Filter Flow made Practical: Massively Parallel and Lock-Free

Sathya N. Ravi
University of Wisconsin-Madison
ravi5@wisc.edu

Yunyang Xiong
University of Wisconsin-Madison
yxiong43@wisc.edu

Lopamudra Mukherjee
University of Wisconsin-Whitewater
mukherjl@uww.edu

Vikas Singh
University of Wisconsin-Madison
vsingh@biostat.wisc.edu

1. Proofs of the Lemmas

Lemma 1.0.1. $||A_i||_F^2 \leq C$ *is a valid inequality for a sufficiently large* $C > 0$.

Proof. From Lagrange multiplier theory, there exists a $\lambda_2 > 0$ such that terms in the objective function $\lambda_2 ||A_i\mathbf{i} - \bar{M}_i||_2^2$ is equivalent to adding the following constraints (used in [1]),

$$A_i\mathbf{i} = \bar{M}_i \;\forall i \tag{1}$$

Denote a as the first row of A_i, b as $\mathbf{i}$ and c is the first coordinate of $\bar{M}_i$. We need to show that a is bounded. Now $a^t b = ||a|| ||b|| \cos\theta = c$, since b and c are bounded, it follows that, $||a||_2^2$ is bounded. This gives the result. □

Lemma 1.0.2. *Denote the objective function as* $f(y) : \mathbb{R}^n \to \mathbb{R}$ *where* f *is convex, smooth and that* y *is partitioned into* $\mathcal{J} = \{1, ..., J\}$ *blocks, that is* $y = [y_1, y_2, ..., y_J]$ *such that* $y_i \in Y_j$, *then we have that* $d_i^T \nabla_i f(y^i(t)) \leq -C'||d_i(t)||_2^2$, *where* d_i *is the direction of update i.e.,* $y_i(t+1) = y_i(t) + \gamma_t d_i(t)$ *and* $C' > 0$ *is a constant.*

Proof. Rewriting the update rule of the algorithm, we have that $y_i(t+1) = (1-\gamma_t)y_i(t) + \gamma_t s_i(t) = y_i(t) + \gamma_t\left(s_i(t) - y_i(t)\right)$. Hence we have that $d_i(t) = s_i(t) - y_i(t)$ where $s_i(t) \in \arg\min_s s^T \nabla f(y_i(t))$. We relax the containment operator to a equality since it does not affect the rest of the proof. Now,

$$d_i(t)^T \nabla f(y_i(t)) = \frac{1}{\gamma_t}\left(s_i(t) - y_i(t)\right)^T \nabla f(y_i(t)) \tag{2}$$

$$= \frac{1}{\gamma_t}\left(s_i(t)^T \nabla f(y_i(t)) - y_i(t)^T \nabla f(y_i(t))\right) \tag{3}$$

$$\leq \frac{1}{\gamma_t} s_i(t)^T \nabla f(y_i(t)) < 0 \tag{4}$$

The first inequality is due to the definition of $s_i(t)$, that is, it is the minimizer of the linear subproblem problem $(\#)$ in algorithm 2 whereas the strictly inequality is due to the fact that $s_i(t)$ is a feasible direction and $\gamma_t > 0$. Hence we can choose C' accordingly to get the desired result. □

2. C_f Calculations

We will first mention few basic convergence results for the deterministic CGM that were proved by others for the sake of completeness. We leave it to the reader to check the proof in the references.

Consider the following constrained finite dimensional optimization problem,

$$\min_x f(x) \quad \text{s.t.} \quad x \in \mathcal{C} \tag{5}$$

where f is a differentiable convex function and $\mathcal{C}$ is a compact convex set.

Theorem 2.0.3. *The deterministic CGM (i.e., Algorithm 1) in the main paper with* $\gamma_t = \frac{2}{2+t}$ *satisfies,*

$$f(x_t) - f(x_*) \leq \frac{4C_f}{t+1} \tag{6}$$

Proof. See [2]. □

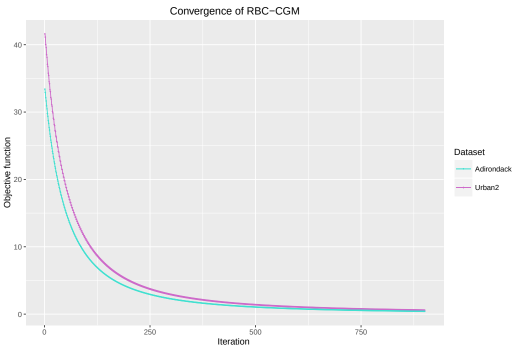
Here C_f is a geometric quantity called as the *curvature constant* that depends on the objective function f, feasible set $\mathcal{C}$ and is defined as,

$$\mathcal{C}_f := \sup_{x,s\in\mathcal{C},\gamma\in[0,1]} \frac{1}{\gamma^2}\left(f\left(x + \gamma\left(s - x\right)\right) - f(x) - \gamma\left\langle s - x, \nabla f(x)\right\rangle\right) \tag{7}$$

Discussion: Intuitively, $\mathcal{C}_f$ measures how close the function is to the linear approximation at $x \in \mathcal{C}$. If C_f is very high, it takes the algorithm many more iterations to converge to a predetermined $\epsilon-$accuracy. For nonsmooth functions, the definition of C_f is tricky, usually the nonsmooth function is approximated by a smooth function (surrogate) before running the algorithm and the analysis is done on the surrogate function.

