Recursive Decomposition for Nonconvex Optimization

Abram L. Friesen and Pedro Domingos {afriesen, pedrod}@cs.washington.edu

Nonconvex Optimization

Global optimization of nonconvex functions is generally intractable because of the combinatorial number of modes in the objective function. Recursive decomposition algorithms can explore a combinatorially large space in sub-exponential time but only exist in discrete domains (e.g., SAT, model counting, probabilistic inference).

We introduce RDIS, a local, recursive decomposition algorithm for continuous optimization. Existing continuous methods are non-recursive, and require that the decomposition be pre-specified, global, and static. However, many problems exhibit local structure (i.e., dependencies change as a function of the state space).

Recursive decomposition allows RDIS to exploit local structure. We show it is able to find the global minimum in exponentially less time than standard algorithms for nonconvex optimization for a class of functions that exhibit local structure.

Other benefits of RDIS include:

- RDIS optimizes small, independent blocks of variables, resulting in updates that are faster, more consistent, and move further.
- RDIS simplifies the objective function, resulting in both reduced computation and smoothing.
- Locality guarantees more decomposition than alternatives.
- Nested restart behavior plus local decomposition leads to exponential reductions in complexity while retaining global convergence guarantees.

Local Structure

Goal is to minimize $f(x)$ for $x \in \mathbb{R}$. Fully decomposable functions, $f(x) = \sum_i f_i(x_i)$, are easy to optimize because $\min f(x) = \sum_i \min f_i(x_i)$, but rare.

Conversely, non-decomposed functions require exponentially more exploration than the decomposed function. For example, consider $f(x) = \sum_i f_i(x_i)$ and let M_i be the modes of f_i. Then the number of modes to explore is $|M| = \sum_i |M_i|$. However, if f is instead optimized directly, then $\prod_i |M_i|$ modes must be explored, which is exponential in n.

To maximize decomposition, we define the following types of structure.

Definition 1. $f(x)$ is globally decomposable if there exists a partition $x = \{x_1, x_2, x_3\}$ such that, for every value $a \in \mathrm{dom}(x_1)$, $f(a, x_2, x_3) = f_1(a, x_2) + f_2(a, x_3)$.

Definition 2. $f(x)$ is locally decomposable in the subspace $x_1 = a$ if there exists a partition $x = \{x_1, x_2, x_3\}$ such that $f(a, x_2, x_3) = f_1(a, x_2) + f_2(a, x_3)$.

Definition 3. $f(x)$ is approximately locally decomposable in a neighbourhood of the subspace $x_1 = a$ if there exists a partition $x = \{x_1, x_2, x_3\}$ and $\delta, \epsilon \geq 0$ such that if $\|b - a\| \leq \delta$ then $|f(b, x_2, x_3) - [f_1(b, x_2) + f_2(b, x_3)]| \leq \epsilon$.

RDIS: Approximate global minimization of a nonconvex function

by dynamic and (R)ecursive (D)ecomposition into locally (I)ndependent (S)ubspaces

Input: Objective function $f(x)$, initial state $\mathbf{x}^0$, approximation error ϵ, subspace optimizer S.
Output: f_{min} such that $|f_{min} - f^*| \leq \epsilon$, where f^* is the global minimum of $f(x)$.

1 Choose variables $\mathbf{x}_c \subseteq \mathbf{x}$

- Choose variables $\mathbf{x}_c$ (giving $\mathbf{x}_u = \mathbf{x} \setminus \mathbf{x}_c$) that induce local decomposition.
- Any heuristic is possible (e.g., VSIDS).
- We use hypergraph partitioning because it ensures decomposition.

2 Choose and set values $\mathbf{x}_c = \mathbf{v}_c$

- Use S to choose $\mathbf{v}_c \leftarrow S(f(\mathbf{x}_c, \mathbf{x} = \mathbf{v}_u))$.
- S could be any nonconvex optimizer, including grid search, EM, or L-BFGS. We use conjugate gradient descent and Levenberg-Marquardt with restarts.
- Remove nodes corresponding to assigned variables from graph.

