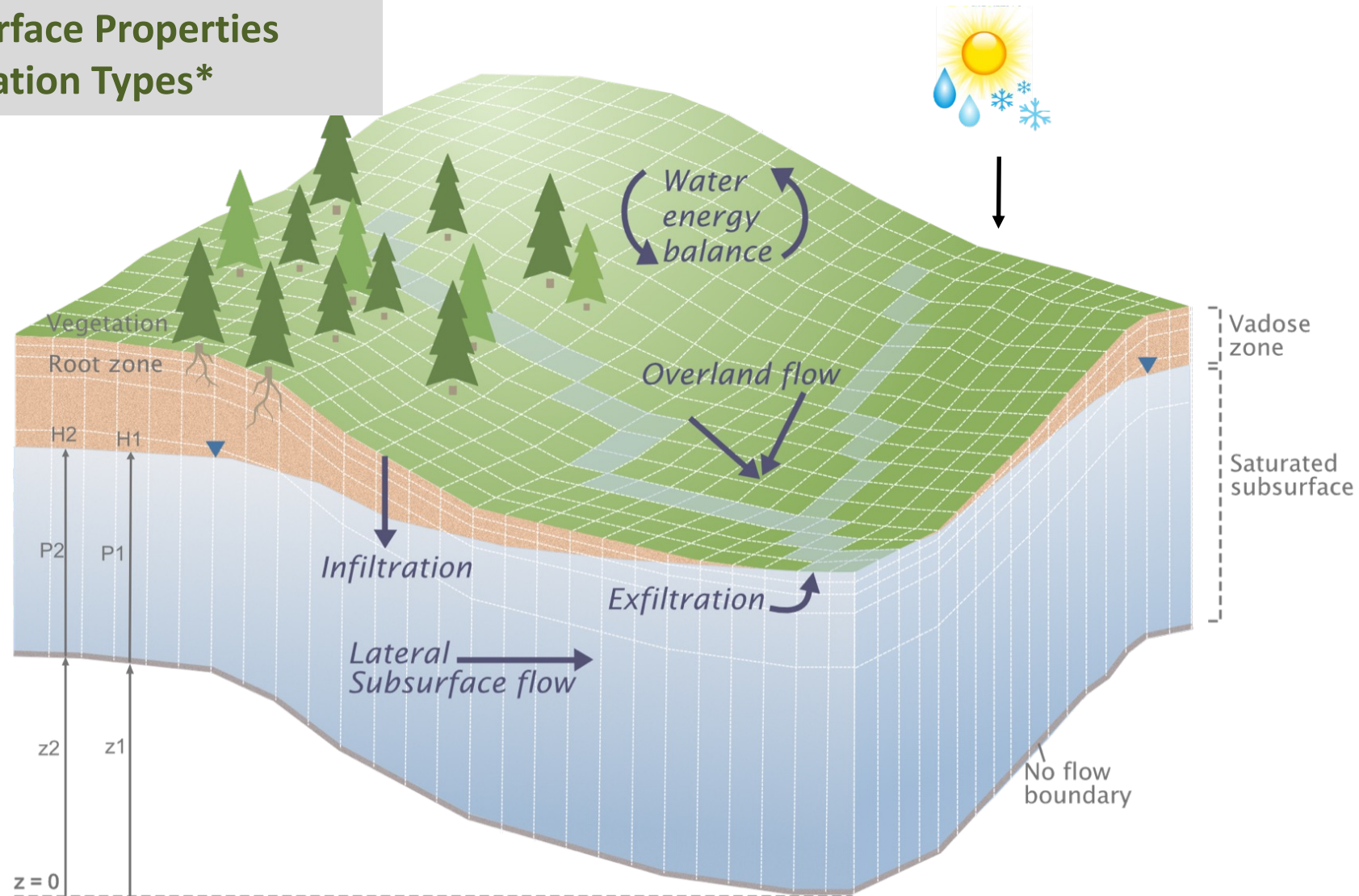
Topography

Subsurface Properties

Vegetation Types*

2. Boundary Conditions

3. Initial Conditions



Workflow Outline

1. Evaluate available model inputs
2. Determine your domain configuration
3. Process topography
4. Setup the subsurface
5. Initialize the model (i.e. spinup)
6. Additional setup for PF-CLM

