Preconditioned Conjugate Gradient Methods in Truncated Newton Frameworks for Large-scale Linear Classification

Chih-Yang Hsia, Wei-Lin Chiang and Chih-Jen Lin
Department of Computer Science & Information Engineering, National Taiwan University

Linear Classification

- Nowadays linear models has become a mature technique for classification problems
- In many **large-scale** applications (e.g., Ads CTR predictions, document classification), linear models are effective and efficient
- However, for such scale (may be up to billions or more), training time can still take hours even through distributed computing is used
- This work aims to speed up the training of large-scale linear classification from the perspective of **optimization algorithms**

Optimization Problem

- We consider the **linear classification** problem with l instance-label pairs $(\boldsymbol{x}_i, y_i)$ and solve the following optimization problem

$$\min_{\boldsymbol{w}} f(\boldsymbol{w}), \text{where } f(\boldsymbol{w}) \equiv \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{l} \xi(y_i \boldsymbol{w}^T \boldsymbol{x}_i),$$

where C: **regularization parameter**, ξ: loss function
- In this work, we mainly consider logistic (LR) loss

$$\xi_{\text{LR}}(y\boldsymbol{w}^T\boldsymbol{x}) = \log\big(1 + \exp(-y\boldsymbol{w}^T\boldsymbol{x})\big)$$

- To find $\boldsymbol{w}$ efficiently, we need to carefully design the optimization algorithm

Newton's Method

- Newton method is commonly used for large-scale linear classification. It considers the **quadratic approximation** at iterate $\boldsymbol{w}^k$ to find direction $\boldsymbol{s}^k$

$$\min_{\boldsymbol{s}} q_k(\boldsymbol{s}) = \nabla f(\boldsymbol{w}_k)^T\boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^T\nabla^2 f(\boldsymbol{w}_k)\boldsymbol{s} \qquad (1)$$

- The direction $\boldsymbol{s}$ can be obtained by solving the **linear system**

$$\nabla^2 f(\boldsymbol{w}_k)\boldsymbol{s} = -\nabla f(\boldsymbol{w}_k) \qquad (2)$$

- However, $\nabla^2 f(\boldsymbol{w}_k)$ is often too large to be **stored**

$$\nabla^2 f(\boldsymbol{w}_k) \in \mathbb{R}^{n\times n}, n: \text{number of features}$$

- A Hessian-free approach is therefore needed to deal with such situations
- In linear classification, $\nabla^2 f(\boldsymbol{w})$ has a special structure

$$\nabla^2 f(\boldsymbol{w}) = I + CX^T DX$$

where D is a diagonal matrix and $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_l]^T$ is the **data matrix**
- Hessian-vector product can be calculated by

$$\nabla^2 f(\boldsymbol{w})\boldsymbol{v} = (I + CX^T DX)\boldsymbol{v} = \boldsymbol{v} + CX^T(D(X\boldsymbol{v}))$$

Hessian-free Newton Method and Conjugate Gradient (CG)

- Iterative methods such as **conjugate gradient (CG)** can solve (2) by a sequence of matrix-vector products

$$\underbrace{\nabla^2 f(\boldsymbol{w})\boldsymbol{d}_1,\ \nabla^2 f(\boldsymbol{w})\boldsymbol{d}_2, \ldots}_{\#\text{CG steps}}$$

Cost of Newton method $\propto$ **total #CG steps**

- When solving $A\boldsymbol{x} = \boldsymbol{b}$, a **smaller** condition number $\texttt{cond}(A)$ usually leads to **fewer** #CG steps
- **Preconditioning** can possibly improve the condition number of a linear system

Preconditioned Conjugate Gradient (PCG)

Suppose we want to solve $A\boldsymbol{x} = \boldsymbol{b}$.
- PCG finds a preconditioner

$$M = EE^T \approx A$$

and transforms

$$A\boldsymbol{x} = \boldsymbol{b}$$

to

$$(E^{-1}AE^{-T})(E^T\boldsymbol{x}) = E^{-1}\boldsymbol{b}$$

- If the approximation is good, $\texttt{cond}(E^{-1}AE^{-T}) \approx 1$ and fewer #CG steps are needed

