Let's Make Block Coordinate Descent Converge Faster

Julie Nutini, Issam Laradji, Mark Schmidt

Block Coordinate Descent (BCD)

Each **BCD** iteration selects and updates a block of variables.
- Can make similar progress to updating all variables.
- For some problems, updating a block is much faster.

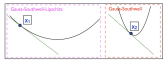
Many choices exist to choose a block and update it

In this work, we provide ways to make BCD converge faster.
- Gives faster algorithm if iteration cost is similar.

Fixed Blocks vs. Variable Blocks

Gauss-Southwell Lipschitz (GSL)

GS: pick block with largest gradient.
GSL: squared gradient over Lipschitz.



Direction of Update and Step Size.

Gradient update
Multiply gradient by a step size of 1/L (Lipshitz smoothness of block).

$$d^k = -\frac{1}{L_{b_k}} \nabla_{b_k} f(x^k)$$

Matrix update
Multiply gradient by upper bound on Hessian block with step size of 1.

$$d^k = -\left(H_{b_k}\right)^{-1} \nabla_{b_k} f(x^k)$$

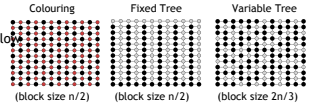
Newton update
Multiply gradient by Hessian and a step size set by backtracking.

$$d^k = -\alpha_k \left(\nabla^2_{b_k b_k} f(x^k)\right)^{-1} \nabla_{b_k} f(x^k)$$

Linear-Time Newton with Tree-Structured Blocks

Cost of Newton is cubic in block size.

But forest-structured dependencies allow linear-time updates.



Bound Constraints and Non-Smooth Regularizers

Gradient updates can be replaced with projected-gradient (cheap).

Newton updates can be replaced with projected-Newton (expensive).

Two-metric projection allows Newton-like updates without extra cost.

Superlinear with greedy rules, large variable blocks, PN/TMP updates.

8 Ways to Make BCD Converge Faster*

1. Use fixed blocks with random rules.
2. Use greedy rules to pick the block instead of cyclic/random.
3. Use variable blocks with greedy rules.
4. Update using Newton's method and a line search.
5. Use Lipschitz information with greedy rules.
6. Increase the block size.
7. Use tree-structured blocks if you have sparse dependencies.
8. Use two-metric projected Newton for non-smooth problems.

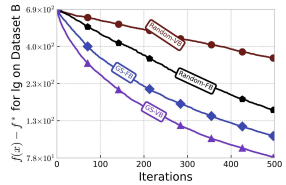
*If it does not significantly increase the iteration cost.

Ask us about superlinear convergence under the right conditions!

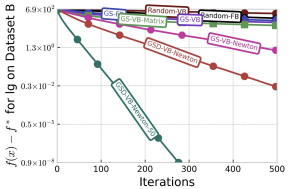
Scan for our codebase

Experimental Results

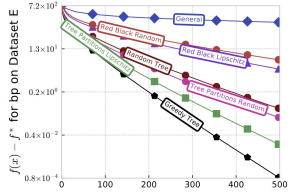
- Random faster with fixed blocks.
- Greedy faster than random.
- Greedy faster with variable blocks.



- Newton updates faster than matrix.
- Greedy+Lipschitz faster than greedy.
- Bigger blocks converge faster.



- Colouring faster than using small blocks.
- Trees converge faster than colouring.



- Projected-Newton faster than gradient.
- TMP similar to Newton with low cost.
- Identifies active set in finite time.