This is also outlined in [Setting Up a Real Domain](#) in the manual

1. Evaluate available model inputs

- Land surface
 - Digital Elevation Model -> slopes
 - Mannings roughness coefficients
- Subsurface configuration: hydrologic properties for soil and and deeper geologic units
 - Permeability
 - Porosity
 - Specific storage
 - Relative permeability
 - Saturation

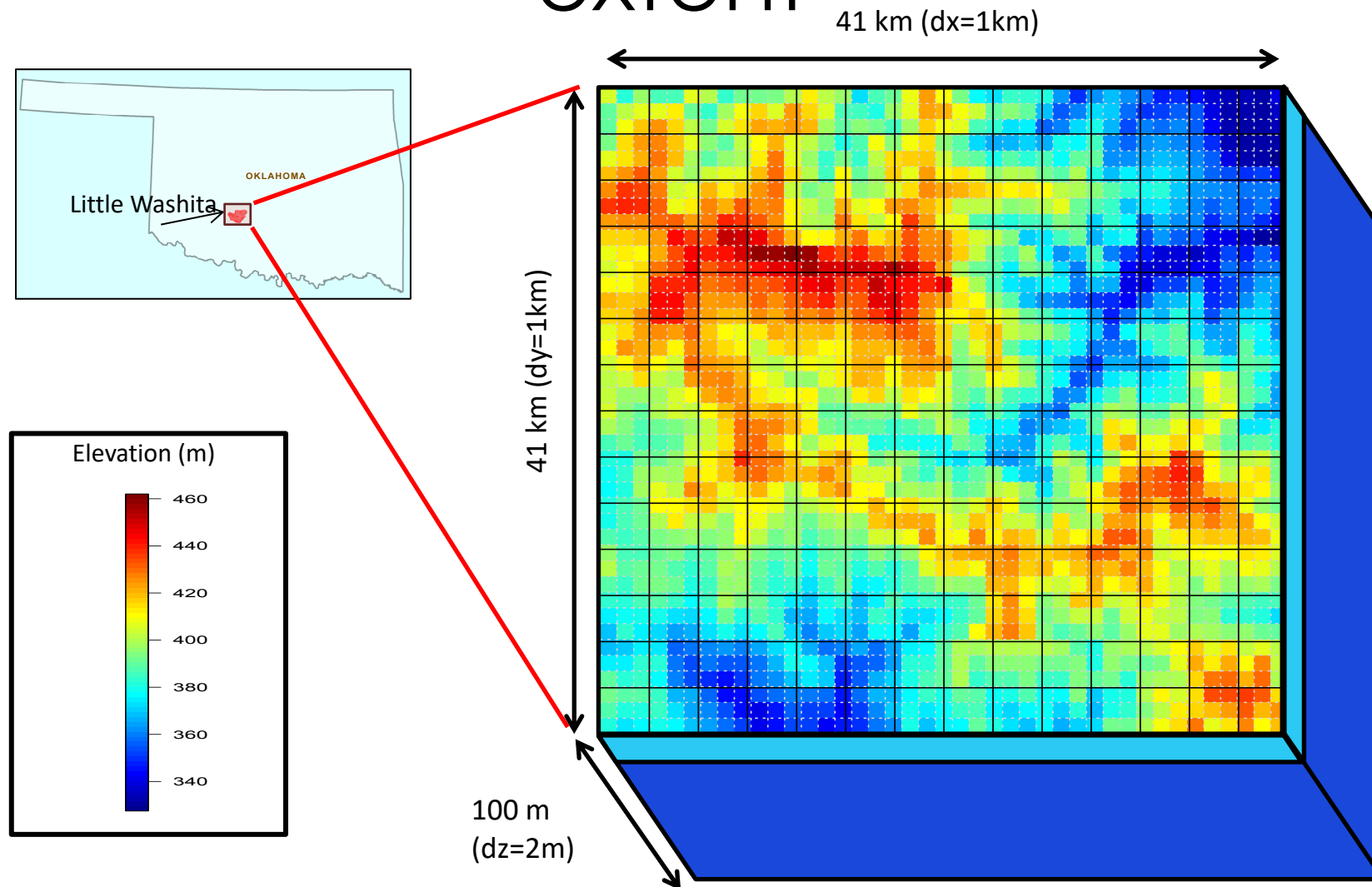
1. Evaluate available model inputs

- Atmospheric forcings:
 - ParFlow: moisture flux for surface
 - ParFlow CLM: precipitation, air temperature, wind, incoming short and longwave radiation, atmospheric pressure, specific humidity
- Land cover: vegetation types and properties (for ParFlow-CLM)

2. Determine your domain configuration

1. *What are the questions you want to answer with your model?*
