waveCUDA: A new CUDA-accelerated R package for wavelet analysis

Julian Waton
Supervisor: Dr Emma McCoy

Imperial College London

2 July 2014
R User Conference 2014

Outline

Purpose of waveCUDA

- Performing CUDA-accelerated wavelet analysis

Purpose of waveCUDA

- Performing CUDA-accelerated wavelet analysis
- Standard DWT already fast

Purpose of waveCUDA

- Performing CUDA-accelerated wavelet analysis
- Standard DWT already fast in 1D

Purpose of waveCUDA

- Performing CUDA-accelerated wavelet analysis
- Standard DWT already fast in 1D - but can be faster

Purpose of waveCUDA

- Performing CUDA-accelerated wavelet analysis
- Standard DWT already fast in 1D - but can be faster
- There are already packages in R for wavelet analysis

Purpose of waveCUDA

- Performing CUDA-accelerated wavelet analysis
- Standard DWT already fast in 1D - but can be faster
- There are already packages in R for wavelet analysis - *wmtsa* (Constantine & Percival, 2013), *wavethresh* (Nason, 2013), *waveslim* (Whitcher, 2013)

Purpose of waveCUDA

- Performing CUDA-accelerated wavelet analysis
- Standard DWT already fast in 1D - but can be faster
- There are already packages in R for wavelet analysis - *wmtsa* (Constantine & Percival, 2013), *wavethresh* (Nason, 2013), *waveslim* (Whitcher, 2013) - but not GPU-accelerated

Purpose of waveCUDA

- Performing CUDA-accelerated wavelet analysis
- Standard DWT already fast in 1D - but can be faster
- There are already packages in R for wavelet analysis - *wmtsa* (Constantine & Percival, 2013), *wavethresh* (Nason, 2013), *waveslim* (Whitcher, 2013) - but not GPU-accelerated
- There are also a small number of CUDA-accelerated R packages

Purpose of waveCUDA

- Performing CUDA-accelerated wavelet analysis
- Standard DWT already fast in 1D - but can be faster
- There are already packages in R for wavelet analysis - *wmtsa* (Constantine & Percival, 2013), *wavethresh* (Nason, 2013), *waveslim* (Whitcher, 2013) - but not GPU-accelerated
- There are also a small number of CUDA-accelerated R packages - *gputools* (Buckner *et al.* , 2013), *HiPLARM* (Montana *et al.* , 2013)

Purpose of waveCUDA

- Performing CUDA-accelerated wavelet analysis
- Standard DWT already fast in 1D - but can be faster
- There are already packages in R for wavelet analysis - *wmtsa* (Constantine & Percival, 2013), *wavethresh* (Nason, 2013), *waveslim* (Whitcher, 2013) - but not GPU-accelerated
- There are also a small number of CUDA-accelerated R packages - *gputools* (Buckner *et al.* , 2013), *HiPLARM* (Montana *et al.* , 2013)
- I want to be able to do wavelet analysis more quickly in R!

Applications of wavelets

- Time-frequency analysis of time series

Applications of wavelets

- Time-frequency analysis of time series
- Data/image smoothing

Applications of wavelets

- Time-frequency analysis of time series
- Data/image smoothing
- Compression - JPEG2000

Applications of wavelets

- Time-frequency analysis of time series
- Data/image smoothing
- Compression - JPEG2000

Applications of wavelets

- Time-frequency analysis of time series
- Data/image smoothing
- Compression - JPEG2000

- Financial time series - multi-temporal analysis

Applications of wavelets

- Time-frequency analysis of time series
- Data/image smoothing
- Compression - JPEG2000

- Financial time series - multi-temporal analysis
- WNN

Applications of wavelets

- Time-frequency analysis of time series
- Data/image smoothing
- Compression - JPEG2000

- Financial time series - multi-temporal analysis
- WNN
- Fault detection

What are Wavelets?

The (1D) DWT convolves a vector with high-pass and low-pass filters.

What are Wavelets?

The (1D) DWT convolves a vector with high-pass and low-pass filters.

$$c_{J-1,k} = \sum_{l=-\infty}^{\infty} g_l x_{2k-l},$$

$$d_{J-1,k} = \sum_{l=-\infty}^{\infty} h_l x_{2k-l},$$

for $k = 1, \ldots, 2^{J-1}$, which gives the first level of filtering.

