# Supplement Materials for "The Common-directions Method for Regularized Empirical Risk Minimization"

**Po-Wei Wang** B97058@CSIE.NTU.EDU.TW
*Department of Computer Science*
*National Taiwan University*
*Taipei 106, Taiwan*
**Ching-pei Lee** CHING-PEI@CS.WISC.EDU
*Department of Computer Sciences*
*University of Wisconsin-Madison*
*Madison, WI 53706-1613, USA*
**Chih-Jen Lin** CJLIN@CSIE.NTU.EDU.TW
*Department of Computer Science*
*National Taiwan University*
*Taipei 106, Taiwan*

**Editor:**

## 1. More Experimental Results

In this supplementary document, we provide more experimental results. We first give an additional experiment on adding a bias term in the logistic regression problem, and then show comparison between our method and some representative variance reduction stochastic methods.

### 1.1 Logistic Regression with a Bias Term

In the main paper, we conducted experiments on logistic regression without a bias term. In this section, we provide the results with a bias term in Figures 1-12. In general, the same trend as that in Section 6 is observed.

### 1.2 Comparison with Variance Reduction Stochastic Methods

We compare the proposed method with two representative variance reduction stochastic methods in this section. We implemented the stochastic L-BFGS (SLBFGS) method by Moritz et al. (2016), whose special setting of using zero historical vectors reduces to the so-called SVRG method by Johnson and Zhang (2013). One concern is that most of the data sets we used are relatively sparse, and thus without a careful consideration, a plain implementation will conduct dense vector addition at every iteration, resulting in a much higher cost per data pass than other methods compared in this work. We notice that implementation tricks like those for the plain stochastic gradient on regularized problems

| Data | Training size ($l$) | Features ($n$) | Density (#nnz/($ln$)) |
|---|---|---|---|
| w8a | 49,749 | 300 | 3.8834 % |
| real-sim | 72,309 | 20,958 | 0.2448 % |
| gisette | 6,000 | 5,000 | 99.1000% |

Table 1: Statistics of the additional data sets.

are applicable to SVRG for dealing with this issue, see, for example, the appendix of Reddi et al. (2015). However, Note that this sort of tricks does not apply to Moritz et al. (2016); The L-BFGS update always costs at least $O(n)$ because it either involves multiplying the gradient by an $n \times n$ dense matrix, or takes a two-loop procedure that requires weighted summation of several dense vectors according to their inner products with the current gradient. In summary, the SLBFGS method is practical only when the data set is dense, while SVRG can be made efficient for sparse data if we added some implementation tricks, but a preliminary comparison conducted on relatively dense data sets already suffices for showing their deficiency as we will see below, so the implementation trick is not added.

Another disadvantage of the stochastic methods is that these approaches usually require step size tuning, and for different regularization parameters we might need different step sizes to ensure convergence or fast convergence. Therefore, even if the performance of these methods are good when the parameters are right, it takes a long time to find the right ones, and one should not ignore the parameter tuning time. Therefore, in terms of the total running time, including these parameter tuning parts, these batch methods that need not tune parameters are more preferable in practice.

In regard of the reasons above, we compare with the SLBFGS method by Moritz et al. (2016) on the rather dense data sets epsilon, covtype, a9a, and the additional small data sets w8a, real-sim, and gisette with the default parameters suggested in either their paper or their experiment code at `https://github.com/pcmoritz/slbfgs/`. For the latter three data sets, their statistics are shown in Table 1.

Note that for a fair comparison, we reimplemented their algorithm in C++, following strictly the algorithm described in the paper that possesses a convergence guarantee.[1] We also compared with the special case of this algorithm when it reduces to SVRG, though we acknowledge that the running time for SVRG can be shortened if we use the implementation trick for sparse data mentioned above.

The parameters we used for SLBFGS are: step size: $10^{-4}$, memory size of the L-BFGS matrix: 10, number of samples for minibatch stochastic gradient: 20, number of samples for Hessian estimation: 200, number of stochastic steps before updating the L-BFGS matrix: 10, number of stochastic steps before recomputing the full gradient: $l/20$, where $l$ is the number of data points. Regarding the parameters we used for SVRG, we follow the suggestion of Johnson and Zhang (2013) to use minibatch size being 1, and the number of stochastic steps before recomputing a full gradient being $2l$. For the step size, there was no suggestion in the paper, so we followed the same setting of SLBFGS to use

1. Their experiment code is written in Julia, and to exclude the performance gap resulted from different programming languages, we reimplemented it.

| | $C = 10^{-3}$ | | $C = 1$ | | $C = 10^3$ | |
|---|---|---|---|---|---|---|
| | SLBFGS | SVRG | SLBFGS | SVRG | SLBFGS | SVRG |
| epsilon: logistic regression | $10^{-4}$ | $10^{-4}$ | $10^{-4}$ | $10^{-5}$ | $10^{-4}$ | $10^{-8}$ |
| covtype: logistic regression | $10^{-10}$ | $10^{-10}$ | $10^{-13}$ | $10^{-13}$ | $10^{-16}$ | $10^{-16}$ |
| a9a: logistic regression | $10^{-4}$ | $10^{-4}$ | $10^{-4}$ | $10^{-5}$ | $10^{-4}$ | $10^{-8}$ |
| w8a: Logistic regression | $10^{-4}$ | $10^{-4}$ | $10^{-4}$ | $10^{-4}$ | $10^{-6}$ | $10^{-7}$ |
| real-sim: Logistic regression | $10^{-4}$ | $10^{-4}$ | $10^{-4}$ | $10^{-4}$ | $10^{-4}$ | $10^{-6}$ |
| gisette: Logistic regression | $10^{-4}$ | $10^{-4}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ | $10^{-8}$ |
| epsilon: L2-loss SVM | $10^{-4}$ | $10^{-4}$ | $10^{-5}$ | $10^{-6}$ | $10^{-9}$ | $10^{-9}$ |
| covtype: L2-loss SVM | $10^{-10}$ | $10^{-11}$ | $10^{-13}$ | $10^{-14}$ | $10^{-16}$ | $10^{-17}$ |
| a9a: L2-loss SVM | $10^{-4}$ | $10^{-4}$ | $10^{-5}$ | $10^{-6}$ | $10^{-9}$ | $10^{-9}$ |
| w8a: L2-loss SVM | $10^{-4}$ | $10^{-4}$ | $10^{-4}$ | $10^{-7}$ | $10^{-8}$ | $10^{-10}$ |
| real-sim: L2-loss SVM | $10^{-4}$ | $10^{-4}$ | $10^{-4}$ | $10^{-5}$ | $10^{-6}$ | $10^{-8}$ |
| gisette: L2-loss SVM | $10^{-4}$ | $10^{-4}$ | $10^{-8}$ | $10^{-8}$ | $10^{-11}$ | $10^{-11}$ |

Table 2: Step sizes of SLBFGS and SVRG for different $C$ and different data sets.

$10^{-4}$. When the current step size provides at the tenth iteration an objective value higher than that of the initial objective, we divide the step size by 10 and retain all other settings to rerun the algorithm. This step size tuning procedure is adopted for both SVRG and SLBFGS. The results are shown in Figures 13-17. Note that for the sparse data sets, the running time of SVRG can be improved if a more efficient implementation is used. However, the result of data passes can still provide enough information.

The step sizes we eventually used are shown in Table 2. It is obvious that it takes quite some time to tune the step sizes for these stochastic methods, for example we needed to decrease the step size at least four times on gisette with $C = 10^3$ and more than ten times on covtype with $C = 10^3$. From the figures, we see that although when the problem is rather easy (when $C$ is small), SVRG often outperforms our method, for more difficult problems like covtype, w8a, real-sim, and gisette, especially when $C$ is large, SVRG failed to converge after reaching some low accuracy level. This might be solved partially by further tuning the step size, but this also means that more time needs to be spent. On the other hand, SLBFGS is often the slowest, even in terms of data pass, and the same convergence problem appeared on SVRG is also observed for SLBFGS.

Although it is possible to conduct acceleration for SVRG on problems with larger condition number to partially solve this problem, a difficulty here is the estimation of the Lipschitz constant is required in those accelerated methods.

## References

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323, 2013.

Philipp Moritz, Robert Nishihara, and Michael I. Jordan. A linearly-convergent stochastic L-BFGS algorithm. In *Proceedings of the Nineteenth International Conference on*

*Artificial Intelligence and Statistics*, 2016.

Sashank J. Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alexander J. Smola. On variance reduction in stochastic gradient descent and its asynchronous variants. In *Advances in Neural Information Processing Systems*, pages 2647–2655, 2015.

