A tree projection algorithm for wavelet-based sparse approximation

Andrew Thompson

Duke University, North Carolina, USA

joint with Coralia Cartis (University of Edinburgh)

Wavelet trees

- Discrete wavelet transforms (DWTs) have an inherent **tree structure**

Wavelet trees

- Discrete wavelet transforms (DWTs) have an inherent **tree structure**
- Discontinuities propagate down branches of the tree

Wavelet trees

- Discrete wavelet transforms (DWTs) have an inherent **tree structure**
- Discontinuities propagate down branches of the tree
- $\Rightarrow$ **Why the DWT is so effective:** it provides sparse representations for piecewise smooth signals

Wavelet trees

- Discrete wavelet transforms (DWTs) have an inherent **tree structure**
- Discontinuities propagate down branches of the tree
- $\Rightarrow$ **Why the DWT is so effective:** it provides sparse representations for piecewise smooth signals
- ...also, we expect it to be **tree-sparse**:
  the coefficients form a rooted subtree.

Tree projection

- z is k**-tree sparse** if it is supported on a rooted tree of cardinality k:

$$z \in \mathcal{T}_k := \{k\text{-tree sparse vectors in } \mathbb{R}^N\}.$$

Tree projection

- z is k**-tree sparse** if it is supported on a rooted tree of cardinality k:

$$z \in \mathcal{T}_k := \{k\text{-tree sparse vectors in } \mathbb{R}^N\}.$$

- Let $x \in \mathbb{R}^N$ and let $\{x_i : i = 1, \ldots, N\}$ correspond to nodes of a tree of order d.

Tree projection

- z is k**-tree sparse** if it is supported on a rooted tree of cardinality k:

$$z \in \mathcal{T}_k := \{k\text{-tree sparse vectors in } \mathbb{R}^N\}.$$

- Let $x \in \mathbb{R}^N$ and let $\{x_i : i = 1, \ldots, N\}$ correspond to nodes of a tree of order d.

- **Tree projection:** Find the k-tree sparse z which is closest in Euclidean distance to x:

$$\mathcal{P}_k(x) := \arg\min_{z \in \mathcal{T}_k} \|z - x\|_2.$$

Applications of tree projections

- Classification and regression trees (CART)

  (Bohanec/Bratko,1994)

Applications of tree projections

- Classification and regression trees (CART)

  (Bohanec/Bratko,1994)

- The *Best Orthogonal Basis* method

  (Donoho,1997)

Applications of tree projections

- Classification and regression trees (CART)

  (Bohanec/Bratko,1994)

- The *Best Orthogonal Basis* method

  (Donoho,1997)

- Wavelet compression

  (Cohen/Dahmen/Daubechies/Devore,2001)

Applications of tree projections

- Classification and regression trees (CART)

  (Bohanec/Bratko,1994)

- The *Best Orthogonal Basis* method

  (Donoho,1997)

- Wavelet compression

  (Cohen/Dahmen/Daubechies/Devore,2001)

- Tree-based Compressed Sensing for wavelets.

  (Baraniuk et al.,2010)

Existing algorithms for wavelets

- **Greedy tree approximation** (GTA)
  Choose $l < k$, take the largest l coefficients, fill out to a rooted tree, repeat until the rooted tree is of size k.
  (Cohen/Dahmen/Daubechies/Devore,2001)

Existing algorithms for wavelets

- **Greedy tree approximation** (GTA)
  Choose $l < k$, take the largest l coefficients, fill out to a rooted tree, repeat until the rooted tree is of size k.
  (Cohen/Dahmen/Daubechies/Devore,2001)
- **Complexity penalized sum of squares** (CPSS)
  (Donoho,1997 - based on Breiman et al.,1984)

Existing algorithms for wavelets

- **Greedy tree approximation** (GTA)

  Choose $l < k$, take the largest l coefficients, fill out to a rooted tree, repeat until the rooted tree is of size k.

  (Cohen/Dahmen/Daubechies/Devore,2001)

- **Complexity penalized sum of squares** (CPSS)

  (Donoho,1997 - based on Breiman et al.,1984)

- **Condensing sort and select algorithm** (CSSA)

  (Baraniuk/Jones,1994)

Optimization formulation

- $\mathcal{P}_k$ preserves the value of selected coefficients.

$$\{\mathcal{P}_k(x)\}_i := \begin{cases} x_i & i \in \Gamma \\ 0 & i \notin \Gamma \end{cases} \qquad \text{where} \qquad \Gamma := \operatorname{supp}\{\mathcal{P}_k(x)\}\,.$$

Optimization formulation

- $\mathcal{P}_k$ preserves the value of selected coefficients.

$$\{\mathcal{P}_k(x)\}_i := \begin{cases} x_i & i \in \Gamma \\ 0 & i \notin \Gamma \end{cases} \quad \text{where} \quad \Gamma := \text{supp}\,\{\mathcal{P}_k(x)\}\,.$$

- $\Rightarrow$ equivalent to finding the rooted tree of size k whose coefficients have maximum energy.