What are Wavelets?

The (1D) DWT convolves a vector with high-pass and low-pass filters.

$$c_{J-1,k} = \sum_{l=-\infty}^{\infty} g_l x_{2k-l},$$

$$d_{J-1,k} = \sum_{l=-\infty}^{\infty} h_l x_{2k-l},$$

for $k = 1, \ldots, 2^{J-1}$, which gives the first level of filtering.
Then we recursively filter the scaling coefficients, $c_{j,k}$

What are Wavelets?

The (1D) DWT convolves a vector with high-pass and low-pass filters.

$$c_{J-1,k} = \sum_{l=-\infty}^{\infty} g_l x_{2k-l},$$

$$d_{J-1,k} = \sum_{l=-\infty}^{\infty} h_l x_{2k-l},$$

for $k = 1, \ldots, 2^{J-1}$, which gives the first level of filtering.
Then we recursively filter the scaling coefficients, $c_{j,k}$

$$c_{j-1,k} = \sum_{l=-\infty}^{\infty} g_l c_{j,2k-l},$$

$$d_{j-1,k} = \sum_{l=-\infty}^{\infty} h_l c_{j,2k-l},$$

for $k = 1, \ldots, 2^j, j = J-1, \ldots, 1$

What are Wavelets?

The (1D) DWT convolves a vector with high-pass and low-pass filters.

$$c_{J-1,k} = \sum_{l=-\infty}^{\infty} g_l x_{2k-l},$$

$$d_{J-1,k} = \sum_{l=-\infty}^{\infty} h_l x_{2k-l},$$

for $k = 1, \ldots, 2^{J-1}$, which gives the first level of filtering.
Then we recursively filter the scaling coefficients, $c_{j,k}$

$$c_{j-1,k} = \sum_{l=-\infty}^{\infty} g_l c_{j,2k-l},$$

$$d_{j-1,k} = \sum_{l=-\infty}^{\infty} h_l c_{j,2k-l},$$

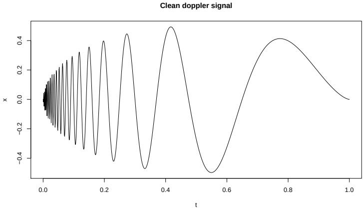
for $k = 1, \ldots, 2^j, j = J-1, \ldots, 1$
Good books on wavelets are (Nason, 2008) and (Percival & Walden, 2000).

What are Wavelets?

The DWT is a time-frequency transform

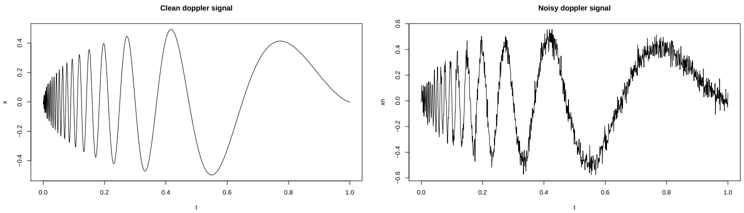
What are Wavelets?

The DWT is a time-frequency transform



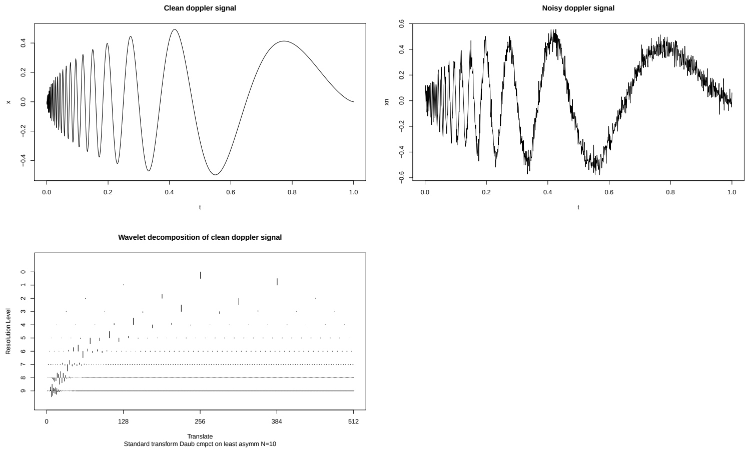
What are Wavelets?