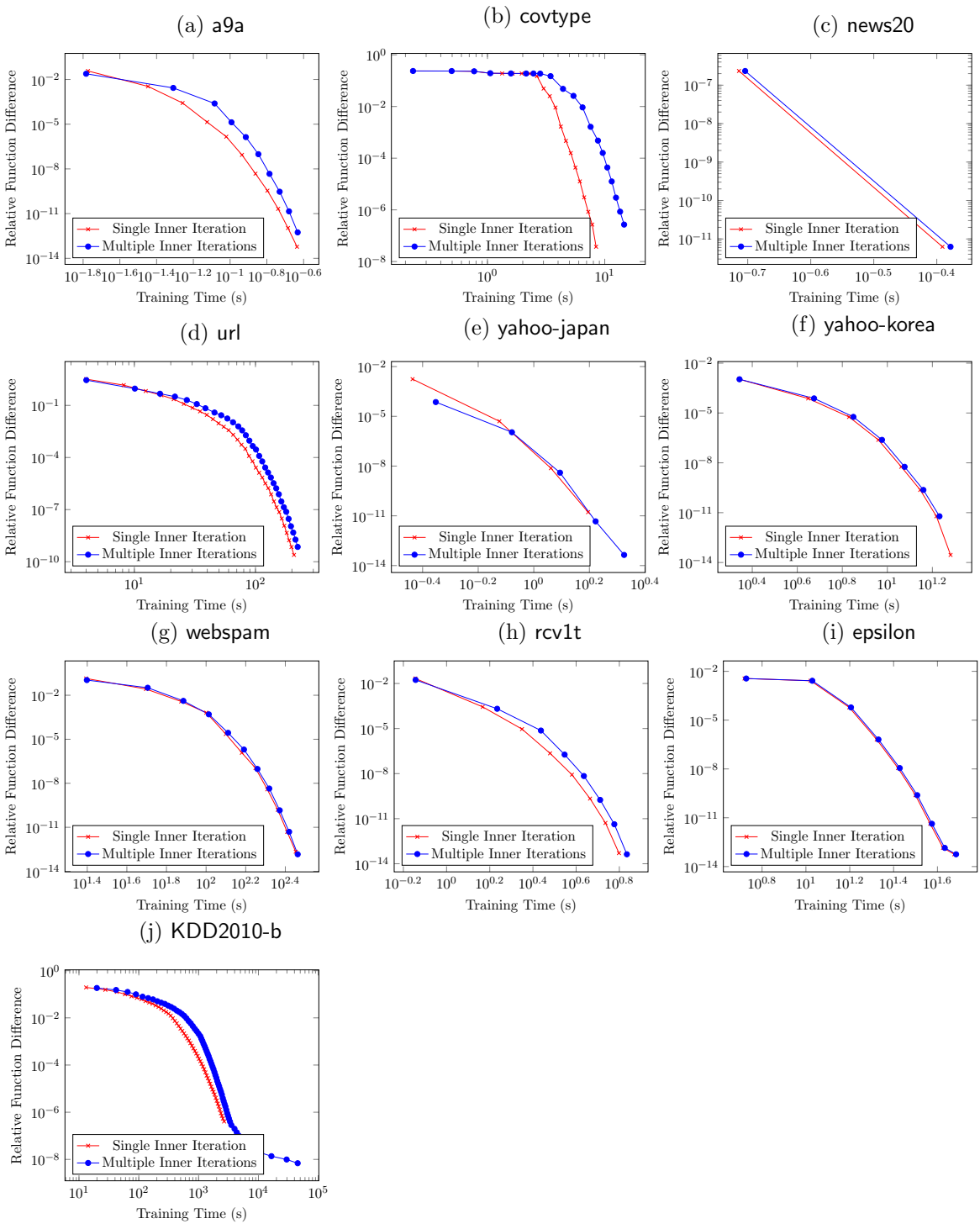


Figure 1: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present training time (in log scale) of logistic regression with a bias term and $C = 10^{-3}$.

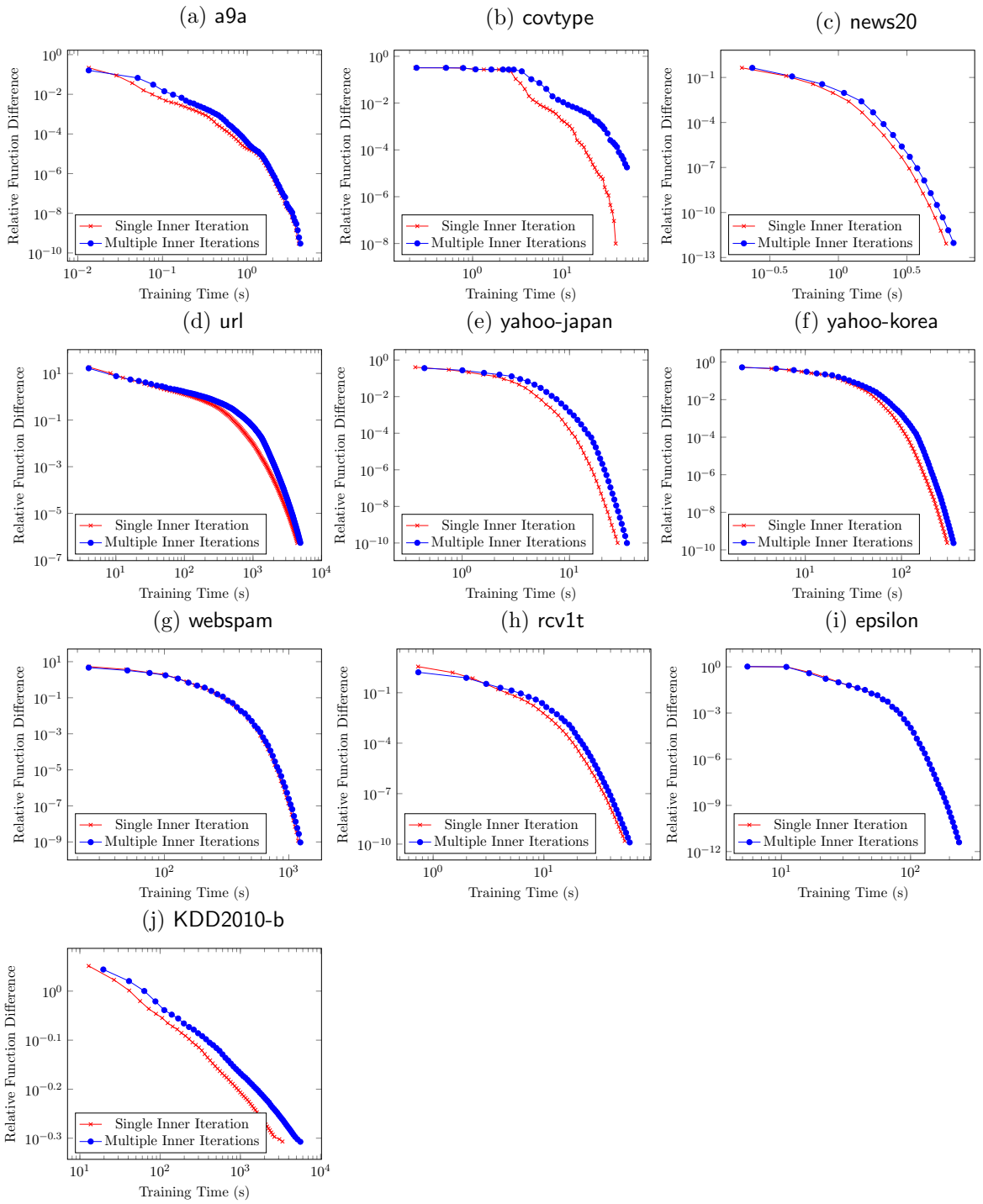


Figure 2: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present training time (in log scale) of logistic regression with a bias term and $C = 10^1$.

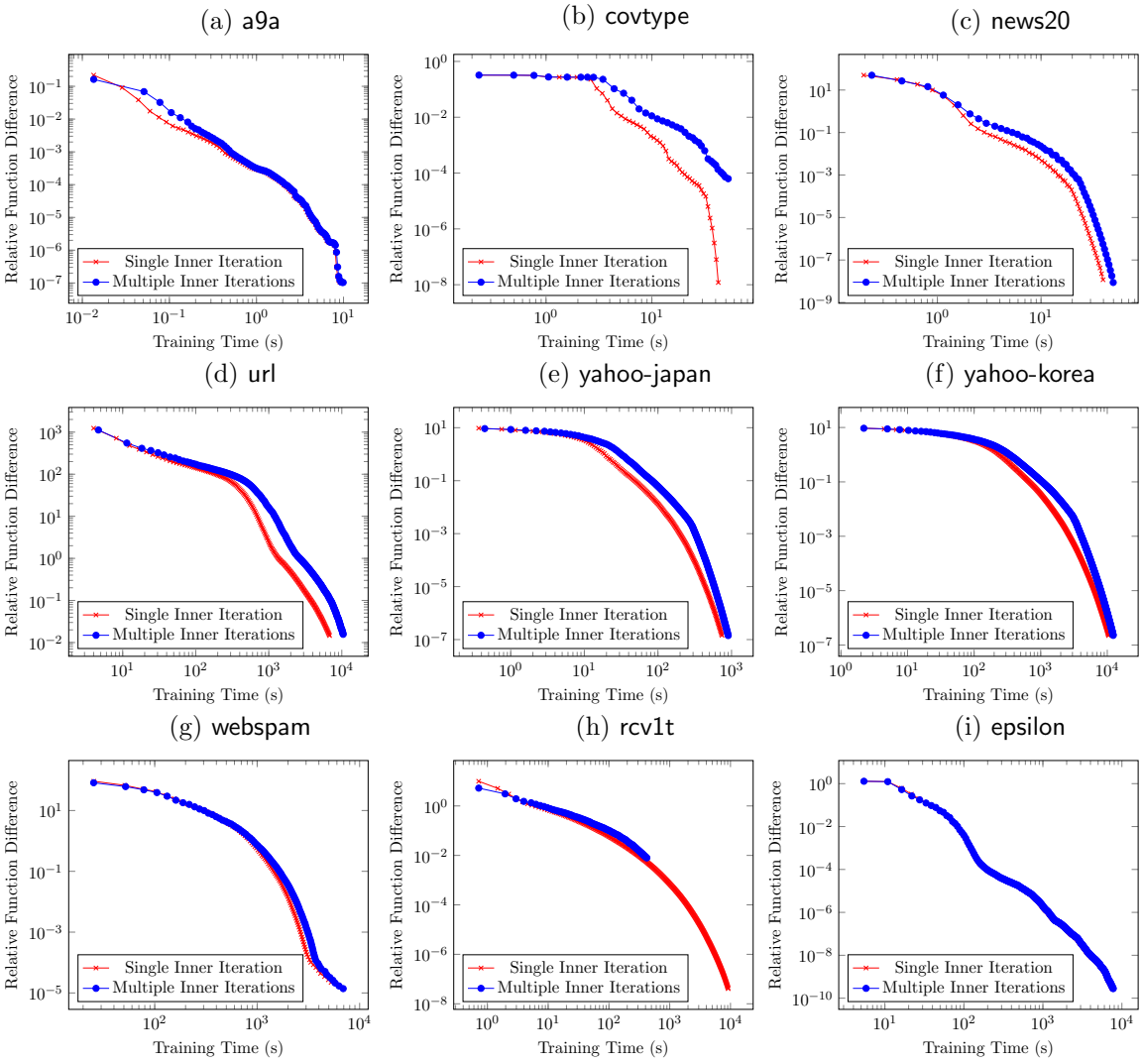


Figure 3: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present training time (in log scale) of logistic regression with a bias term and $C = 10^3$.

