Efficient Algorithms For Flow Modeling

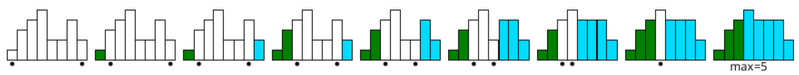
github.com/r-barnes/richdem

Richard Barnes

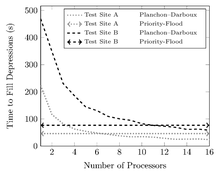
richdem.readthedocs.io

Faster

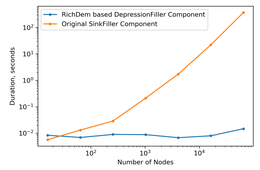
Depression-Filling with Priority-Flood



Does with one core what TauDEM does with 12



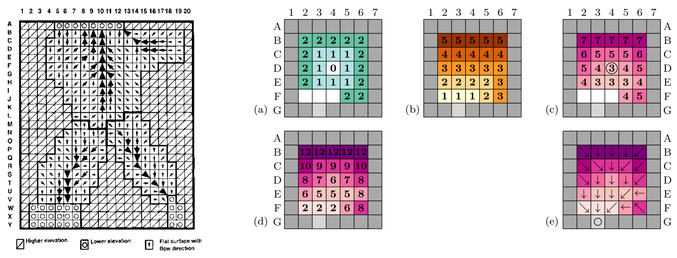
25,000x faster than LandLab's SinkFiller



Barnes, Lehman, Mulla. 2014. "Priority-Flood..."
Computers & Geosciences. doi: 10.1016/j.cageo.2013.04.024

Flat Resolution

1,763x faster than TauDEM
(53.3 minutes on 16 cores vs 30 seconds on 1 cores)

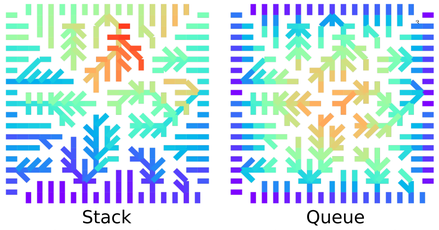


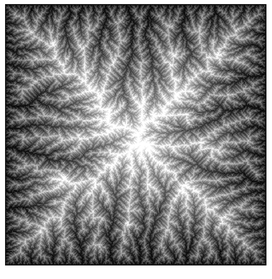
Barnes, Lehman, Mulla. 2014. "Efficient Assignment of Drainage Direction..."
Computers & Geosciences. doi: 10.1016/j.cageo.2013.01.009

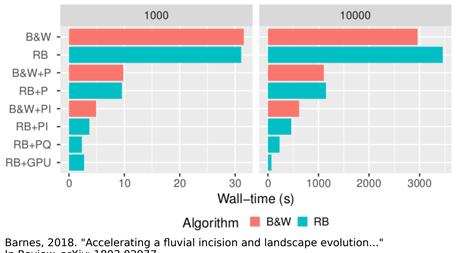
Parallelized Landscape Evolution with GPUs

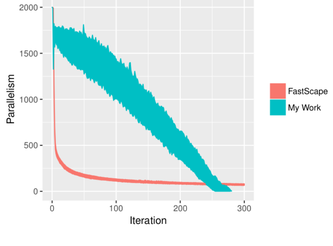
43x faster than FastScape
(70s vs. 3,000s on a 10,000 x 10,000 input)

When Cells Are Processed









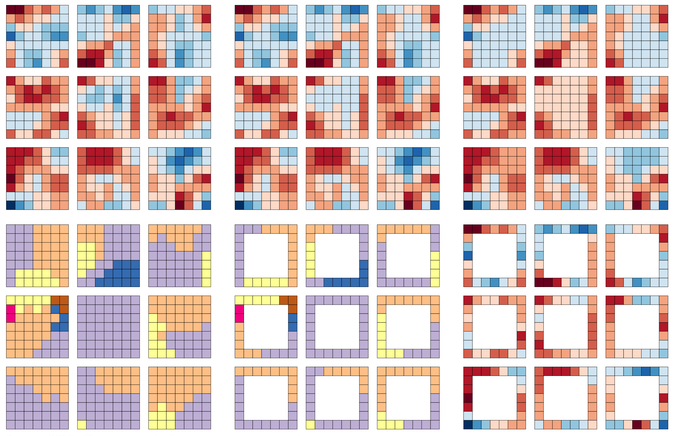
Barnes, 2018. "Accelerating a fluvial incision and landscape evolution..."
In Review. arXiv: 1803.02977