The DWT is a time-frequency transform



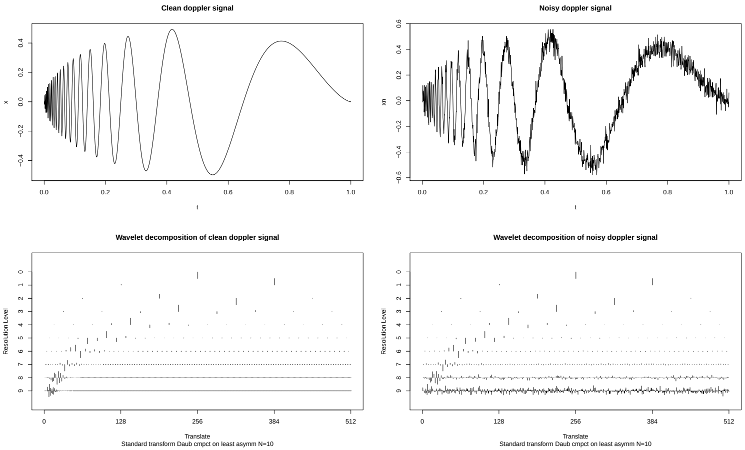
What are Wavelets?

The DWT is a time-frequency transform



What are Wavelets?

The DWT is a time-frequency transform



What is CUDA?

- *Compute Unified Device Architecture*

What is CUDA?

- *Compute Unified Device Architecture* !
- Markup language for C/C++ (Nvidia, 2011)

What is CUDA?

- *Compute Unified Device Architecture* !
- Markup language for C/C++ (Nvidia, 2011)
- Enables programmers to use graphics card as a GPGPU (*General Purpose Graphics Processor Unit*)

What is CUDA?

- *Compute Unified Device Architecture* !
- Markup language for C/C++ (Nvidia, 2011)
- Enables programmers to use graphics card as a GPGPU (*General Purpose Graphics Processor Unit*)
- Graphics cards are ideal for highly parallel SIMT (Simple Instruction, Multiple Thread) tasks

What is CUDA?

- *Compute Unified Device Architecture* !
- Markup language for C/C++ (Nvidia, 2011)
- Enables programmers to use graphics card as a GPGPU (*General Purpose Graphics Processor Unit*)
- Graphics cards are ideal for highly parallel SIMT (Simple Instruction, Multiple Thread) tasks
- Thanks to the gaming industry, GPU technology has been advancing at a higher rate than Moore's Law

What is CUDA?

- *Compute Unified Device Architecture* !
- Markup language for C/C++ (Nvidia, 2011)
- Enables programmers to use graphics card as a GPGPU (*General Purpose Graphics Processor Unit*)
- Graphics cards are ideal for highly parallel SIMT (Simple Instruction, Multiple Thread) tasks
- Thanks to the gaming industry, GPU technology has been advancing at a higher rate than Moore's Law

	Intel i7-3740QM CPU	**NVIDIA GeForce 650M GPU**
Cores	4	384
RAM	16GB	2GB
Cache	6MB	256KB
Clock speed	2.7 GHz	0.83GHz
Max memory bandwidth	25.6GB/s	80GB/s

Parallelising wavelets with CUDA

- DWT is parallelisable thanks to Sweldens (Sweldens, 1996, 1998; Daubechies & Sweldens, 1998)

Parallelising wavelets with CUDA

- DWT is parallelisable thanks to Sweldens (Sweldens, 1996, 1998; Daubechies & Sweldens, 1998)
- Wavelet transforms have already been implemented in CUDA (van der Laan, 2011)

Parallelising wavelets with CUDA

- DWT is parallelisable thanks to Sweldens (Sweldens, 1996, 1998; Daubechies & Sweldens, 1998)
- Wavelet transforms have already been implemented in CUDA (van der Laan, 2011)
- Optimisations used - shared memory, block size, fiddly kernels performing multiple layers of transform

Parallelising wavelets with CUDA

- DWT is parallelisable thanks to Sweldens (Sweldens, 1996, 1998; Daubechies & Sweldens, 1998)
- Wavelet transforms have already been implemented in CUDA (van der Laan, 2011)
- Optimisations used - shared memory, block size, fiddly kernels performing multiple layers of transform
- What has been implemented so far: Haar, Daubechies 4, thresholding, cross-validation

Speedups

I can do a live demo! Wish me luck...