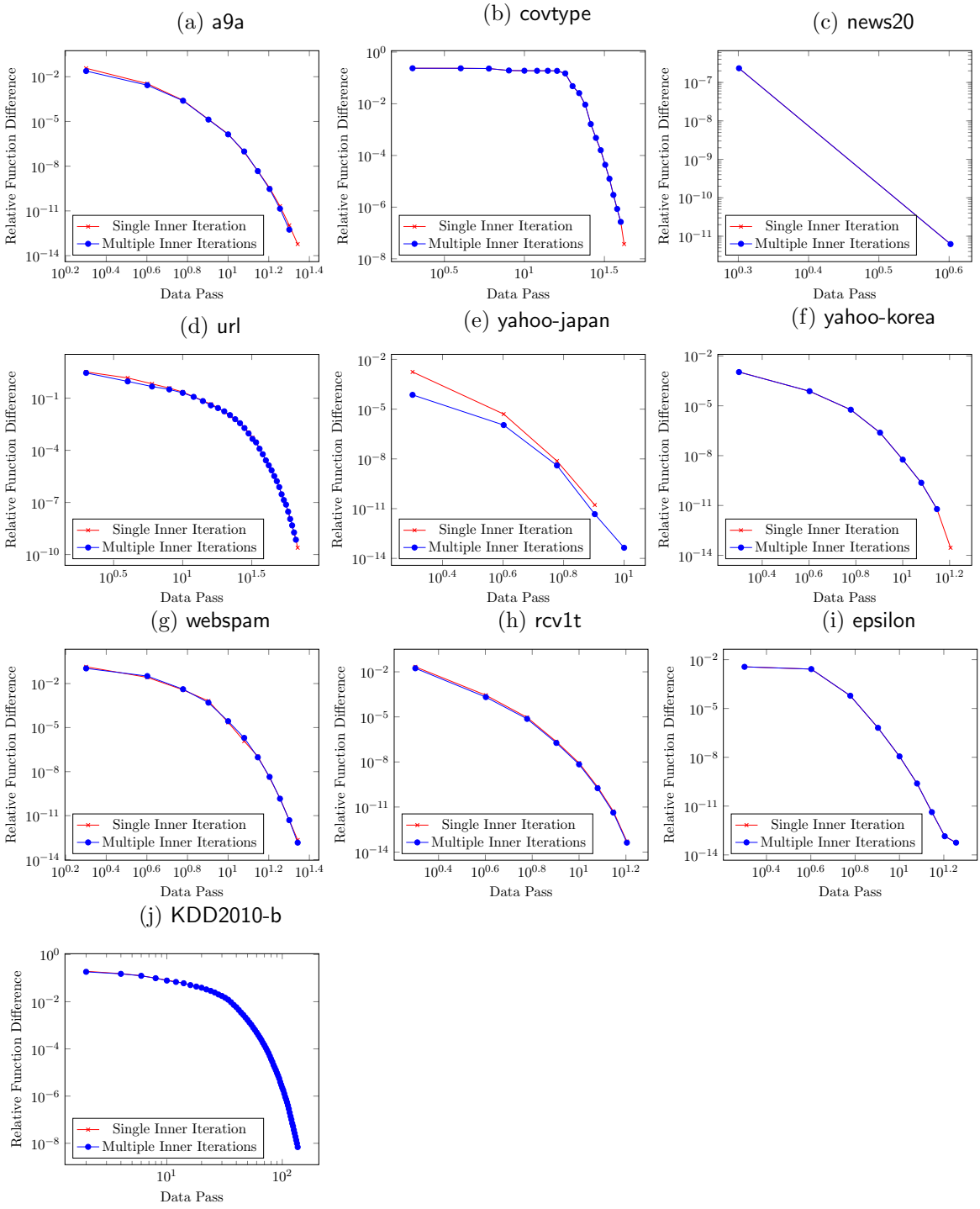


Figure 4: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present data passes(in log scale) of logistic regression with a bias term and $C = 10^{-3}$.

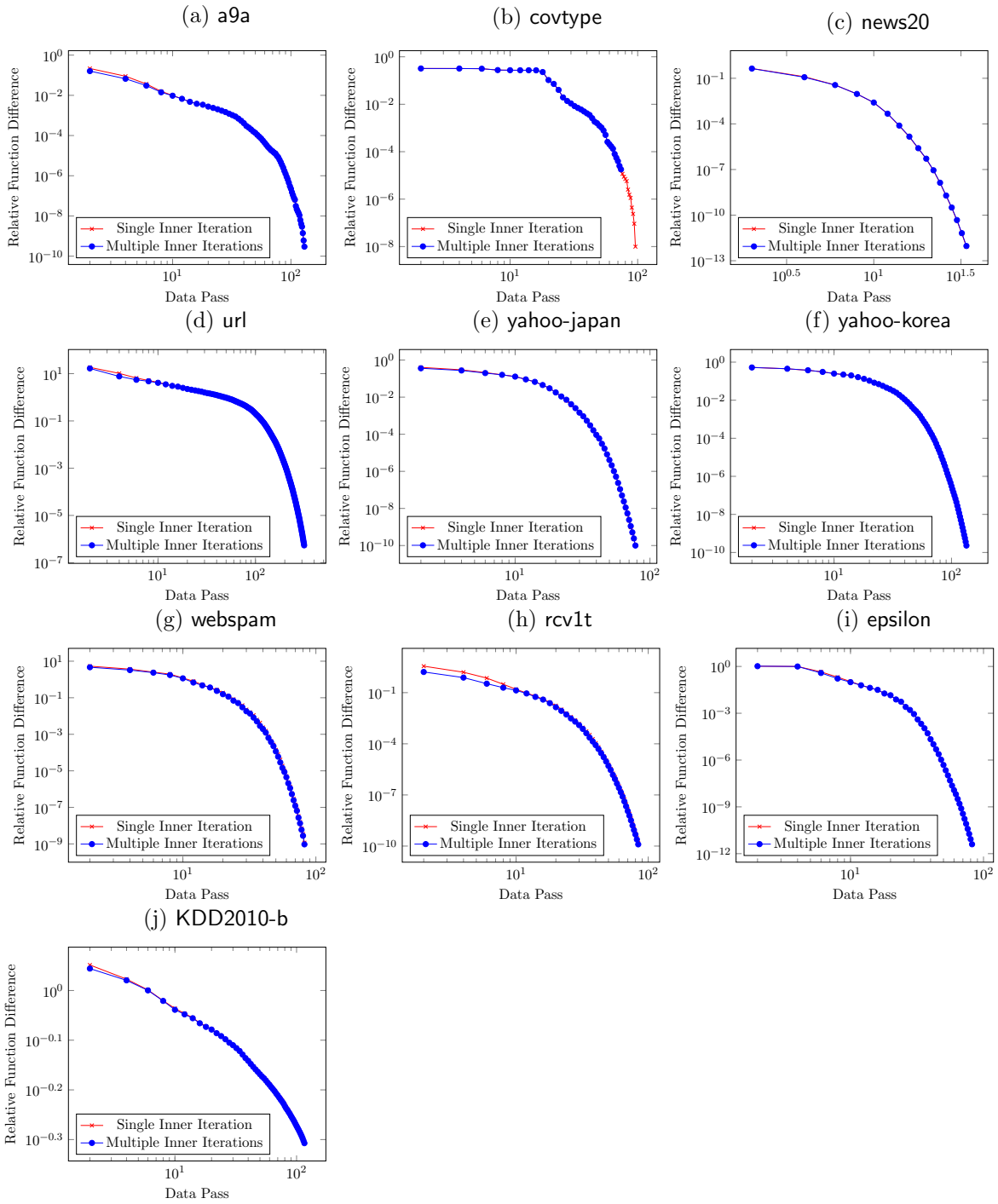


Figure 5: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present data passes (in log scale) of logistic regression with a bias term and $C = 10^1$.