Larger

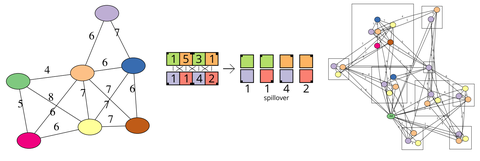
Source	Year	Cells	Dimensions	Adjective	Time (min)
This work (RichDEM)	2016	$2 \cdot 10^{12}$	$\sim$1,291,715^2	*rather* large	287
Gomes et al. [2012]	2012	$3 \cdot 10^{9}$	50,000 x 50,000	huge	58
Do et al. [2010]	2010	$2 \cdot 10^{9}$	36,002 x 54,002	huge	21
Do et al. [2011]	2011	$2 \cdot 10^{9}$	36,002 x 54,002	huge	??
Yıldırım et al. [2015] (TauDEM)	2015	$2 \cdot 10^{9}$	45,056 x 49,152	large	??
Arge et al. [2003] (GRASS)	2003	$1 \cdot 10^{9}$	33,454 x 31,866	massive	3720
Lindsay [2015] (Whitebox GAT)	2015	$9 \cdot 10^{8}$	37,201 x 25,201	massive	8.6
Tesfa et al. [2011]	2011	$6 \cdot 10^{8}$	24,856 x 24,000	large	20
Wallis et al. [2009] (TauDEM)	2009	$4 \cdot 10^{8}$	14,949 x 27,174	large	8
Danner et al. [2007]	2007	$3 \cdot 10^{8}$	??	massive	445
Metz et al. [2010, 2011] (GRASS)	2010	$2 \cdot 10^{8}$	??	massive	32

Parallel Continent-Scale Depression-Filling

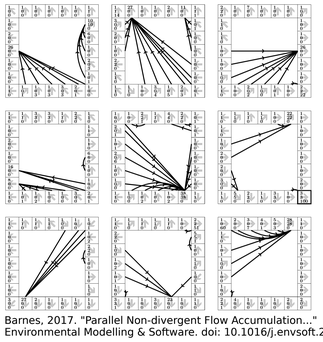
2 Trillion Cells in 15 Minutes



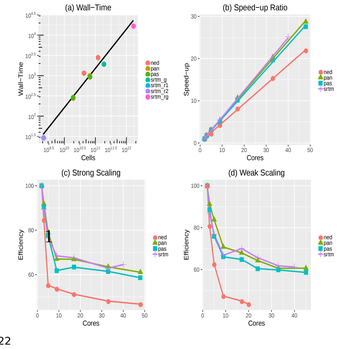
Barnes. 2016. "Parallel priority-flood depression filling..."
Computers & Geosciences. doi: 10.1016/j.cageo.2016.07.001



Parallel Continent-Scale Flow Accumulation



Linear Scaling Over 3 Orders of Magnitude



Barnes, 2017. "Parallel Non-divergent Flow Accumulation..."
Environmental Modelling & Software. doi: 10.1016/j.envsoft.2017.02.022

DEM	Cells	Size GB	Wall-Time Min	All Time Hrs	Producer Sec	Tx MB	Tx/Tile KB	Tile Ram MB	Peak RAM MB
SRTM Resampled	$1.7 \cdot 10^{12}$	3,340	287	223	84	4,159	291	1,307	12,236
SRTM Global	$1.9 \cdot 10^{11}$	371	33	26	37	1,481	104	209	6,011
NED	$1.2 \cdot 10^{11}$	478	48	37	6	386	377	1,725	1,295
PAMAP North	$6.5 \cdot 10^{10}$	260	17	13	12	729	109	234	1,943
PAMAP South	$6.6 \cdot 10^{10}$	263	16	12	10	719	107	233	1,703
SRTM Region 1	$2.1 \cdot 10^{9}$	4	0.5	0.3	0.3	16	99	184	478
SRTM Region 2	$2.1 \cdot 10^{9}$	4	0.5	0.3	0.4	17	105	162	494