Challenges of Applying PCG

- To solve $A\boldsymbol{x} = \boldsymbol{b}$
  - Preconditioning generally reduces #CG steps, **but not always**
  - Applying preconditioning incurs **extra costs**. Fewer #CG steps may not imply less **running time**
- New Challenges of PCG in Newton
  - We now solve **a sequence** of linear systems

$$\text{Newton iteration 1: } \nabla^2 f(\boldsymbol{w}_1)\boldsymbol{s} = -\nabla f(\boldsymbol{w}_1)$$
$$\text{Newton iteration 2: } \nabla^2 f(\boldsymbol{w}_2)\boldsymbol{s} = -\nabla f(\boldsymbol{w}_2)$$
$$\vdots$$

A preconditioner useful for one linear system may not be for another.
Most past PCG studies focus on **one linear system**
  - Now we don't explicitly have $\nabla^2 f(\boldsymbol{w}_k)$. Many existing preconditioners can not be applied

Applying PCG to Newton

- We first consider the diagonal preconditioner, which is possible to get even if Hessian is not formed

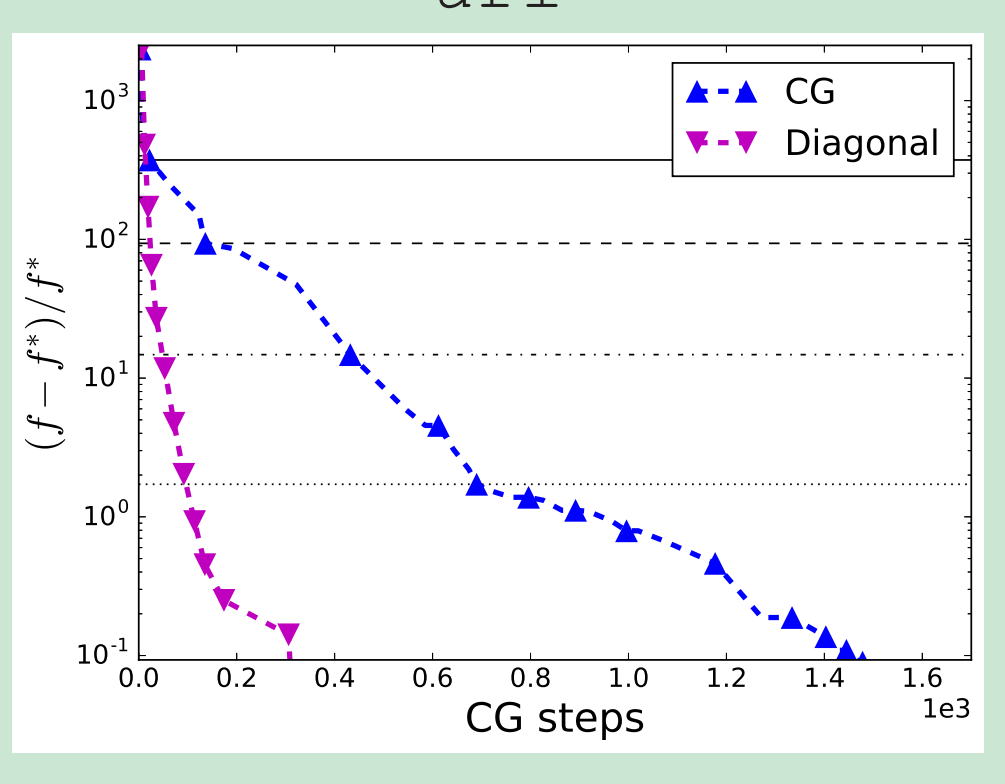
$$M = \text{diag}(\nabla^2 f(\boldsymbol{w})) = \text{diag}(I + CX^T DX) \qquad (3)$$