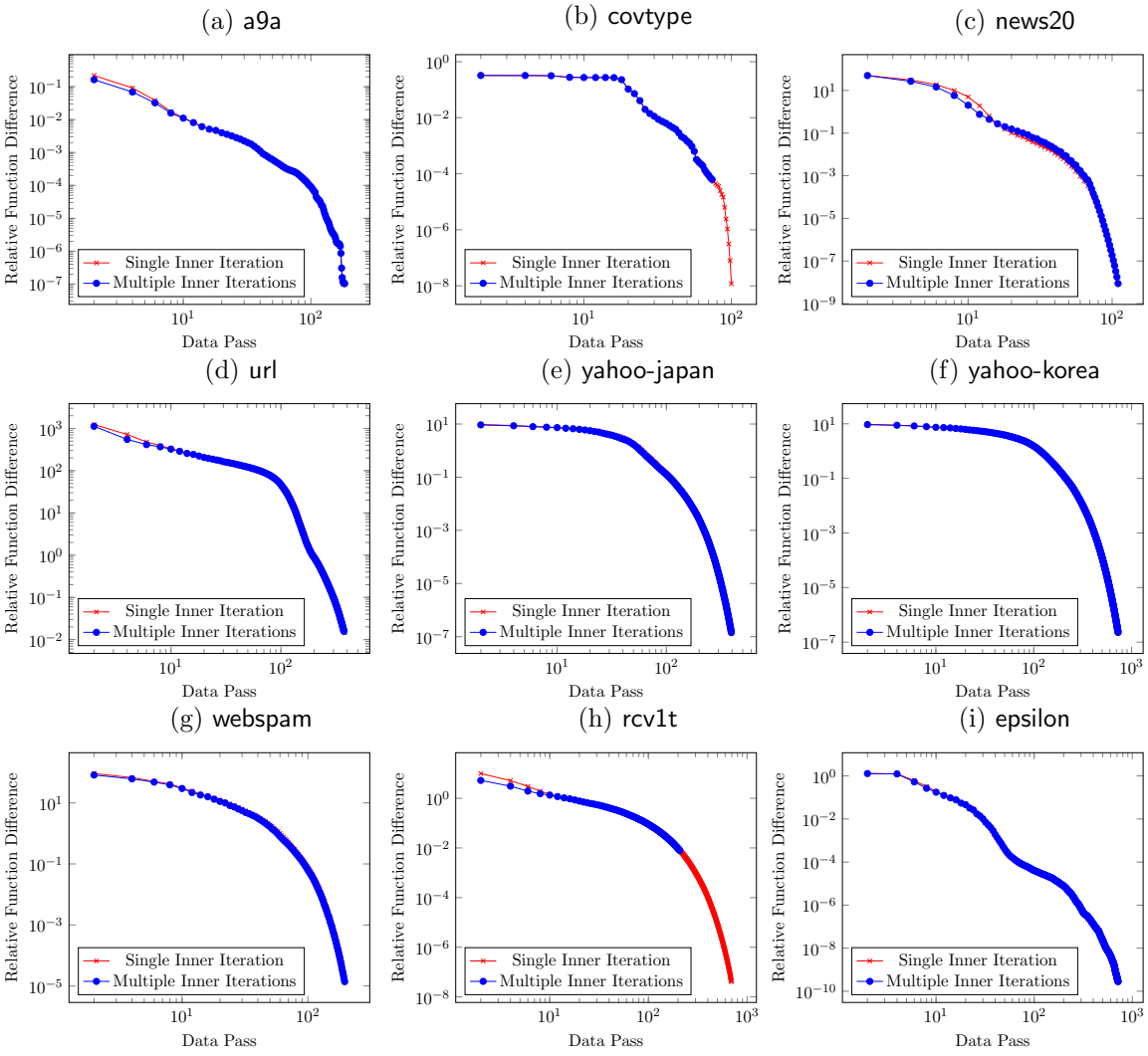


Figure 6: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present data passes (in log scale) of logistic regression with a bias term and $C = 10^3$.

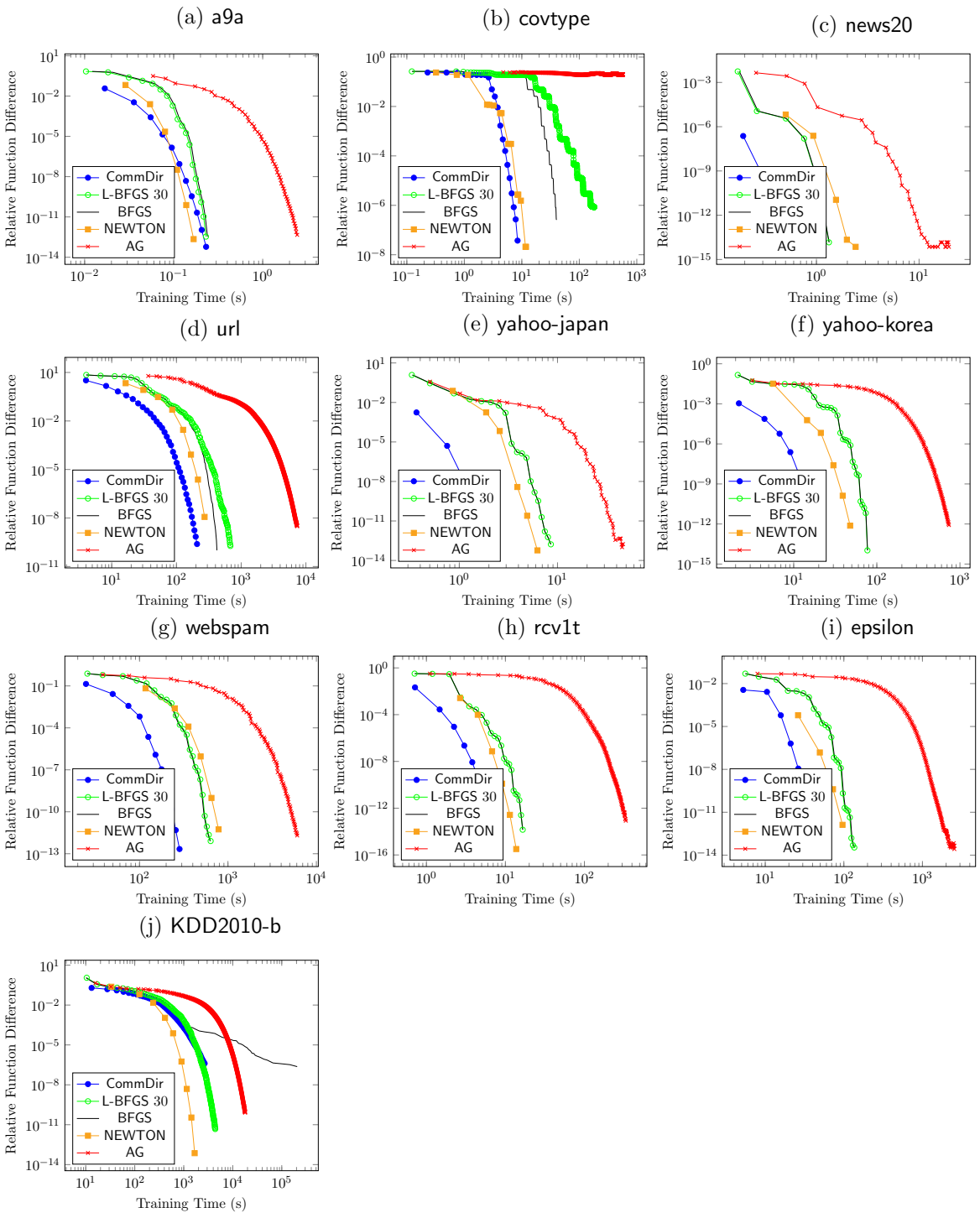


Figure 7: Training time of logistic regression with a bias term and $C = 10^{-3}$.

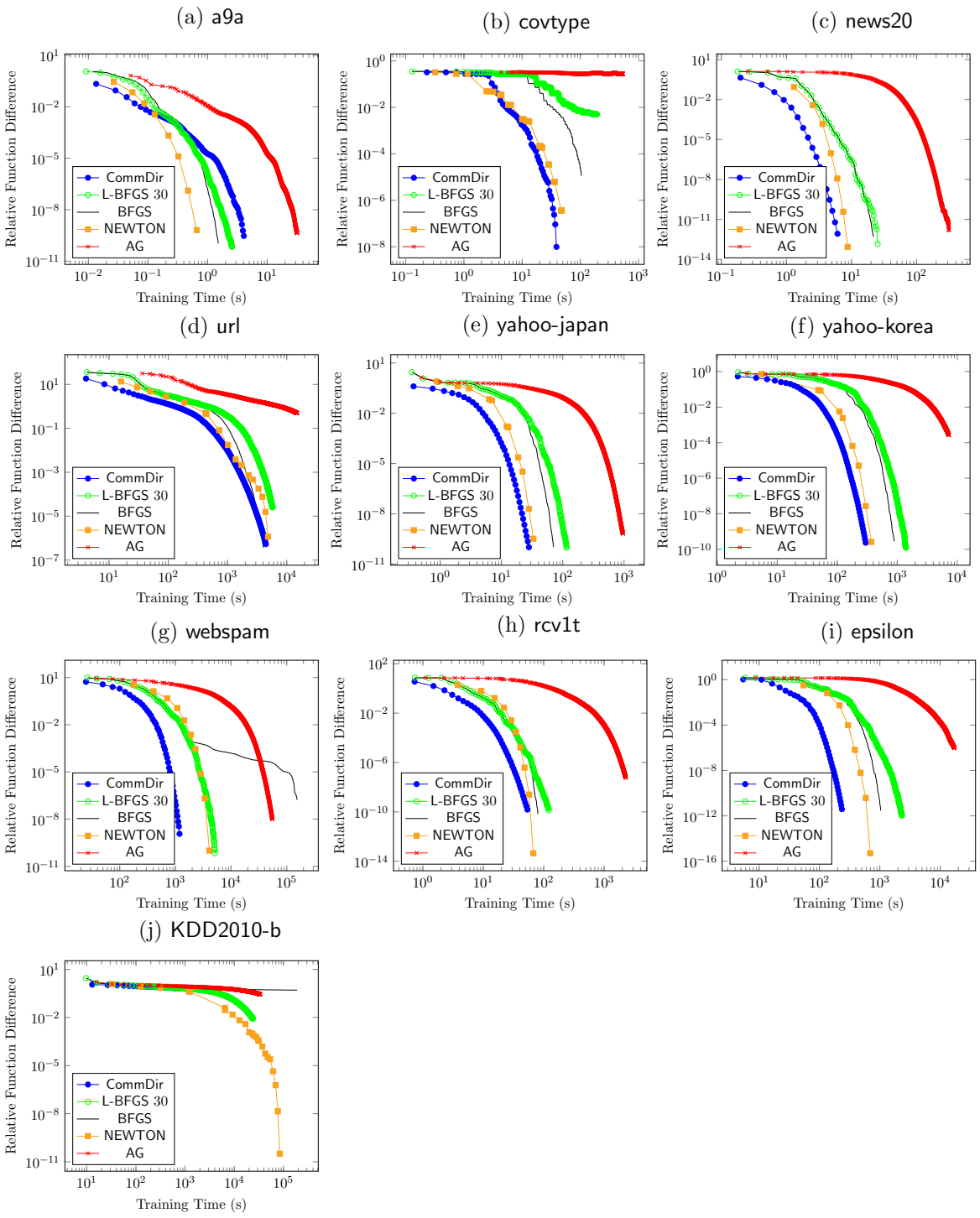


Figure 8: Training time of logistic regression with a bias term and $C = 1$.