Optimization formulation

- $\mathcal{P}_k$ preserves the value of selected coefficients.

$$\{\mathcal{P}_k(x)\}_i := \begin{cases} x_i & i \in \Gamma \\ 0 & i \notin \Gamma \end{cases} \quad \text{where} \quad \Gamma := \text{supp}\,\{\mathcal{P}_k(x)\}\,.$$

- $\Rightarrow$ equivalent to finding the rooted tree of size k whose coefficients have maximum energy.

- $\Rightarrow$ we can formulate as an Integer Program (IP).

Optimization formulation/relaxations

$$\max_{\tau \in \mathbb{Z}^N} \sum_{i=1}^{N} x_i^2 \tau_i \quad \text{subject to} \quad \begin{aligned} &\tau \geq 0 \\ &\{\tau_i\} \text{ tree-nonincreasing} \\ &\sum \tau_i = k \\ &\tau_1 = 1. \end{aligned} \qquad \text{(IP)}$$

- $\{\mathcal{P}_k(x)\}_i = x_i \tau_i^*$, where τ^* solves (IP).

Optimization formulation/relaxations

$$\max_{\tau \in \mathbb{Z}^N} \sum_{i=1}^{N} x_i^2 \tau_i \quad \text{subject to} \quad \begin{array}{l} \tau \geq 0 \\ \{\tau_i\} \text{ tree-nonincreasing} \\ \sum \tau_i = k \\ \tau_1 = 1. \end{array} \qquad \text{(IP)}$$

- $\{\mathcal{P}_k(x)\}_i = x_i \tau_i^*$, where τ^* solves (IP).
- Linear Programming relaxation
  $\Rightarrow$ CSSA (Baraniuk/Jones,1994): complexity $\mathcal{O}(N \log N)$

Optimization formulation/relaxations

$$\max_{\tau \in \mathbb{Z}^N} \sum_{i=1}^{N} x_i^2 \tau_i \quad \text{subject to} \quad \begin{array}{l} \tau \geq 0 \\ \{\tau_i\} \text{ tree-nonincreasing} \\ \sum \tau_i = k \\ \tau_1 = 1. \end{array} \qquad \text{(IP)}$$

- $\{\mathcal{P}_k(x)\}_i = x_i \tau_i^*$, where τ^* solves (IP).
- Linear Programming relaxation
  $\Rightarrow$ CSSA (Baraniuk/Jones,1994): complexity $\mathcal{O}(N \log N)$
- Langrangian relaxation: penalize $\sum \tau_i = k$ in the objective
  $\Rightarrow$ CPSS (Donoho,1997): complexity $\mathcal{O}(N \log N)$

Optimization formulation/relaxations

$$\max_{\tau \in \mathbb{Z}^N} \sum_{i=1}^{N} x_i^2 \tau_i \quad \text{subject to} \quad \begin{array}{l} \tau \geq 0 \\ \{\tau_i\} \text{ tree-nonincreasing} \\ \sum \tau_i = k \\ \tau_1 = 1. \end{array} \qquad \text{(IP)}$$

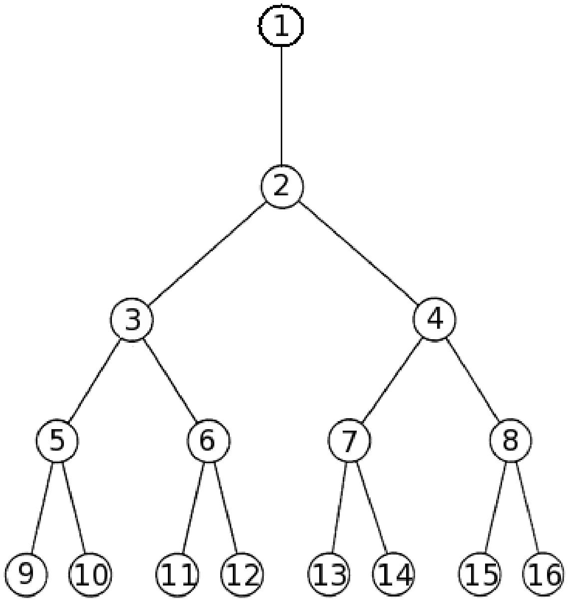
- $\{\mathcal{P}_k(x)\}_i = x_i \tau_i^*$, where τ^* solves (IP).
- Linear Programming relaxation
  $\Rightarrow$ CSSA (Baraniuk/Jones,1994): complexity $\mathcal{O}(N \log N)$
- Langrangian relaxation: penalize $\sum \tau_i = k$ in the objective
  $\Rightarrow$ CPSS (Donoho,1997): complexity $\mathcal{O}(N \log N)$
- $\Rightarrow$ neither guaranteed to exactly find $\mathcal{P}_k$.

