Faster Quasi-Newton Methods for Linear Composition Problems

Betty Shea, Mark Schmidt
University of British Columbia

Linear composition problems

- An objective in the form $f(x) = F(Ax)$
  where f is the composition of linear map Ax and F
- Includes many common objectives

$$f(x) = \sum_{i=1}^{m} \log(1 + \exp(-y_i x^\intercal a_i)) \quad \text{(Logistic regression)}$$

$$f(x) = \sum_{i=1}^{m} \max\{0, 1 - y_i x^\intercal a_i\} + \frac{\lambda}{2}\|x\|^2 \quad \text{(SVM)}$$

$$f(x) = \|Ax - y\|^2 \quad \text{(Least squares)}$$

Quasi-Newton methods

- Approximates Newton's direction by satisfying the secant equation $B_{k+1}s_k = y_k$,
  where $s_k \triangleq x_{k+1} - x_k$, $y_k \triangleq \nabla f(x_{k+1}) - \nabla f(x_k)$ and B_k is positive definite
- BFGS performs a rank 2 update of B_k

$$B_{k+1} = B_k - \frac{B_k s_k s_k^\intercal B_k}{s_k^\intercal B_k s_k} + \frac{y_k y_k^\intercal}{y_k^\intercal s_k}$$

- A limited memory version of BFGS (l-BFGS) stores only a small number of vector pairs s_k and y_k

Wolfe conditions

- Popular inexact step size methods: *Armijo*, *Wolfe*, *or Goldstein*
- Some quasi-Newton methods (incl. BFGS but not l-BFGS) with step sizes satisfying Wolfe conditions have local super-linear convergence
- Wolfe conditions for subspace search

$$f(x_k + P_k\alpha_k) \leq f(x_k) + c_1 \nabla f(x_k)^\intercal (P_k\alpha_k) \quad \text{(sufficient decrease)}$$

$$\nabla f(x_k + P_k\alpha_k)^\intercal (P_k\alpha_k) \geq c_2 \nabla f(x_k)^\intercal (P_k\alpha_k) \quad \text{(curvature)}$$

with parameters $c_1 \in (0, \frac{1}{2})$ and $c_2 \in (c_1, 1)$

TL;DR

- l-BFGS with Wolfe conditions is widely used in practice
- For linear composition problems, adding a momentum term may lead to finding a more accurate solution in less time
- We set step sizes set with inexact subspace optimization

Subspace search

- Uses d > 1 search directions
- Conceptually, one primary direction, many secondary directions
- Primary direction from popular methods
- Secondary directions add favorable properties to method
- For linear composition problems, efficient in number of matrix-vector multiplications (Table 1)

Step size selection	Mat-vec multiplications	Count
Fixed $1/L$	$\nabla f = A^\intercal \nabla F$, $A\nabla f$	2
Line search, e.g. Armijo, Wolfe	as above	2
Plane search with momentum direction	as above	2
Plane search with generic direction	as above and Ap_2	3

Table 1: Number of matrix-vector multiplications per iteration of l-BFGS or GD for objectives of the form $f(x) = F(Ax)$ and second direction p_2

- Store directions as columns of matrix P. On kth iteration

$$x_{k+1} = x_k + P_k\alpha_k$$

  where α_k is a vector of step sizes
- Inexact SO is efficient for linear composition problems

$$f(x_{k+1}) = F(A(x_k + P_k\alpha_k)) = F(Ax_k + AP_k\alpha_k) = f(v_k + P'_k\alpha_k)$$

  where $P'_k = AP_k$ is stored and reused

Practical issues

- Initial step size choice and extrapolating to next trial step size
- Sub-method to use and subproblem accuracy