then we have

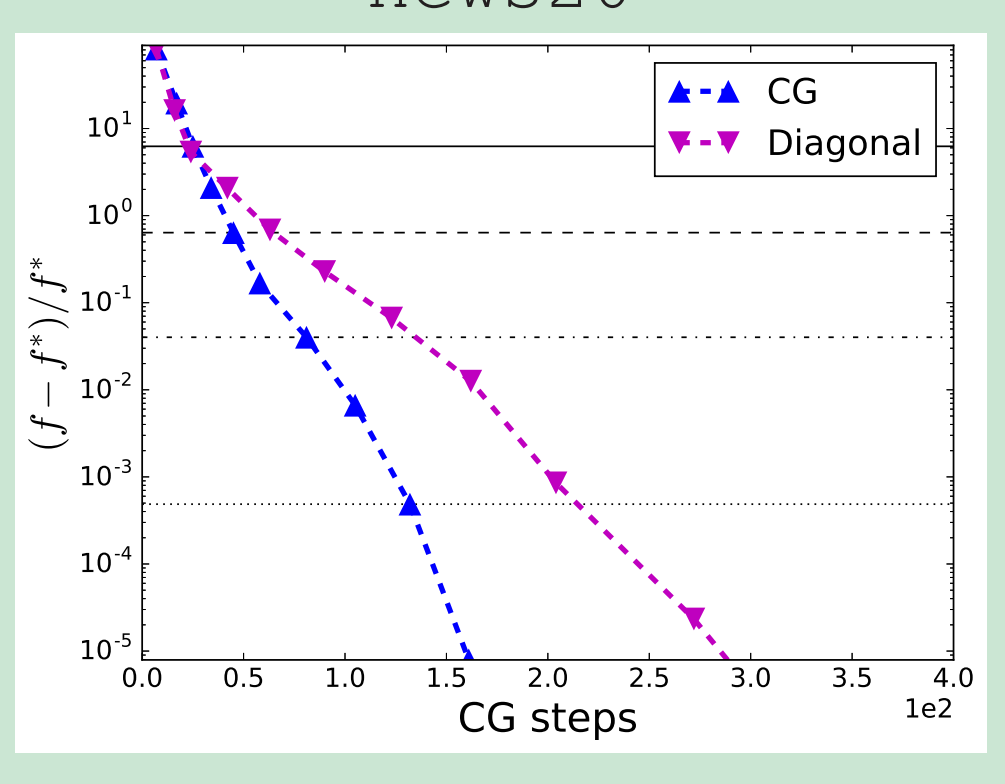
$$M_{ij} = \begin{cases} (\nabla^2 f(\boldsymbol{w}_k))_{jj} = 1 + C\sum_k D_{kk}X_{ki}^2, & \text{if } i = j, \\ 0, & \text{otherwise.} \end{cases}$$

- Examples of using diagonal preconditioner

url



news20



  - Preconditioning can be very useful, **but not always**
  - If something goes wrong in one linear system, **the whole sequence may be bad**

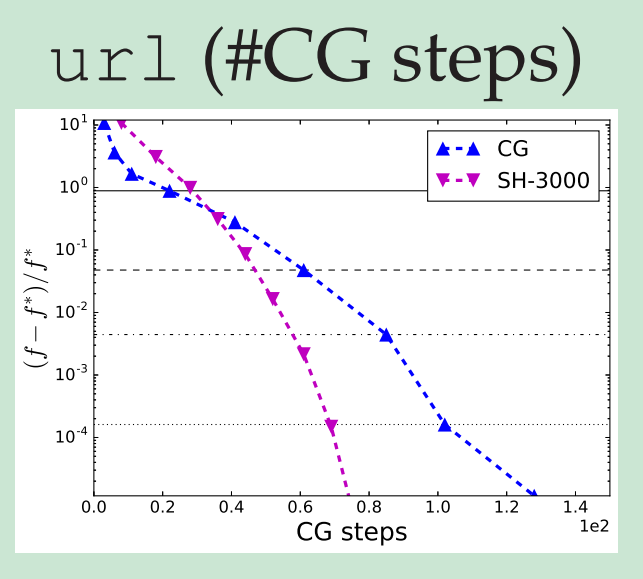
Existing Preconditioner: Sub-sampled Hessian as Preconditioner

- Another idea of approximating the Hessian is by **sub-sampling**
- If we consider a subset of data $\bar{X}$ with $\bar{l}$ instance-label pairs, a **sub-sampled Hessian** can be constructed as a reasonable preconditioner