2. *What kind of inputs do you have available to build your model with?*

Lateral resolution and domain extent



Setting up the subsurface model

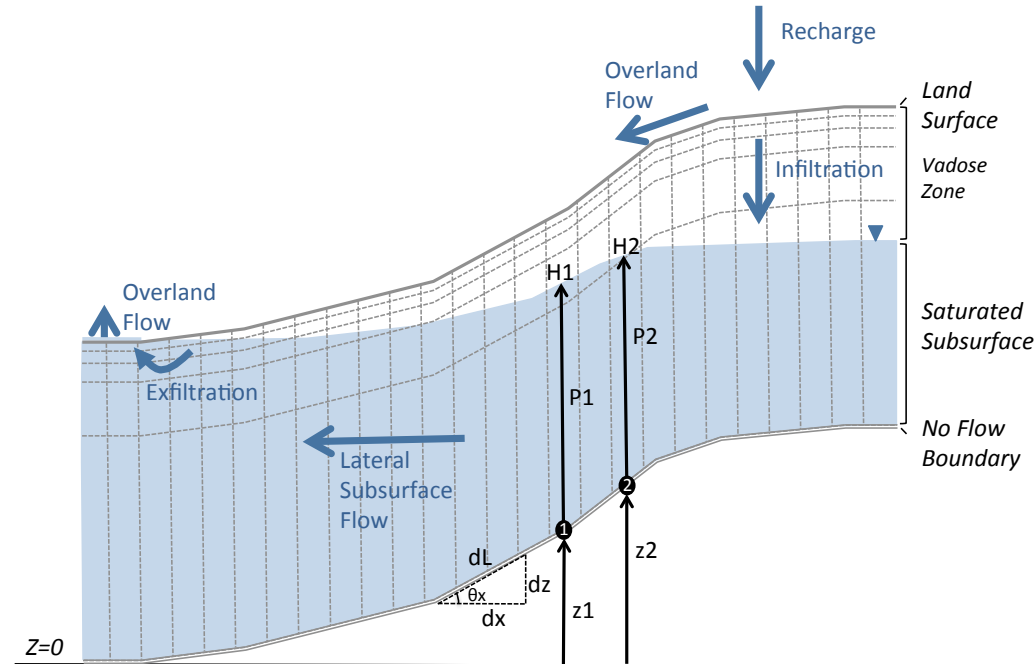
- Computational Grid (variable dz?)
- Geometries
- Domain
- Parameters
 - Permeability
 - Porosity
 - Specific storage
 - Relative permeability
 - Saturation
 - Toposlopes
 - Mannings coefficient
- Timing
 - Time steps
 - Time cycles
- Boundary Conditions
- Initial Conditions
- CLM (yes/no)
- Terrain following grid (yes/no)
- Solver settings

3. Processing Topography

Convert from a digital elevation model to slopes and adjust values to ensure a realistic drainage network

Processing Topography

- ParFlow requires slopes not elevations




Built in PFTools commands can be used to calculate slopes from elevations, but this is **not** the recommended approach for watershed domains.