Exact tree projection (ETP)

- **Principal of optimality:** optimal trees rooted at each node depend only on optimal trees rooted at each of its children

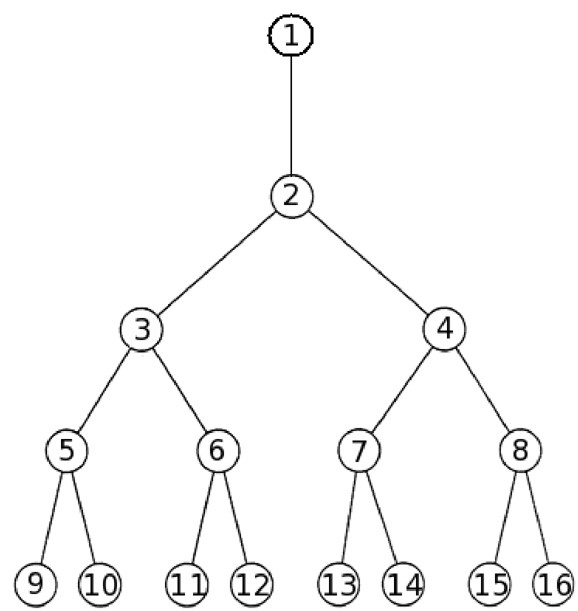
Exact tree projection (ETP)

- **Principal of optimality:** optimal trees rooted at each node depend only on optimal trees rooted at each of its children
- $\Rightarrow$ use fine-to-coarse Dynamic Programming



Exact tree projection (ETP)

- **Principal of optimality:** optimal trees rooted at each node depend only on optimal trees rooted at each of its children
- $\Rightarrow$ use fine-to-coarse Dynamic Programming



- A similar algorithm was used for CART (Bohanec/Bratko,1994).

Exact tree projection (ETP)...

- Proceed from fine to coarse scales

Exact tree projection (ETP)...

- Proceed from fine to coarse scales
- Find optimal subtrees for all $1 \leq \tilde{k} \leq k$ for each node

Exact tree projection (ETP)...

- Proceed from fine to coarse scales
- Find optimal subtrees for all $1 \leq \tilde{k} \leq k$ for each node
- Record precedence information at each node

Exact tree projection (ETP)...

- Proceed from fine to coarse scales
- Find optimal subtrees for all $1 \leq \tilde{k} \leq k$ for each node
- Record precedence information at each node
- ...and finally for the root node

Exact tree projection (ETP)...

- Proceed from fine to coarse scales
- Find optimal subtrees for all $1 \leq \tilde{k} \leq k$ for each node
- Record precedence information at each node
- ...and finally for the root node
- Trace back precedences to identify solution.

Complexity of ETP

- **Theorem:** ETP requires at most $(3d^2Nk + N)$ additions/comparisons to calculate $\mathcal{P}_k$.
  $\Rightarrow$ complexity is $\mathcal{O}(Nk)$

Complexity of ETP

- **Theorem:** ETP requires at most $(3d^2Nk + N)$ additions/comparisons to calculate $\mathcal{P}_k$.
  $\Rightarrow$ complexity is $\mathcal{O}(Nk)$

- An improvement on $\mathcal{O}(N^2)$, which was obtained for the similar CART algorithm (Bohanec/Bratko,1994)

Complexity of ETP

- **Theorem:** ETP requires at most $(3d^2Nk + N)$ additions/comparisons to calculate $\mathcal{P}_k$.
  $\Rightarrow$ complexity is $\mathcal{O}(Nk)$

- An improvement on $\mathcal{O}(N^2)$, which was obtained for the similar CART algorithm (Bohanec/Bratko,1994)

- **Comparison with approximate algorithms:** both CPSS and CSSA are $\mathcal{O}(N \log N)$

Complexity of ETP

- **Theorem:** ETP requires at most $(3d^2Nk + N)$ additions/comparisons to calculate $\mathcal{P}_k$.
  $\Rightarrow$ complexity is $\mathcal{O}(Nk)$

- An improvement on $\mathcal{O}(N^2)$, which was obtained for the similar CART algorithm (Bohanec/Bratko,1994)

- **Comparison with approximate algorithms:** both CPSS and CSSA are $\mathcal{O}(N \log N)$

- $\Rightarrow$ if $\log N \ll k$, ETP guarantees exact optimality at the expense of a worse order of complexity.