3 Simplify $f(\mathbf{x}_c = \mathbf{v}_c, \mathbf{x}_u)$

- Set terms (or factors) with narrow bounds to constants (locally).
- Remove edges corresponding to assigned terms (or factors) from graph.

4 Decompose $f(\mathbf{x}_c = \mathbf{v}_c, \mathbf{x}_u)$

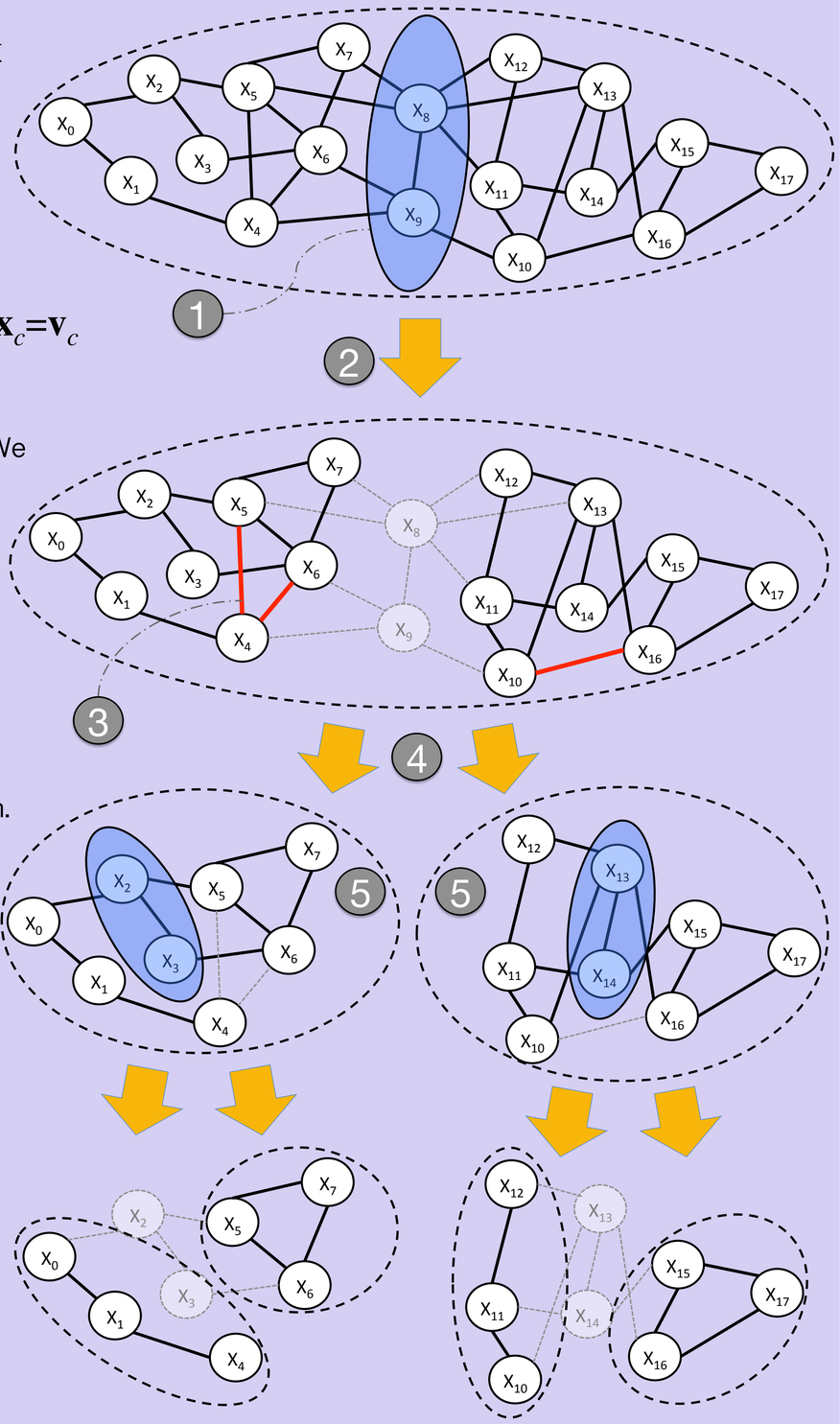
- Divide dependency graph into its connected components.
- Connected components can be optimized independently.