Why shouldn't you use slopes from your raw DEM directly?

- When you calculate slopes from DEMs you are not guaranteed to get a fully connected drainage network
- This is particularly problematic for lower resolution DEMs


This is a well known problem and there are many approaches to address this



Contents lists available at [ScienceDirect](#)

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo



Routing overland flow through sinks and flats in interpolated raster terrain surfaces ☆

Frank Kenny *, Bryce Matthews, Kent Todd

Water Resources Information Program, Ontario Ministry of Natural Resources, 300 Water St., Peterborough, Ontario, Canada K9J 8M5

HYDROLOGICAL PROCESSES
Hydrol. Process. **30**, 846–857 (2016)
Published online 20 September 2015 in Wiley Online Library
(wileyonlinelibrary.com) DOI: 10.1002/hyp.10648

Efficient hybrid breaching-filling sink removal methods for flow path enforcement in digital elevation models

John B. Lindsay*

Department of Geography, The University of Guelph, 50 Stone Road East, Guelph N1G 2W1, Canada

EARTH SURFACE PROCESSES AND LANDFORMS
Earth Surf. Process. Landforms **41**, 658–668 (2016)
Copyright © 2015 John Wiley & Sons, Ltd.
Published online 13 January 2016 in Wiley Online Library
(wileyonlinelibrary.com) DOI: 10.1002/esp.3888


The practice of DEM stream burning revisited

John B. Lindsay*

Department of Geography, University of Guelph, Guelph, Canada

Received 27 July 2015; Revised 7 December 2015; Accepted 7 December 2015


*Correspondence to: john.lindsay@uoguelph.ca



Contents lists available at [ScienceDirect](#)

Computers & Geosciences


journal homepage: www.elsevier.com/locate/cageo



Priority-flood: An optimal depression-filling and watershed-labeling algorithm for digital elevation models

Richard Barnes ^{a,*}, Clarence Lehman ^b, David Mulla ^c

*^aEcology, Evolution, & Behavior, University of Minnesota, USA
^bCollege of Biological Sciences, University of Minnesota, USA
^cSoil, Water, & Atmosphere Sciences, University of Minnesota, USA*



Hydrology and Earth System Sciences

Hydrol. Earth Syst. Sci., **14**, 1153–1165, 2010
www.hydrol-earth-syst-sci.net/14/1153/2010/
doi:10.5194/hess-14-1153-2010
© Author(s) 2010. CC Attribution 3.0 License.

On the uncertainty of stream networks derived from elevation data: the error propagation approach