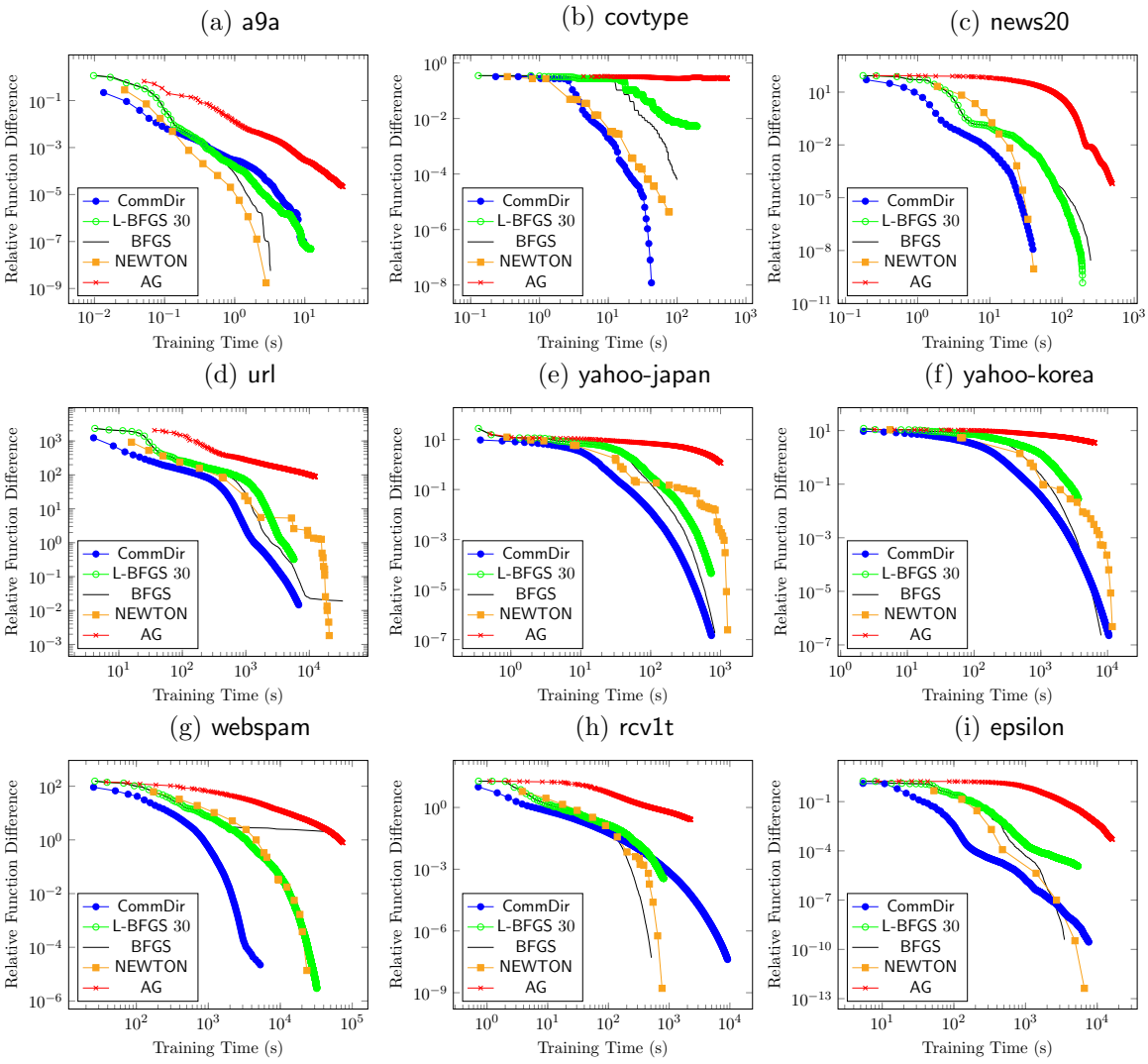


Figure 9: Training time of logistic regression with a bias term and $C = 10^3$.

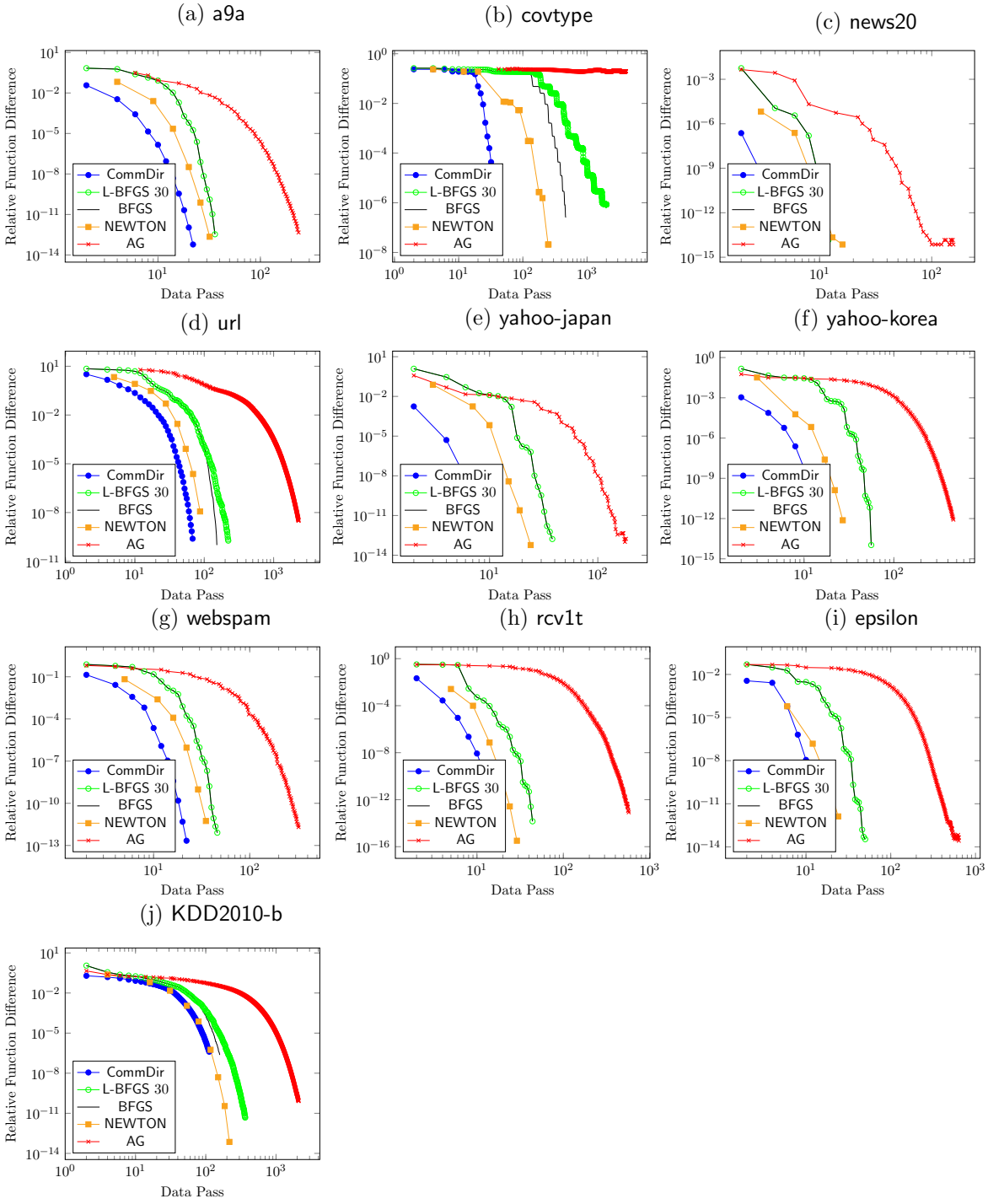


Figure 10: Number of data passes of logistic regression with a bias term and $C = 10^{-3}$.