5 Recurse

- Recursively call RDIS on each connected component.
- Globally optimizes $f(\mathbf{x}_c = \mathbf{v}_c, \mathbf{x}_u)$

6 Loop to 2 until done

- Either restart or terminate upon convergence.



Theoretical Results

At each recursion level, let d be the number of variables chosen, $k > 1$ be the number of independent sub-functions the function decomposes into, and $\xi(d)$ be the number of iterations required for the subspace optimizer to find the global minimum of a space of dimension d.

Proposition 1. The time complexity of RDIS is $O\left(\frac{n}{d}\xi(d)^{\log_k(n/d)}\right)$.

Grid search, with S steps per variable, has complexity $O(S^n) = O\left(S^{d(n/d)}\right)$.

Proposition 2. $\mathrm{RDIS}_{\mathrm{GS}}$ has complexity $O\left(\frac{n}{d}S^{d\log_k(n/d)}\right)$, which is exponentially faster than grid search.

If DR is a descent method with restarts, l^n is the volume of the global basin of attraction, and L^n is the volume of the space, then the probability of randomly restarting in the global basin is $(l/L)^n = p^n$ and the expected number of restarts for DR to find the global basin is p^{-n}. If the number of iterations to reach the stationary point of the current basin is τ, then the expected complexity of DR is $O(\tau p^{-n}) = O\left(\tau p^{-d(n/d)}\right)$.

Proposition 3. $\mathrm{RDIS}_{\mathrm{DR}}$ has expected complexity $O\left(\frac{n}{d}(\tau p^{-d})^{\log_k(n/d)}\right)$, which is exponentially faster than DR.

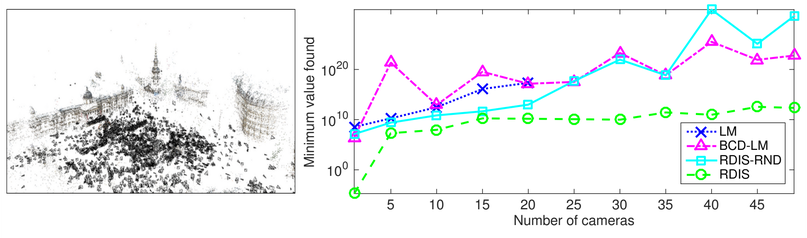
$\mathrm{RDIS}_{\mathrm{DR}}$ behaves like an inexact Gauss-Seidel method, so we can state the following, where $v = l^n$ and $V = L^n$.

Proposition 4. If the subspace optimizer satisfies standard progress conditions and $\epsilon = 0$, then $\mathrm{RDIS}_{\mathrm{DR}}$ returns the global minimum after t restarts with probability $1 - (1 - (v/V))^t$.

Experimental Results

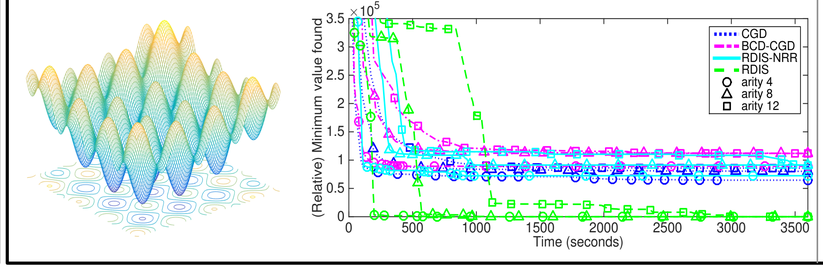
Structure from Motion

Bundle adjustment requires minimizing the squared error between a set of 2-D image points and a projection of fitted 3-D points from a scene's geometry onto fitted camera models.



High-dimensional Sinusoid

A high-dimensional sinusoid in a quadratic basin. Functions with higher arities have more dependencies and are more challenging to optimize.



Protein Folding – Continuous Sidechain Placement

Task is to predict the placement of the protein side-chains when the backbone atoms are fixed. This is equivalent to finding the MAP assignment of a continuous pairwise Markov random field, where the conformations are Boltzmann distributed.