T. Hengl¹, G. B. M. Heuvelink², and E. E. van Loon¹

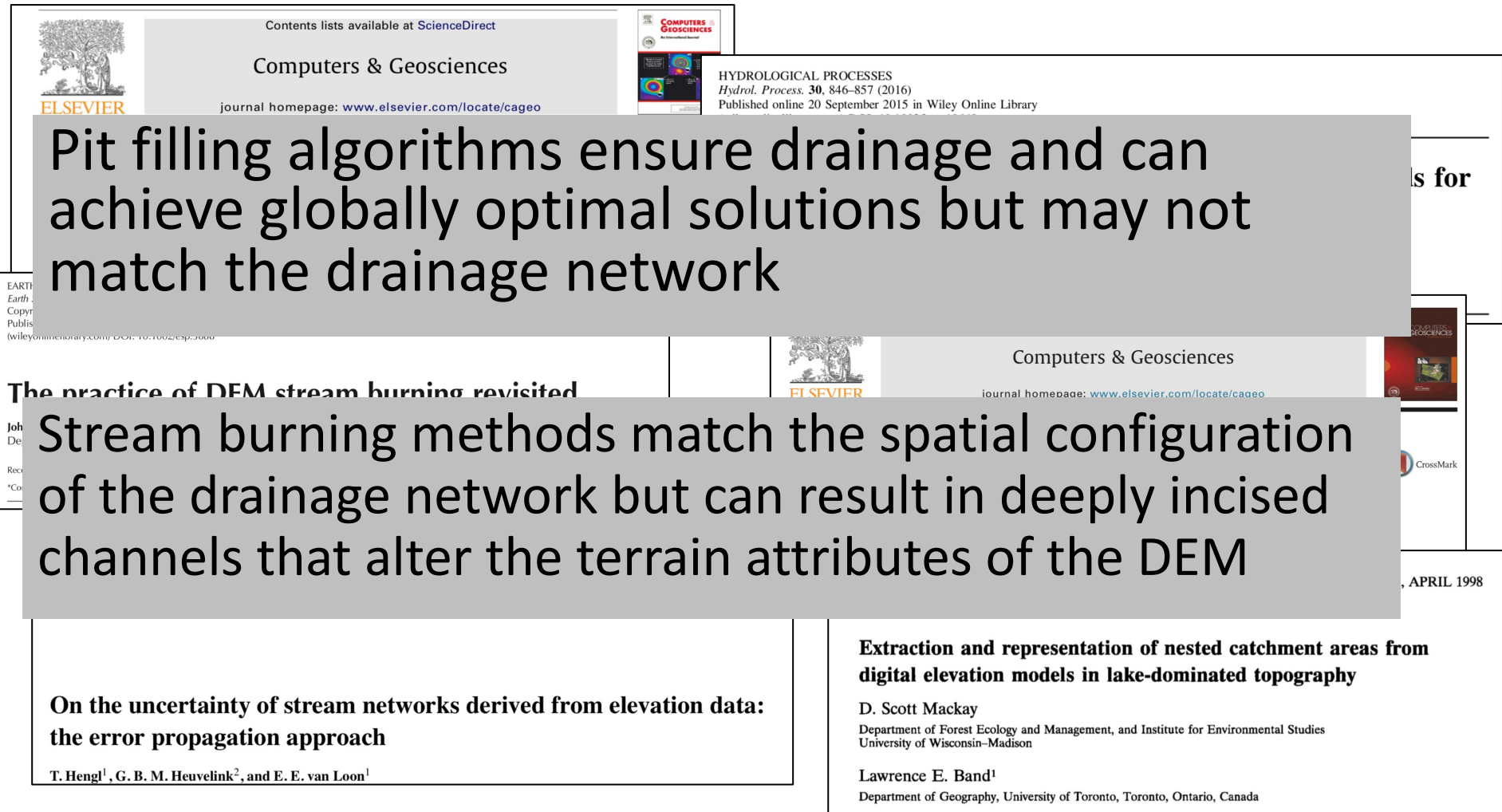
WATER RESOURCES RESEARCH, VOL. 34, NO. 4, PAGES 897–901, APRIL 1998

Extraction and representation of nested catchment areas from digital elevation models in lake-dominated topography

D. Scott Mackay
*Department of Forest Ecology and Management, and Institute for Environmental Studies
University of Wisconsin–Madison*

Lawrence E. Band¹
Department of Geography, University of Toronto, Toronto, Ontario, Canada

This is a well known problem and there are many approaches to address this



Pit filling algorithms ensure drainage and can achieve globally optimal solutions but may not match the drainage network

Stream burning methods match the spatial configuration of the drainage network but can result in deeply incised channels that alter the terrain attributes of the DEM

On the uncertainty of stream networks derived from elevation data: the error propagation approach
T. Hengl¹, G. B. M. Heuvelink², and E. E. van Loon¹

Extraction and representation of nested catchment areas from digital elevation models in lake-dominated topography
D. Scott Mackay
Department of Forest Ecology and Management, and Institute for Environmental Studies
University of Wisconsin–Madison
Lawrence E. Band¹
Department of Geography, University of Toronto, Toronto, Ontario, Canada