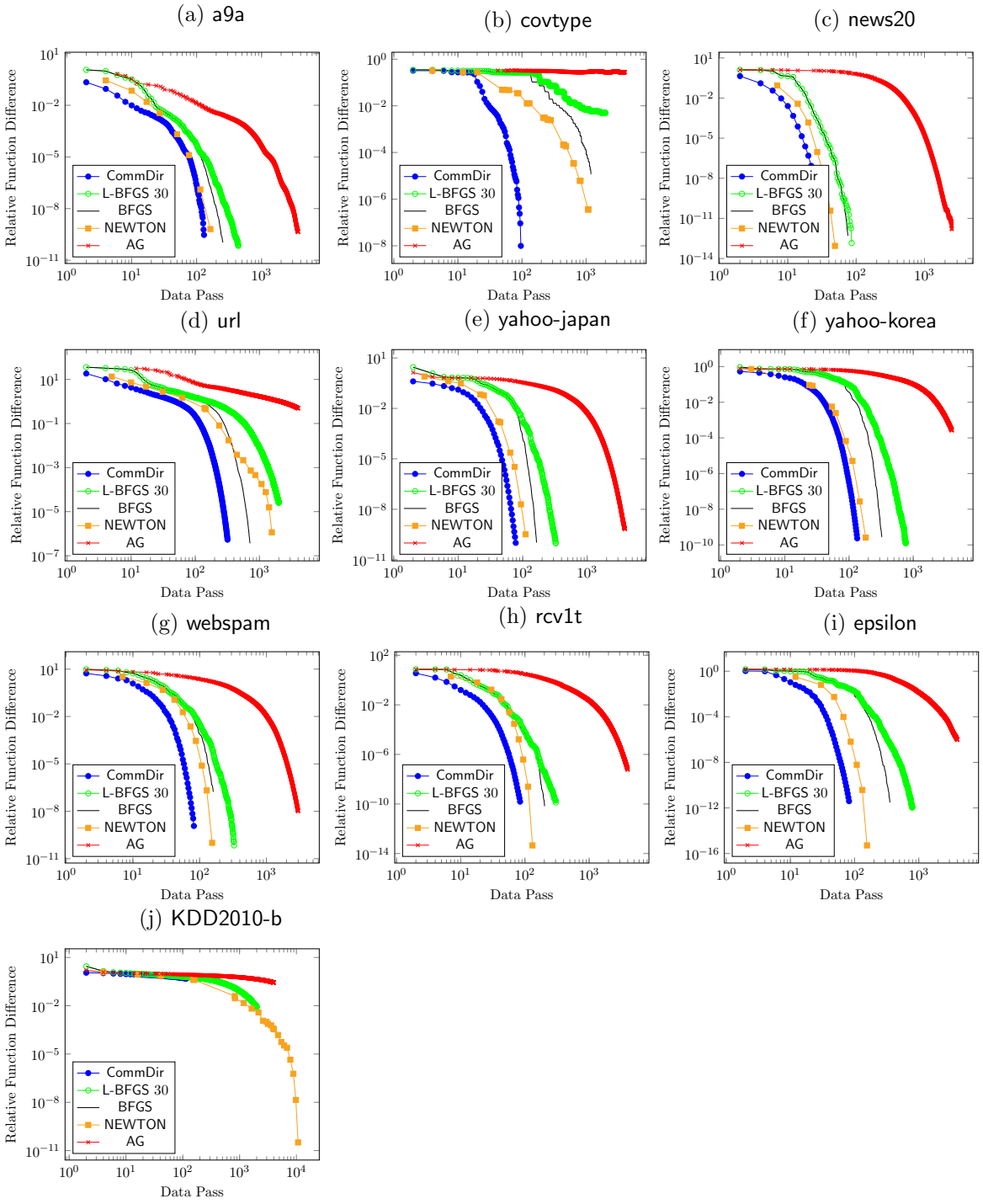


Figure 11: Number of data passes of logistic regression with a bias term and $C = 1$.

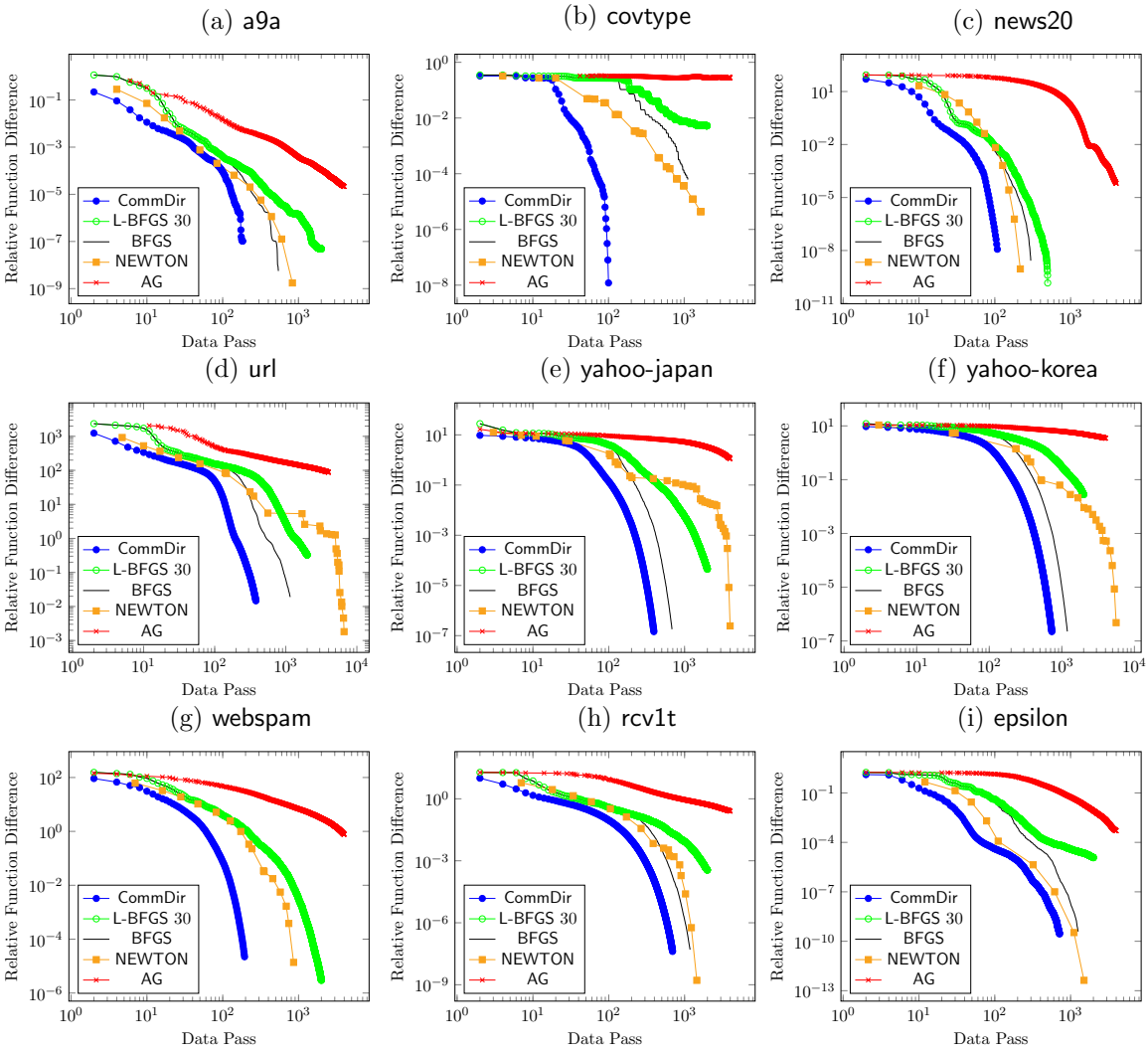


Figure 12: Number of data passes of logistic regression with a bias term and $C = 10^3$.

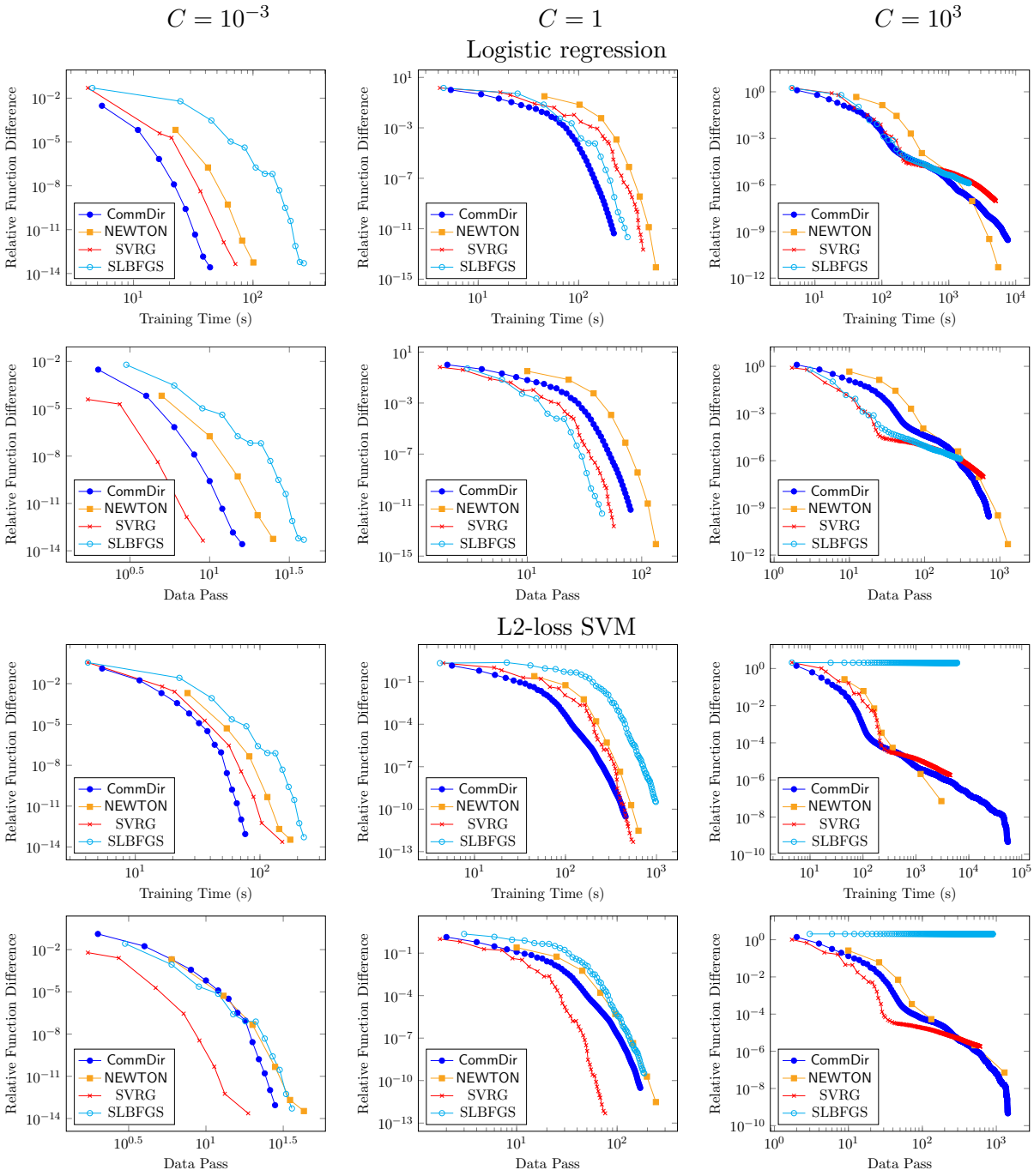


Figure 13: Comparison with the stochastic methods on epsilon.