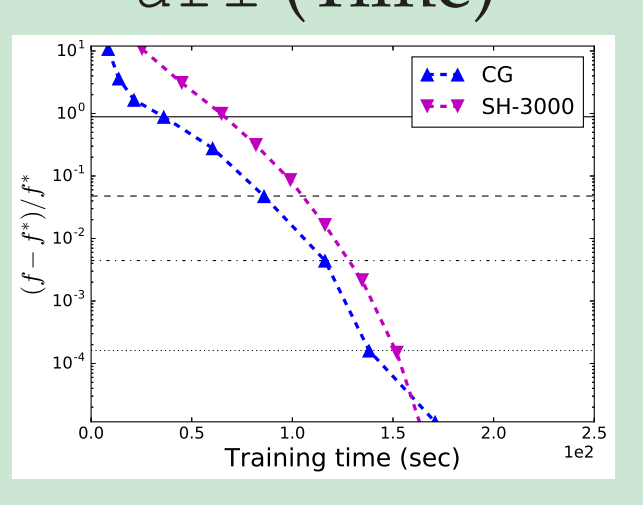
$$M = I + C\frac{l}{\bar{l}}\bar{X}^T\bar{D}\bar{X} \approx \nabla^2 f(\boldsymbol{w}_k)$$

- The preconditioner $M \in \mathbb{R}^{n\times n}$, with the same size as $\nabla^2 f(\boldsymbol{w}_k)$, is too large to be **stored**. ? utilize the special structure of M to make the calculation feasible
- However, the extra costs incurred by this preconditioner can sometimes be **huge**

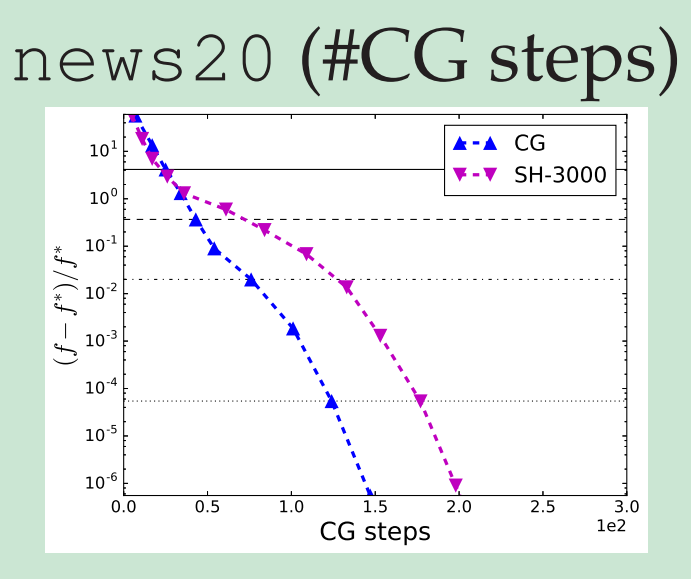
url (#CG steps)



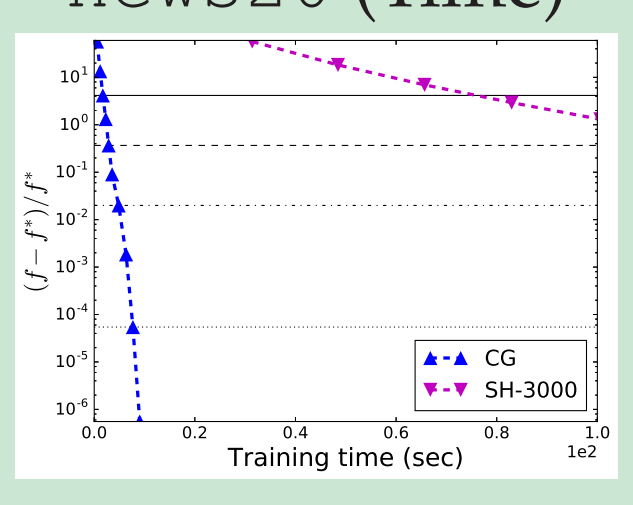
url (Time)



news20 (#CG steps)



news20 (Time)



Making Preconditioner More Robust

- Our work aims to improve the **robustness** of preconditioning for the **overall** procedure
- In Newton method, we solve a series of linear systems

$$A_1\boldsymbol{x} = \boldsymbol{b}_1,\ A_2\boldsymbol{x} = \boldsymbol{b}_2,\ \ldots$$

A preconditioner $M = EE^T$ may work for some linear systems but not others
- Can we **slightly change** M for better robustness of the overall procedure?
- We hope $\bar{M} = \bar{E}\bar{E}^T$ satisfies

$$\texttt{cond}(\bar{E}^{-1}\nabla^2 f(\boldsymbol{w})\bar{E}^{-T}) \approx \min\{\texttt{cond}(\nabla^2 f(\boldsymbol{w})), \texttt{cond}(E^{-1}\nabla^2 f(\boldsymbol{w})E^{-T})\} \qquad (4)$$