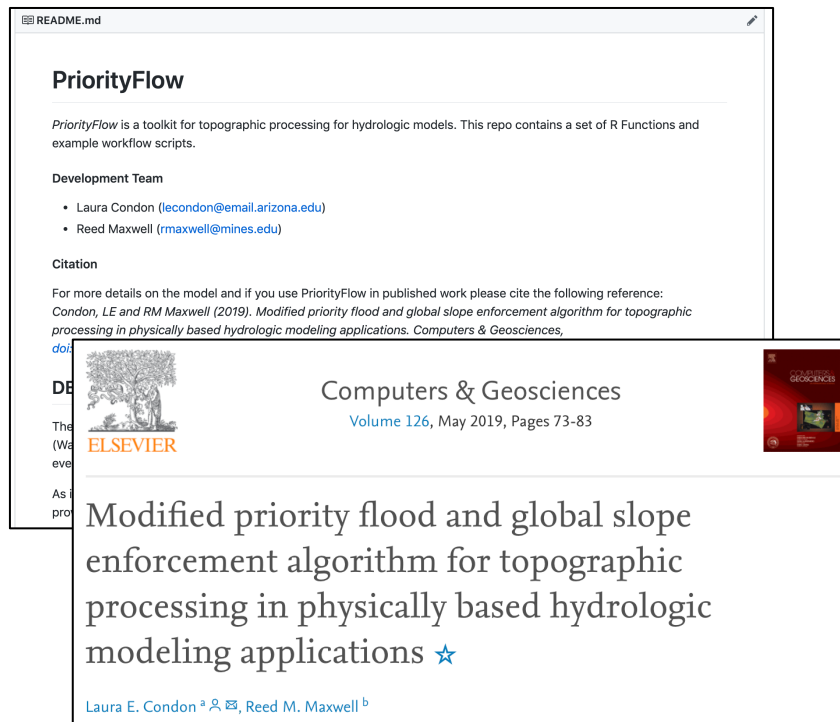
Processing Topography: GRASS Workflow

- GIS tools can help you evaluate your drainage network
- Example workflow using the watershed analysis tool in grass is documented on the ParFlow Blog (<http://parflow.blogspot.com/2015/08/terrain-processing.html>)
- This uses the A* algorithm to ensure that every cell in the domain drains out
- Make sure to look closely at the drainage network developed by GRASS to ensure its consistent with what you expect.

Processing Topography: PriorityFlow Tool

GitHub Repo

<https://github.com/lecondon/PriorityFlow>



- Modified priority flood algorithm for depression filling and stream enforcement
- Slope processing tools and workflows for D4 and D8 routing
- Options for stream network smoothing
- Additional options for handling primary and secondary flow directions
- More control of DEM Processing and generated PF Consistent slope outputs

Processing topography

- There are built in tools to calculate slopes and do some processing
- Refer to the [Manual Section 8.2](#) for details
- For more advanced processing you can use:
 - Priority Flodd Tool (<https://github.com/lecondon/PriorityFlow>)
 - GIS tools (*See GRASS example on ParFlow blog, <http://parflow.blogspot.com/2015/08/terrain-processing.html>*)

```
set      dem      [pload -sa dem.format_header.txt]
pfsetgrid {41 41 1} {0.0 0.0 0.0} {1000 1000 1.0} $dem

# Fill flat areas (if any)
# this routine interpolates across the bounds of flat areas to ensure nonzero slopes)
set      flatfill  [pffillflats $dem]

# Pitfill
# (this routine uses a standard pit-fill method to remove local minima
set      pitfill   [pfpitfilldem $flatfill 0.01 10000]

# Slopes
# (uses 1st-order upwind differences, consistent with PF overland flow scheme)
set      slope_x   [pfslopes $pitfill]
set      slope_y   [pfslopes $pitfill]

# PFB (slopes only...needed as PFB for parflow input)
pfsave $slope_x -pfb klam.slope_x.pfb
pfsave $slope_y -pfb klam.slope_y.pfb
```