An application of ETP

- **Compressed sensing:** Recover $x \in \mathbb{R}^n$ from $b = Ax \in \mathbb{R}^m$ where $m < n$

An application of ETP

- **Compressed sensing:** Recover $x \in \mathbb{R}^n$ from $b = Ax \in \mathbb{R}^m$ where $m < n$

- Assume x has simplicity structure: **tree-sparse**

An application of ETP

- **Compressed sensing:** Recover $x \in \mathbb{R}^n$ from $b = Ax \in \mathbb{R}^m$ where $m < n$

- Assume x has simplicity structure: **tree-sparse**

- **Gradient projection algorithm:**

  (Blumensath/Davies,2009;Baraniuk et al.,2010)

$$x^{i+1} := \mathcal{P}_k \left\{ x^i + \alpha A^T (b - Ax^i) \right\}$$

An application of ETP...

$$x^{i+1} := \mathcal{P}_k \left\{ x^i + \alpha A^T (b - Ax^i) \right\}$$

Recovery guarantee for A Gaussian: (CC/AT,2013)

Theorem: Let

$$n \to \infty \text{ and } \frac{k}{n} \to \rho.$$

Then

$$\mathbb{P}(\text{recovery of } x) \to 1$$

provided $\rho < 0.0201$.

An application of ETP...

$$x^{i+1} := \mathcal{P}_k \left\{ x^i + \alpha A^T (b - Ax^i) \right\}$$

Recovery guarantee for A Gaussian: (CC/AT,2013)

Theorem: Let

$$n \to \infty \quad \text{and} \quad \frac{k}{n} \to \rho.$$

Then

$$\mathbb{P}(\text{recovery of } x) \to 1$$

provided $\rho < 0.0201$.

$\Rightarrow \quad \mathcal{P}_k$ must be exact for result to hold.

Conclusions

- We have proposed the ETP algorithm for wavelets

Conclusions

- We have proposed the ETP algorithm for wavelets
- First algorithm in this context guaranteed to be exact

Conclusions

- We have proposed the ETP algorithm for wavelets
- First algorithm in this context guaranteed to be exact
- ETP has complexity $\mathcal{O}(Nk)$

Conclusions

- We have proposed the ETP algorithm for wavelets
- First algorithm in this context guaranteed to be exact
- ETP has complexity $\mathcal{O}(Nk)$
- Practitioner's choice of algorithm will depend upon whether algorithm running time or guaranteed precision is the overriding priority

Conclusions

- We have proposed the ETP algorithm for wavelets
- First algorithm in this context guaranteed to be exact
- ETP has complexity $\mathcal{O}(Nk)$
- Practitioner's choice of algorithm will depend upon whether algorithm running time or guaranteed precision is the overriding priority
- **Paper:** *An exact tree projection algorithm for wavelets*

Conclusions

- We have proposed the ETP algorithm for wavelets
- First algorithm in this context guaranteed to be exact
- ETP has complexity $\mathcal{O}(Nk)$
- Practitioner's choice of algorithm will depend upon whether algorithm running time or guaranteed precision is the overriding priority
- **Paper:** *An exact tree projection algorithm for wavelets*
- **Code:** both available at `www.math.duke.edu/~thompson`

References

- *Trading accuracy for simplicity in decision trees*; M.Bohanec, I.Bratko (Machine Learning, 1994)
- *A signal-dependent time-frequency representation: Fast algorithm for optimal kernel design*; R.Baraniuk, D.Jones (IEEE Trans. on Signal Proc., 1994)
- *CART and best ortho-basis: A connection*; D.Donoho (Annals of Statistics, 1997)
- *Tree approximation and optimal encoding*; A.Cohen, W.Dahmen, I.Daubechies, R.DeVore (Applied and Comp. Harm. Anal., 2001)
- *Classification and regression trees*; L.Breiman, J.Friedman, R.Olshen, C.Stone (Chapman and Hall, 1984)
- *Model-based compressive sensing*; R.Baraniuk, V.Cevher, M. Duarte, C. Hegde (IEEE Trans. on Inform. Theory, 2010)
- *Sampling theorems for signals from the union of finite-dimensional linear subspaces*; T.Blumensath, M.Davies (IEEE Trans. on Inform. Theory, 2009)
- *Quantitative recovery conditions for iterative tree projection*; CC, AT (in preparation).