Experiments

Comparing four methods on a logistic regression objective

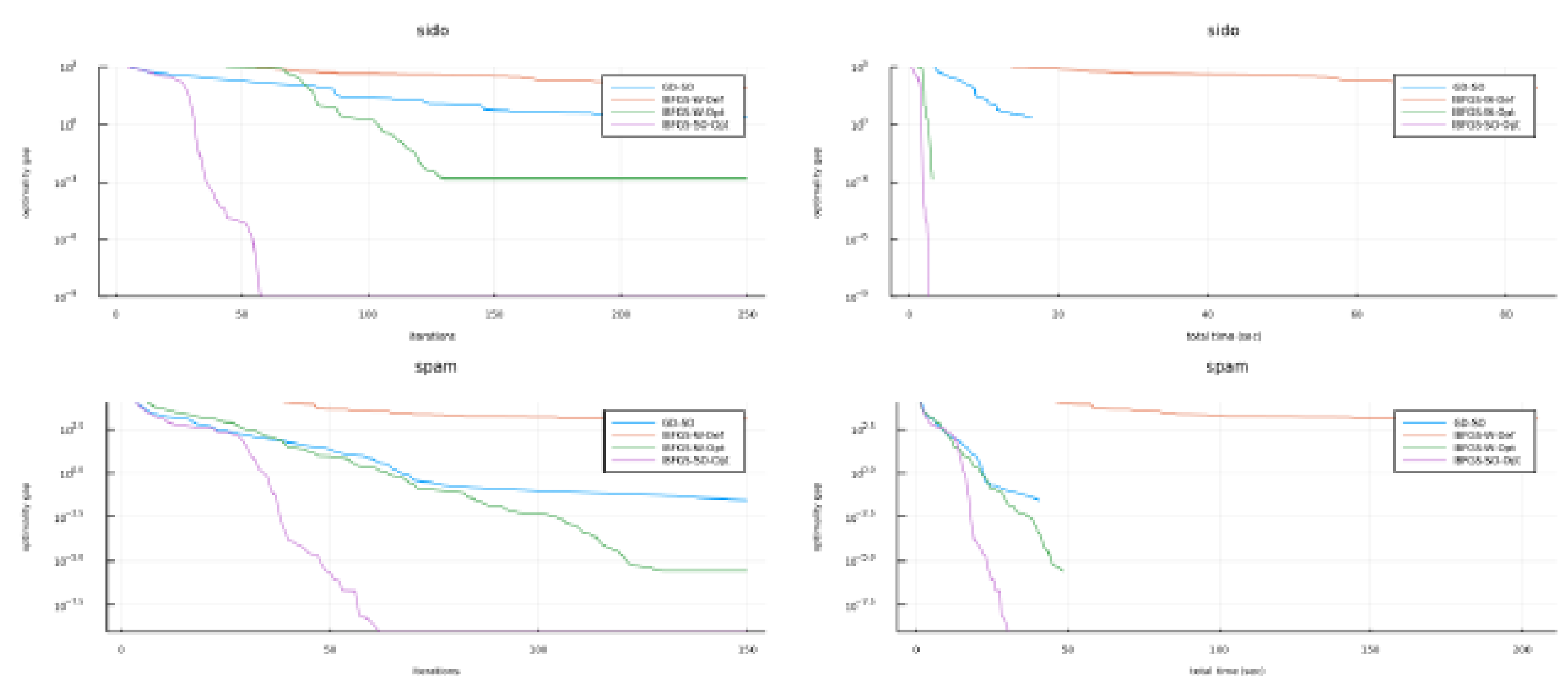

Figure 1: Binary classification of the *sido* and *spam* datasets. Left plot shows accuracy by number of iterations and right plot shows accuracy by time taken in seconds.

sido	Method	$f(x_*)$	Time (sec)	Outer Iters	Inner Iters	Mat-Vec
	GD-SO	2.516832407	21.7	250	21,791	7,423
	l-BFGS-Wolfe-default	75.74420322	85.1	250	1,375	4,054
	l-BFGS-Wolfe-optimized	0.001548940	3.2	130	805	262
	l-BFGS-SO	0.000000017	2.7	57	2,777	1,729
spam	Method	$f(x_*)$	Time (sec)	Outer Iters	Inner Iters	Mat-Vec
	GD-SO	0.030161545	40.6	150	6,695	4,791
	l-BFGS-Wolfe-default	1176.191233	205.1	150	969	2,814
	l-BFGS-Wolfe-optimized	0.000003176	47.0	130	829	262
	l-BFGS-SO	0.000000549	27.8	61	1,973	1,675

Table 2: Binary classification of the *sido* and *spam* datasets.

Takeaways

- Compared to l-BFGS and Wolfe, our method finds a solution that is >5 times more accurate in roughly half the time
- Details such as method used to solve the subproblem matter (we chose Barzilai-Borwein as the submethod)

Key References

[nar2005] Guy Narkiss and Michael Zibulevsky. (2005). Sequential subspace optimization method for large-scale unconstrained problems. *Technical report, Technion – Israel Institute of Technology*.

[sch2005] Mark Schmidt. (2005). MinFunc: Unconstrained differentiable multivariate optimization in Matlab. URL: *https://www.cs.ubc.ca/~schmidtm/Software/minFunc.html*.

compute canada | calcul canada

www.github.com/sheaws/minFuncSO

THE UNIVERSITY OF OF BRITISH COLUMBIA