Parking lot test

- Make your domain impervious, rain on it and look at the drainage network



Parking lot test

1. Set the permeability very low

```
model.Geom.Perm.Names = "domain"  
model.Geom.domain.Perm.Type = "Constant"  
model.Geom.domain.Perm.Value = 0.000001
```

2. Rain on the domain intermittently

```
#setup the timing of the cycles  
model.Cycle.Names ="constant rainrec"  
  
model.Cycle.rainrec.Names = "rain rec"  
model.Cycle.rainrec.rain.Length = 5  
model.Cycle.rainrec.rec.Length = 20  
model.Cycle.rainrec.Repeat = -1  
  
#Setup the rainfall rate  
model.Patch.top.BCPressure.Type = "OverlandKinematic"  
model.Patch.top.BCPressure.Cycle = "rainrec"  
model.Patch.top.BCPressure.rain.Value = -0.05  
model.Patch.top.BCPressure.rec.Value = 0.0
```

Note that the rainfall value is negative to indicate a flux in the negative z direction and units are L/T

Parking lot test

1. Set the permeability very low

```
model.Geom.Perm.Names = "domain"  
model.Geom.domain.Perm.Type = "Constant"  
model.Geom.domain.Perm.Value = 0.000001
```

2. Apply constant rain

```
#setup a time cycle for length of the simulation  
model.Cycle.Names = "constant"  
  
model.Cycle.constant.Names = "alltime"  
model.Cycle.constant.alltime.Length = 1  
model.Cycle.constant.Repeat = -1  
  
#Setup the rainfall rate  
model.Patch.top.BCPressure.Cycle = "constant"  
model.Patch.top.BCPressure.constant.Value = -2.1E-05
```

What if it won't run?

1. Check that ParFlow is actually installed correctly and that you are able to run the test problems.
2. Look at the out.txt file to see if you are missing a key in your tcl script
3. Make sure that all of your input files are where they should be
4. If you are running on multiple processors make sure that every input file is being distributed and that the slopes files are distributed with NZ=1
5. Use PFTools to convert your inputs to silo and look to make sure they aren't corrupted and the dimensions are right

<http://parflow.blogspot.com/2015/08/troubleshooting-models-that-dont-run-at.html>

4. Setup the subsurface

Developing a gridded representation of subsurface

Subsurface inputs

1. Identify unique subsurface units and provide subsurface geometries to ParFlow
 - Either using solid files or indicator files
2. Assign the hydrologic properties to each unit
 - permeability, specific storage, porosity and van Genuchten parameters

<http://parflow.blogspot.com/2015/08/subsurface-setup-options.html>

5. Initializing the model (spinup)

Determining the starting groundwater configuration for your simulations

Spinup

- Goal: develop a domain that is stable and solving nicely before you start making runs to answer questions
- There is no best practice for spinup, results and approaches will vary depending on your domain and the questions you want to answer with your model
- Groundwater is the slowest moving part so its often easiest to start with a simplified system and get a stable water table before adding in land surface processes

<http://parflow.blogspot.com/2015/08/spinning-up-watershed-model.html>

One Approach to Spinup

1. Initialize your water table somewhere
 - Completely saturate the domain
 - Make it completely dry
 - Make an intelligent guess about where you think the water table should be
2. Run for a long time with a constant or periodic forcing at the land surface until you get a 'stable' configuration

One Approach to Spinup

1. Initialize your water table somewhere

You can set the water table to a constant depth like this:

```
model.ICPressure.Type = "HydroStaticPatch"  
model.ICPressure.GeomNames = "domain"  
model.Geom.domain.ICPressure.Value = -10.0  
model.Geom.domain.ICPressure.RefGeom = "domain"  
model.Geom.domain.ICPressure.RefPatch = "top"
```

or start your domain off completely dry like this:

```
model.ICPressure.Type = "HydroStaticPatch"  
model.ICPressure.GeomNames = "domain"  
model.Geom.domain.ICPressure.Value = 0.0  
model.Geom.domain.ICPressure.RefGeom = "domain"  
model.Geom.domain.ICPressure.RefPatch = "bottom"
```

or you can read in an input file with pressure heads like this:

```
model.ICPressure.Type = "PFBFile"  
model.ICPressure.GeomNames = "domain"  
model.Geom.domain.ICPressure.RefPatch = "top"  
model.Geom.domain.ICPressure.FileName = "press.init.pfb"
```