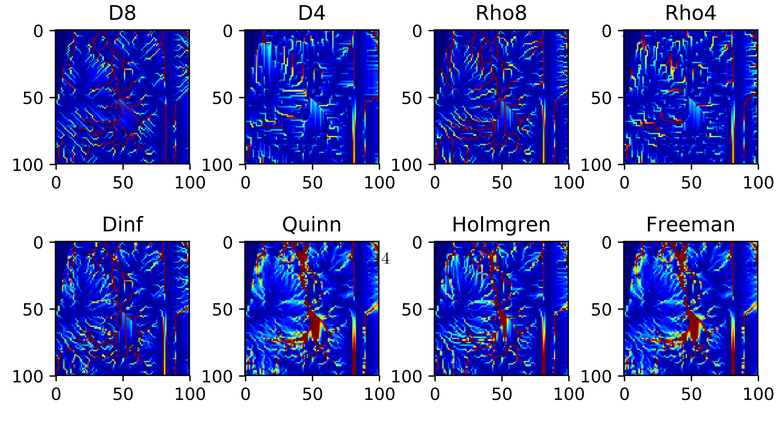
Easier

Open-source, well-commented C++ library for model primitives accessible cross-platform via Python, R, and Fortran wrappers as well as the command-line

```python
import richdem as rd

#Load data into a numpy array
dem = rd.rdarray("my_dem.tif", no_data=-9999)

#Breach depressions, return the answer in a new Numpy array
breached = rd.BreachDepressions(beau, in_place=False)

#Fill depressions while enforcing a drainage gradient
#Do not allocate new memory
rd.FillDepressions(dem, epsilon=True, in_place=True)

#Calculate flow accumulation according to the D-infinity method
#A new Numpy array is created
accum_d8 = rd.FlowAccumulation(dem, method='Dinf')

#Calculate a terrain attribute
profile_curvature = rd.TerrainAttribute(dem, attrib='profile_curvature')
```

```cpp
#include <richdem/richdem.hpp>
using namespace richdem;

int main(){
  Array2D<float> dem("my_dem.tif");      //Prepare file for opening
  dem.loadData();                        //Explicitly load data when needed
  CompleteBreaching_Lindsay2016(dem);    //Use a specific breaching algorithm
  Array2D<double> planform(dem);         //Construct a new array
  TA_planform_curvature(dem, planform); //Calculate planform curvature
}
```

Many options



Automated Provenance tracking

```
2018-05-23 15:48:33.061 UTC | RichDEM (Python 0.2.0) (hash=091f312) | LoadGDAL(filename=/home/rick/data/gis/beaufort.tif, no_data=-9999.0)
2018-05-23 15:48:33.065 UTC | RichDEM (Python 0.2.0) (hash=091f312) | /home/rick/.local/bin/rd_depression_filling /home/rick/data/gis/beaufort.tif /z/filled.tif
2018-05-23 15:48:33.066 UTC | RichDEM (Python 0.2.0) (hash=091f312) | FillDepressions(dem, epsilon=False)
2018-05-23 15:49:19.449 UTC | RichDEM (Python 0.2.0) (hash=091f312) | LoadGDAL(filename=/z/filled.tif, no_data=-9999.0)
2018-05-23 15:49:19.451 UTC | RichDEM (Python 0.2.0) (hash=091f312) | /home/rick/.local/bin/rd_flow_accumulation -m Quinn /z/filled.tif /z/accum.tif
2018-05-23 15:49:19.469 UTC | RichDEM (Python 0.2.0) (hash=091f312) | FlowAccumulation(dem, method=Quinn, exponent=None, weights=None,
                                                                       in_place=False)
```


rbar@berkeley.edu

NSF DOE CSGF