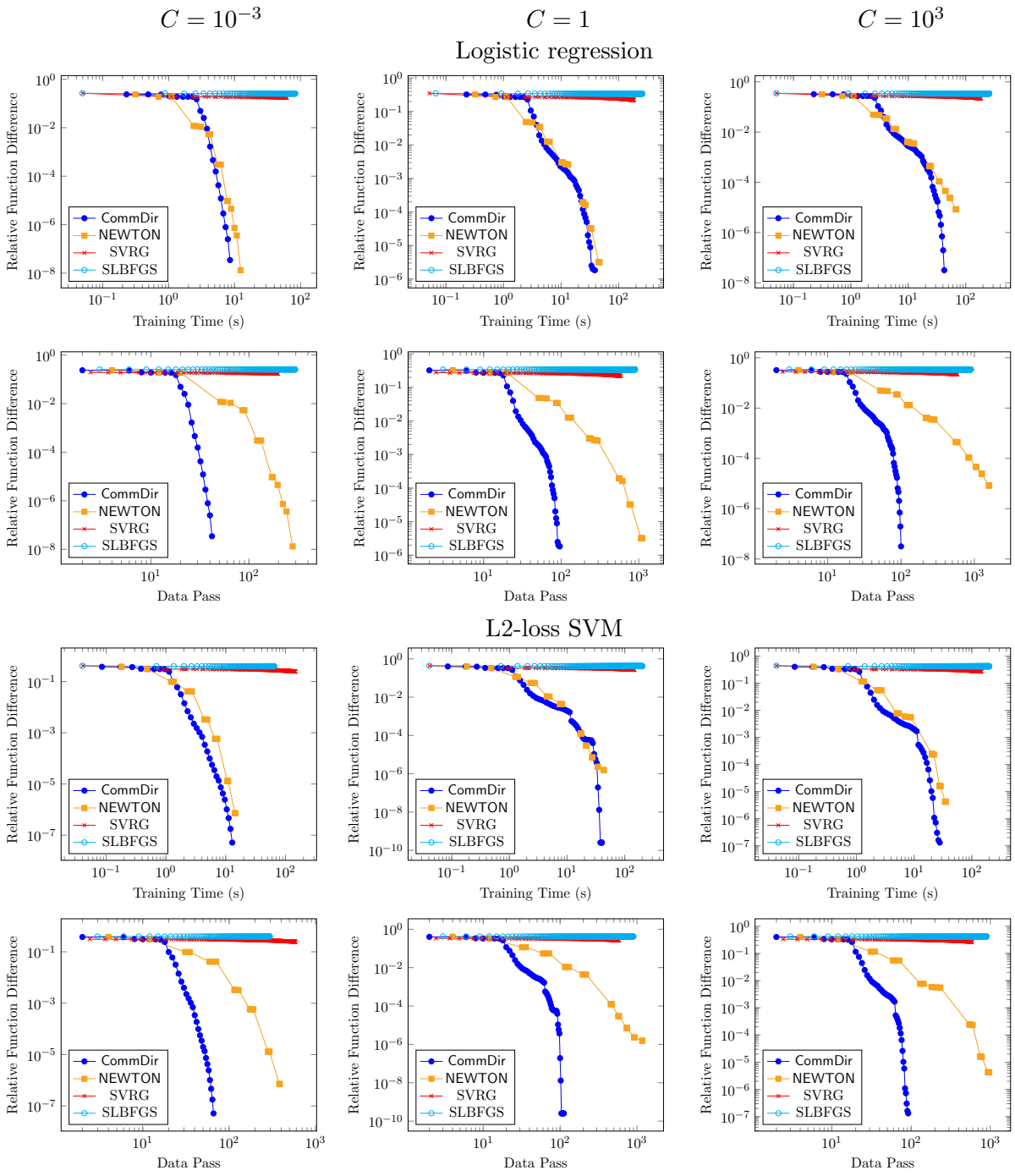


Figure 14: Comparison with the stochastic methods on covtype.

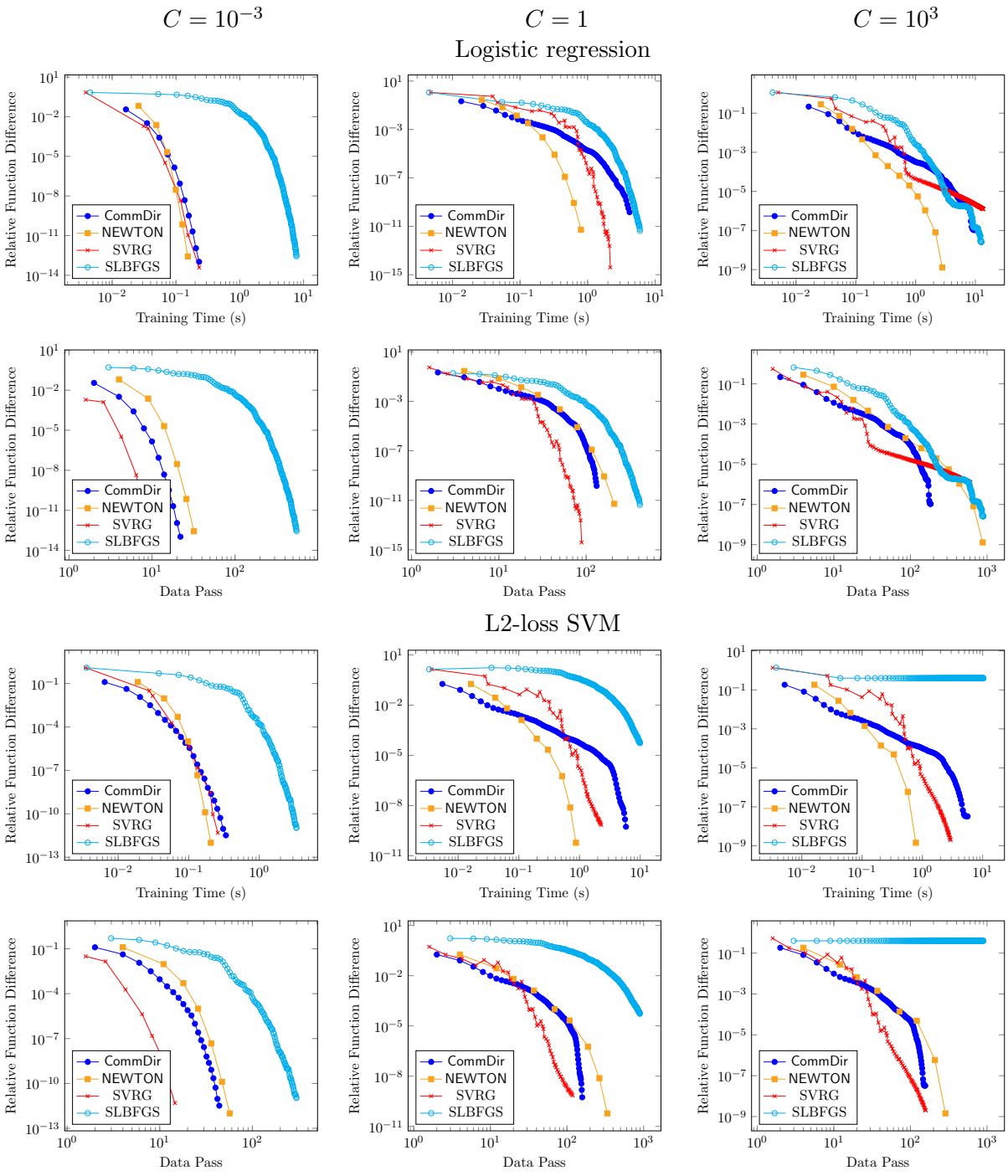


Figure 15: Comparison with the stochastic methods on a9a.