One Approach to Spinup

2. Run for a long time with a constant recharge forcing and no surface water flow
- You can set a spatially variable flux using a pfb file like this or you can set a homogenous flux using the options shown for the parking lot test

```
model.Solver.EvapTransFile = True  
model.Solver.EvapTrans.FileName = "PmE.flux.pfb"  
model.dist("PmE.flux.pfb")
```

- The units of this flux should be $1/T$ so if you have a flux that is L/T remember to divide by the thickness of each layer
- This is a 3D file so if you just want a flux applied to the top of the domain set the values for all other layers to zero

Spinup: One approach

2. Use seepage face as upper boundary (this will drain the domain more efficiently without overland flow)

```
Model.patch.top.BCPressure.Type "SeepageFace"
```

- This key turns on seepage face i.e. a drainage across boundary with $p = 0$.
- This drains the domain more efficiently and does not calculate overland flow.
- After equilibration with SeepageFace, turn OverlandFlow, OverlandKinematic or OverlandDiffusive on to include overland flow.

How do I know if I'm done?

Spinup Analysis

Look at your outputs and check if you are converging.

- Is your water table changing between time steps?
- Is groundwater storage changing (local versus global)?
- Are your surface water outflows changing?

Spinup Analysis

Look at your outputs and check if you are converging.

- You can do this visually with Visit
- You should also calculate your water balance and outflows at important locations in your domain

(see [section 4.9](#) for details on calculating a water balance, [section 8.3](#) for calculation examples with tcl and the [hydrology module](#) for tools to calculate water balance components with python)

What do I do if I'm stuck?

Getting stuck in spinup is very common. It can be challenging and will likely require close monitoring and frequent interventions. If you are stuck some of the first things to do are:

1. Look at your outputs and make sure you don't have something wrong with your domain.
2. Look at the kinsol.log file and see how the model is solving

Things to look for if it's not solving

1. Areas of very high or very low pressure in your outputs
 - These could be physical or they could be a problem with your inputs. Either way they make the problem hard to solve.
2. Lots of places where overland flow is just starting to form.
 - Switching on and off overland flow creates a lot of work for the solver. If streams are just starting to flow, this can explain slow behavior.

Spinup Knobs to Turn

2. Make your time step smaller or switch to a growth time step

```
model.TimeStep.Type = "Growth"  
model.TimeStep.InitialStep = 1.0  
model.TimeStep.GrowthFactor = 1.1  
model.TimeStep.MaxStep = 100  
model.TimeStep.MinStep = 1
```

3. Experiment with different initial conditions

- Sometimes it may solve better starting with the domain completely dry or completely wet rather than making a more intelligent initial guess

Spinup Knobs to Turn

4. Change your solver settings

Refer to the [manual section 6.34](#) for details and test scripts for examples. Some common options are:

- Change the preconditioner
(*Solver.Linear.Preconditioner.SymmetricMat*)
- Increase the number of nonlinear iterations you allow
(*Solver.Nonlinear.MaxIter*)
- Increase the convergence tolerance
(*Solver.Nonlinear.ResidualTol*)
- NOTE: Increasing the number of processors will improve your run time but it does not fix problems that don't converge

6. Additional setup for PF-CLM

Additional Inputs needed:

1. Land cover type
2. Properties of each land cover type (IGBP land cover classifications already setup)
3. Meteorological forcing data which can be spatially heterogeneous or homogeneous
 - Visible or short-wave radiation [W/m^2]
 - Long wave radiation [W/m^2]
 - Precipitation [mm/s]
 - Air Temperature [K]
 - East-west wind speed [m/s]
 - South-to-North wind speed [m/s]