How to get waveCUDA

Requirements:

- nVidia CUDA-capable GPU
- CUDA 5.5
- Linux

Download package from CRAN

How to get waveCUDA

Requirements:

- nVidia CUDA-capable GPU
- CUDA 5.5
- Linux

Download package from CRAN ...coming soon to CRAN repositories near you!

Further work

- This is a package in its infancy!

Further work

- This is a package in its infancy!
- Further optimisations are planned - more speedups!

Further work

- This is a package in its infancy!
- Further optimisations are planned - more speedups!
  - in/output vectors - avoids extra kernels

Further work

- This is a package in its infancy!
- Further optimisations are planned - more speedups!
  - in/output vectors - avoids extra kernels
  - streams - run simultaneous transforms

Further work

- This is a package in its infancy!
- Further optimisations are planned - more speedups!
  - in/output vectors - avoids extra kernels
  - streams - run simultaneous transforms
- More wavelet filters

Further work

- This is a package in its infancy!
- Further optimisations are planned - more speedups!
  - in/output vectors - avoids extra kernels
  - streams - run simultaneous transforms
- More wavelet filters
- Other wavelet transforms

Further work

- This is a package in its infancy!
- Further optimisations are planned - more speedups!
  - in/output vectors - avoids extra kernels
  - streams - run simultaneous transforms
- More wavelet filters
- Other wavelet transforms
  - Non-decimated/maximal overlap/stationary/a trous

Further work

- This is a package in its infancy!
- Further optimisations are planned - more speedups!
  - in/output vectors - avoids extra kernels
  - streams - run simultaneous transforms
- More wavelet filters
- Other wavelet transforms
  - Non-decimated/maximal overlap/stationary/a trous
  - 2D transform - even bigger speedups

Interest

Are you interested in this package? (rhetorical question that doesn't need answering now)
Email wavecuda@imperial.ac.uk if you would like to be updated when it is uploaded onto CRAN.

Acknowledgements

We are very grateful to BP for providing funding for this research.

References

Buckner, Josh, Seligman, Mark, & Wilson, Justin. 2013. gputools: A few GPU enabled functions. *http://CRAN.R-project.org/package=gputools*, R package version 0.28.

Constantine, William, & Percival, Donald. 2013. wmtsa: Wavelet Methods for Time Series Analysis. *http://cran.r-project.org/package=wmtsa*, R package version 2.0–0.

Daubechies, I, & Sweldens, W. 1998. Factoring wavelet transforms into lifting steps. *Journal of Fourier analysis and applications*, 1–26.

Montana, Giovanni, Nash, Peter, & Szeremi, Vendel. 2013. HiPLARM: High Performance Linear Algebra in R. *http://cran.r-project.org/web/packages/HiPLARM*, R package version 0.1.

Nason, Guy. 2008. *Wavelet Methods In Statistics With R*. Springer.

Nason, Guy. 2013. wavethresh: Wavelets statistics and transforms. *http://cran.r-project.org/package=wavethresh*, R package version 4.6.6.

Nvidia. 2011. NVIDIA CUDA programming guide.

Percival, Donald, & Walden, Andrew. 2000. *Wavelet Methods For Time Series Analysis*. Cambridge University Press.

Sweldens, W. 1998. The lifting scheme: A construction of second generation wavelets. *SIAM Journal on Mathematical Analysis*, **29**(2), 511–546.

Sweldens, Wim. 1996. The Lifting Scheme: A Custom-Design Construction of Biorthogonal Wavelets. *Applied and Computational Harmonic Analysis*, **3**(2), 186–200.

van der Laan, WJ. 2011. Accelerating wavelet lifting on graphics hardware using CUDA. *IEEE transactions on parallel and distributed systems*, **22**(1), 132–146.

Whitcher, Brandon. 2013. waveslim: Basic wavelet routines for one-, two- and three-dimensional signal processing. *http://cran.r-project.org/package=waveslim*, R package version 1.7.3.