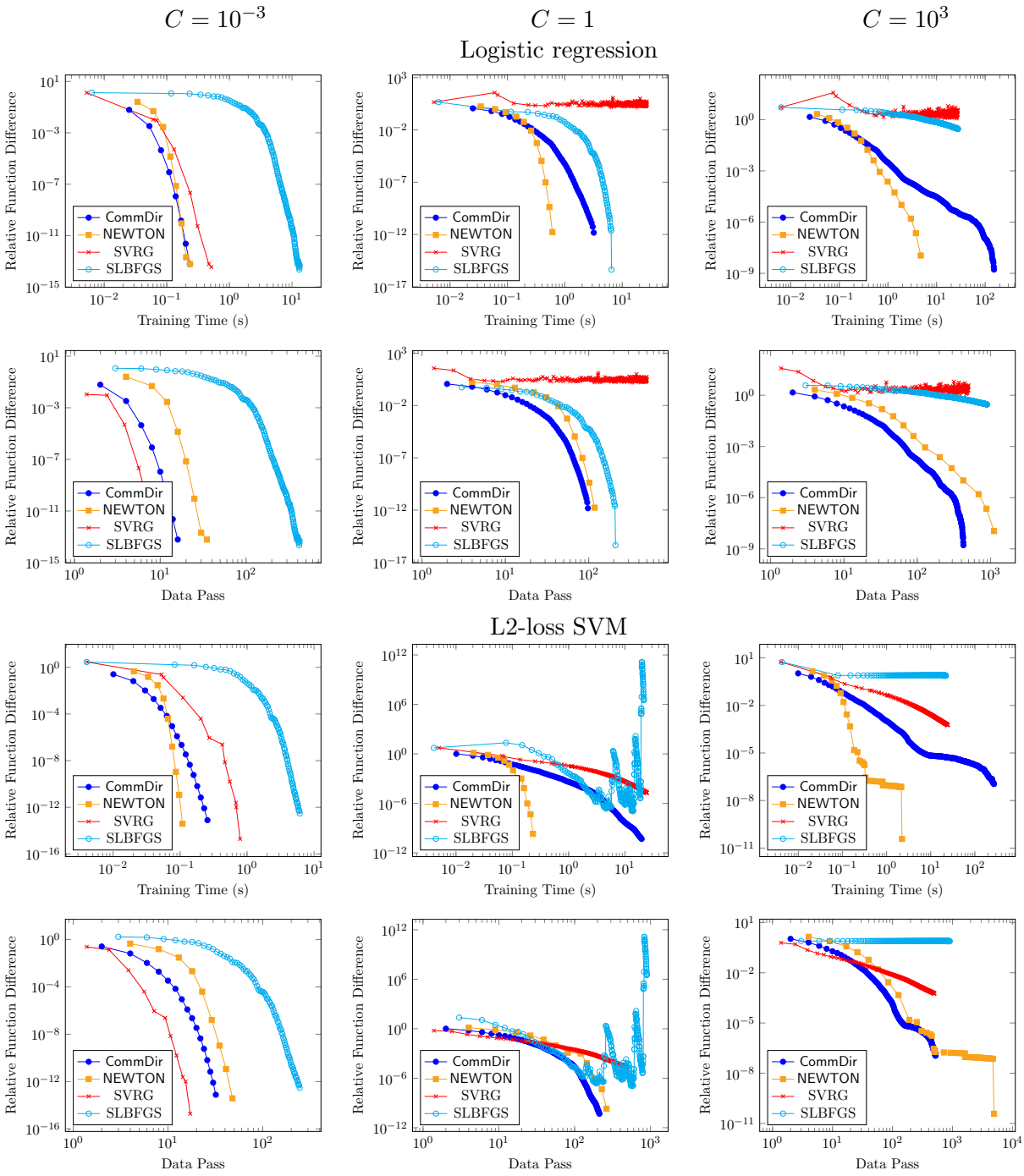


Figure 16: Comparison with the stochastic methods on w8a.

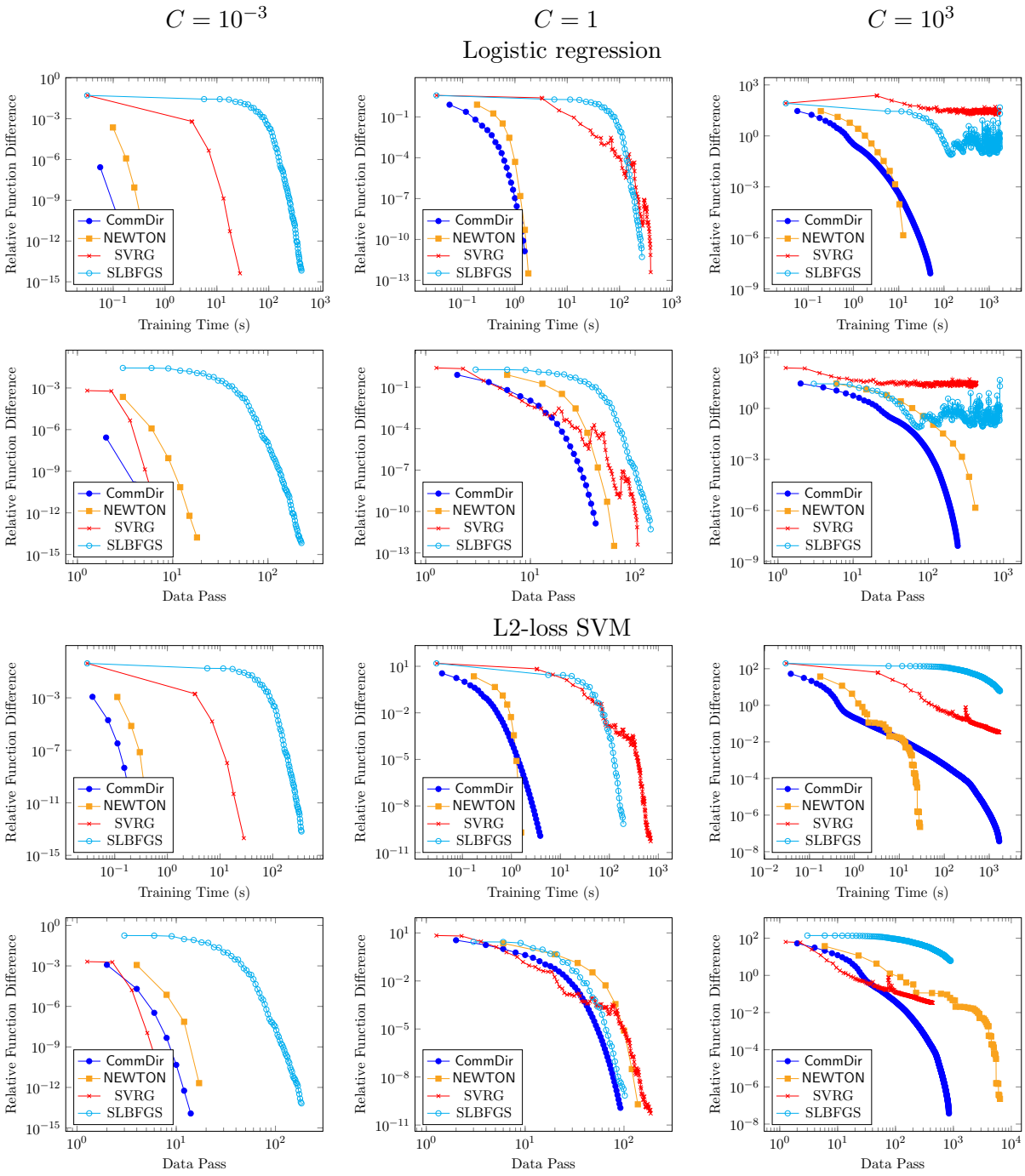


Figure 17: Comparison with the stochastic methods on realsim.

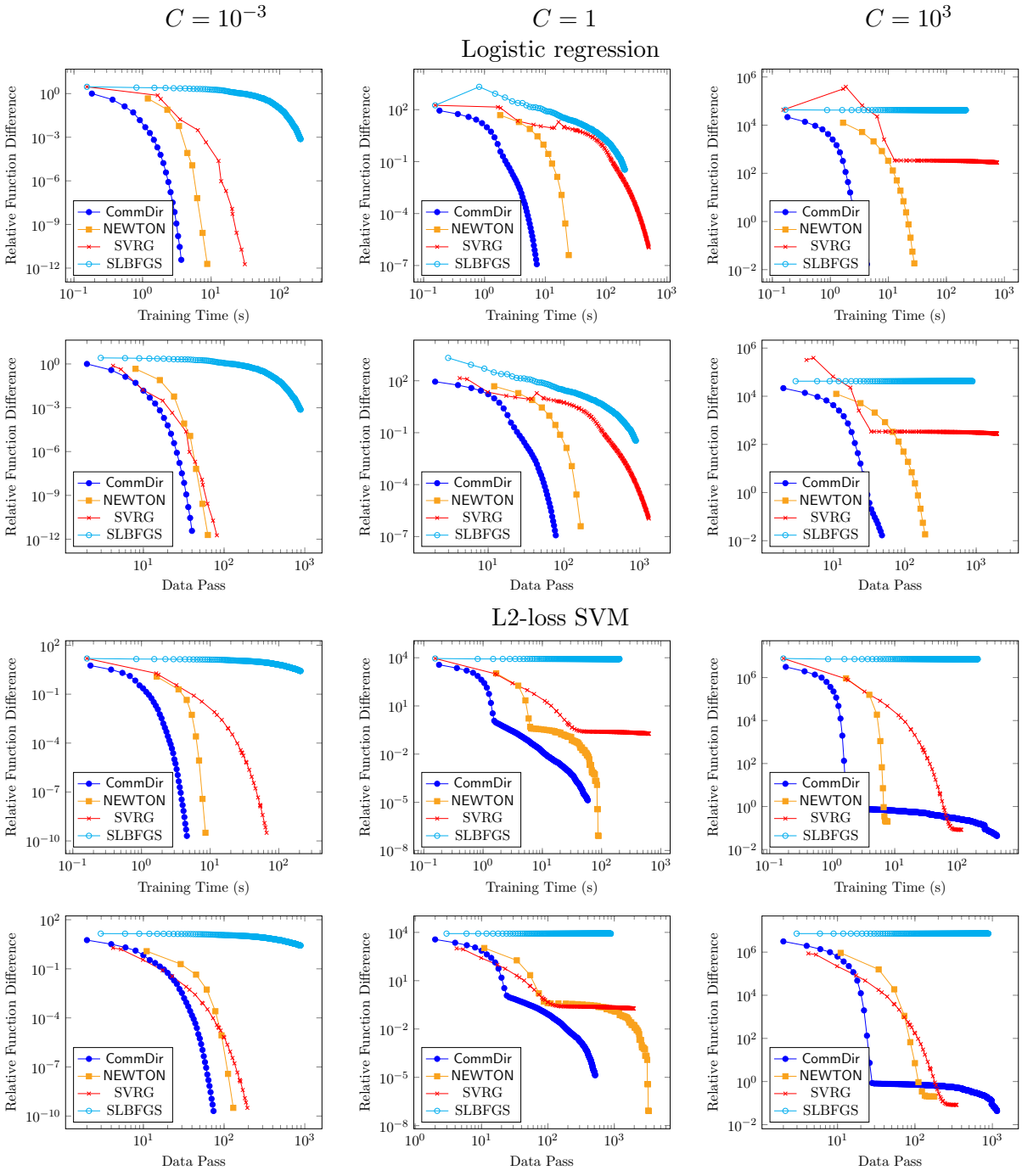


Figure 18: Comparison with the stochastic methods on gisette.