That is, we choose the better between with and without the preconditioner

Our Proposed Methods

- Run in Parallel
A direct way to achieve (4) is by **running standard CG and PCG in parallel** and choose the one with fewer CG steps.

$$\bar{M} = \begin{cases} I, & \text{if CG uses less steps,} \\ M, & \text{if PCG uses less steps.} \end{cases}$$

- Weighted Avarage
Parallelizaion may not be always possible. We revise the goal to be more modest

$$\kappa(\bar{E}^{-1}\nabla^2 f(\boldsymbol{w})\bar{E}^{-T}) < \max\{\kappa(\nabla^2 f(\boldsymbol{w})), \kappa(E^{-1}\nabla^2 f(\boldsymbol{w})E^{-T})\} \qquad (5)$$

That is, we avoid the worse one. We prove

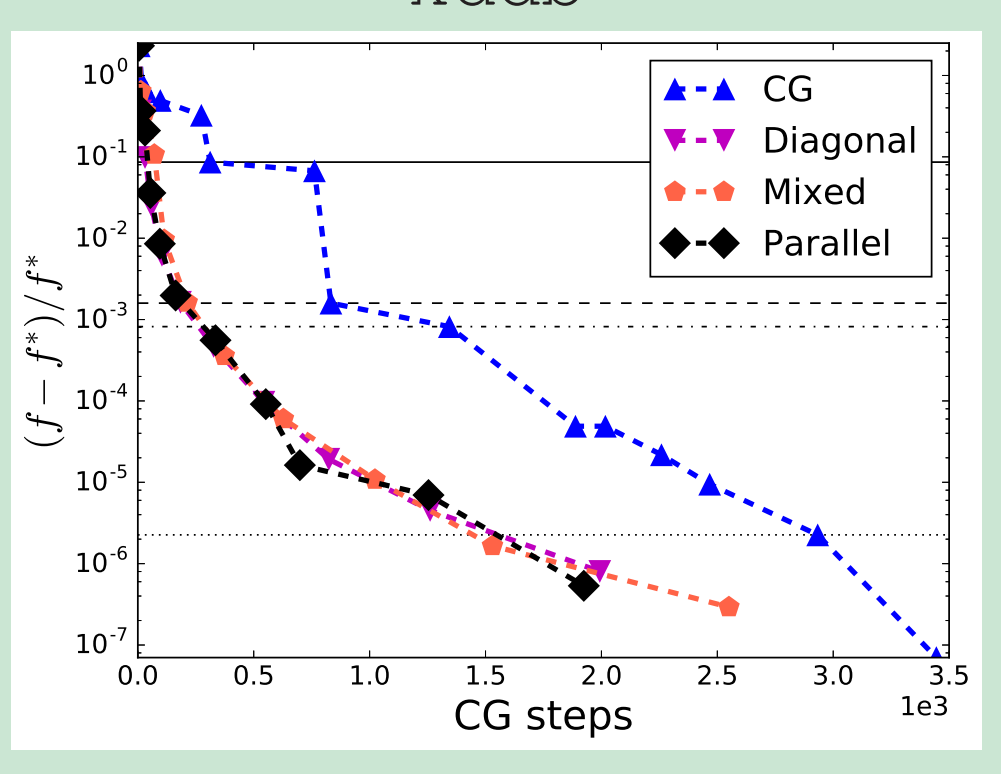
$$\bar{M} = \alpha M + (1-\alpha)I, \text{ where } 0 < \alpha < 1,$$