Since derivative is a linear operator, trivially we see that,

$$C_{\sum f_i} = \sum_i C_{f_i}. \tag{8}$$



(a)

Figure 1: shows convergence of our parallel algorithm for optical flow with two image pairs.

Our objective function is the sum of two types of terms viz data term and smoothness term where each of these two is a sum of terms summed over the pixels. Hence we show how to calculate for one such term and then we can use equation 8 to finish.

To that end, let $f_i := g_i + h_i$ where g_i is the data term associated with pixel i and h_i is the smoothness associated with pixel i. We now show how to calculate C_{f_i} by calculating C_{g_i} and C_{h_i} separately. Since g_i is twice differentiable with $\nabla^2 g_i = I_1 I_1^T$, using Taylor's series approximation and the fact that $\gamma \in [0,1]$, we have that,

$$\begin{aligned} C_{g_i} \leq \sup_{x,s\in\mathcal{C}} ||I_1^T(x-s)||_2^2 &\leq \sup_{x,s\in\mathcal{C}} ||I_1||_2^2 ||x-s||_2^2 \\ &= ||I_1||_2^2 \sup_{x,s\in\mathcal{C}} ||x-s||_2^2 \end{aligned} \tag{9}$$

where the inequality is due to Cauchy-Schwarz inequality. Supremum is nothing but the squared diameter of $\mathcal{C}$, using the fact the $\mathcal{C}$ for the data term is just the probability simplex, we have that,

$$C_{g_i} \leq \sqrt{2}||I_1||_2^2 \tag{10}$$

We can make the bound tighter since the filter size is not the whole image, $C_{g_i} \leq \sqrt{2}||I_1^{\mathcal{N}^i}||_2^2$ where $I_1^{\mathcal{N}^i}$ denotes the coordinates of I_1 that are in the neighborhood $\mathcal{N}^i$ of pixel i.
Similarly, using the fact that $\nabla^2 h_i = |\mathcal{N}^i| I$ where $|\cdot|$ is the cardinality function and I is the identity matrix, we can compute that,

$$C_{h_i} \leq \sqrt{6|\mathcal{N}^i|} \tag{11}$$

where the number 6 appears because of the equivalence of $p-$norms in $\mathbb{R}^n$.

3. κ for initialization

As mentioned in the main paper, we initialization our algorithm using the modified γ_t sequence from theorem 2.0.3 defined as $\gamma_t := \frac{2}{\kappa+2+t}$. κ is defined as (see [3] for details),

$$\kappa = \frac{2C_f}{|B_1 - f(x_1)|} \tag{12}$$

C_f is the quantity computed in the previous, $|B_1 - f(x_1)|$ is Wolfe duality gap and B_1 can be computed as $f(x_1) + \nabla f(x_1)^T(s_1 - x_1)$. In order to get a good estimate of B_1, we randomly initialize $x_1 \in \mathcal{C}$. In our cases we the duality gap to be the sum of duality gaps between each pixel, that is, we randomly pick a feasible point on the probability simplex and initialze T. For problems that require LOCA-M constraints (mentioned in the main paper), we solve a linear program containing 6 variables for each pixel and get A_i such that $A_i\mathbf{i} = \bar{M}_i$. That is,

$$\min_{A} \quad 0 \quad \text{s.t.} \quad A\mathbf{i} = \bar{M}, -C \leq A \leq C \tag{13}$$

Note that the above optimization problem is always from lemma 1.0.1 and we *do not* require a linear programming solver to solve problem 13.

4. More Experiments

Figure 1 shows that the $(1/T)$ convergence rate of our randomized asynchoronous algorithm which coincides with its sequential counterpart as suggested by the theory in the main paper.

References

[1] Seitz, S.M., Baker, S.: Filter flow. In: Computer Vision, 2009 IEEE 12th International Conference on, IEEE (2009) 143–150

[2] Jaggi, M.: Sparse Convex Optimization Methods for Machine Learning. PhD thesis, ETH Zurich (October 2011)

[3] Freund, R.M., Grigas, P.: New analysis and results for the conditional gradient method. (2013)