satisfies (5).
- If M is the diagonal preconditioner, we have

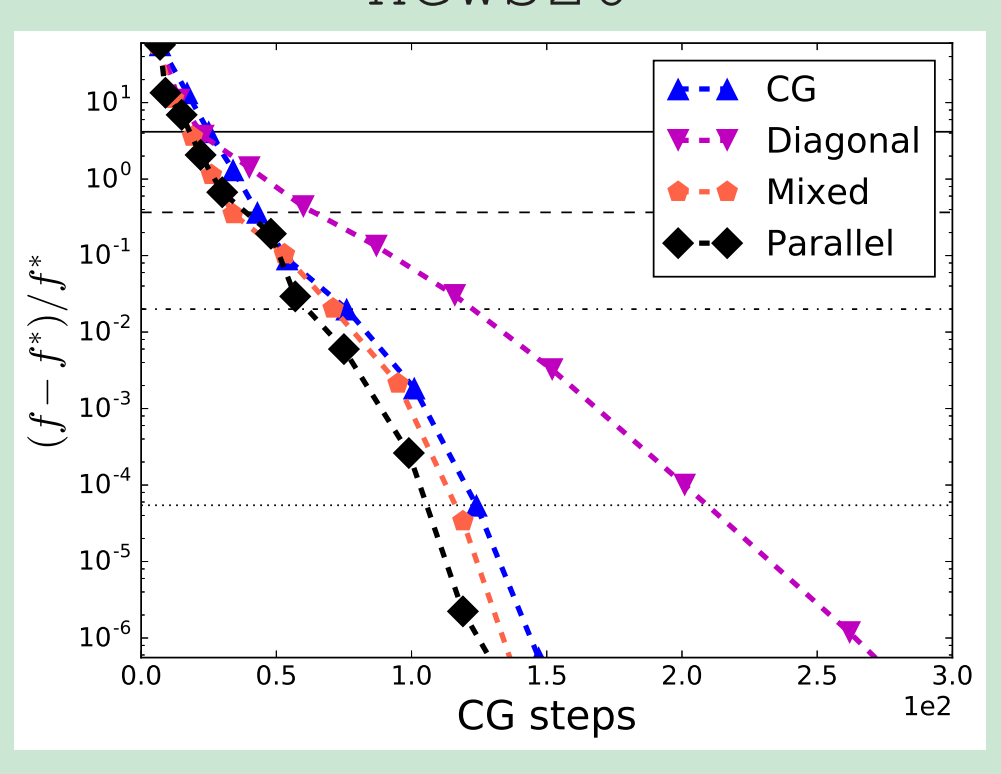
$$\bar{M} = \alpha \times \text{diag}(H_k) + (1-\alpha) \times I$$

The technique effectively improves the robustness of the diagonal preconditioner

kddb



news20



We show more robustness improvement in the experiments section.

Experiments

- The following preconditioners are considered
  - `Diag`: the diagonal preconditioner
  - `Parallel`: running CG and diagonal preconditioner in parallel
  - `Mixed`: our proposed method with a diagonal preconditioner
  - `SH-3000`: a sub-sampled Hessian preconditioner with $\bar{l} = 3{,}000$

Our settings are close to real-world scenarios. We measure the preconditioners for achieving a **suitable stopping condition** with $C \approx C_{\text{Best}}$ on the following data sets:

Data sets	#instances	#features	C_{Best}
news20	19,996	1,355,191	2^{9}
url	2,396,130	3,231,962	2^{-7}
yahookr	460,554	3,052,939	2^{6}
kddb	19,264,097	29,890,095	2^{-1}
criteo	45,840,617	1,000,000	2^{-15}
kdd12	149,639,105	54,686,452	2^{-4}

C_{Best} is the best regularization parameter selected by cross-validation

- Robustness improvement
We compare different preconditioners with the ratio

$$(\#\text{PCG steps})/(\#\text{CG steps}),\ \text{ratio} > 1 \text{ indicates PCG is not useful}$$

	$C = C_{\text{Best}}$			$C = 100C_{\text{Best}}$		
Data	Diag	Parallel	Mixed	Diag	Parallel	Mixed
news20	**1.61**	**1.06**	0.98	**2.38**	0.85	0.98
url	**1.25**	0.86	0.87	**1.14**	0.50	**1.29**
yahookr	0.29	0.44	0.67	0.31	0.11	0.16
kddb	0.24	0.25	0.28	0.04	0.03	0.05
kdd12	0.19	0.19	0.31	0.29	0.10	0.38
criteo	0.65	0.68	0.70	0.82	0.37	0.49

The proposed techniques effectively improve the robustness of the diagonal preconditioner. The behavior **is not sensitive** to the selection of α (details in paper)

- Running time comparison of using different preconditioners
We show the ratio

$$(\text{PCG running time})/(\text{CG running time}),\ \text{ratio} > 1 \text{ indicates PCG is not useful}$$

	$C = C_{\text{Best}}$			$C = 100C_{\text{Best}}$		
Data	Diag	Mixed	SH-3000	Diag	Mixed	SH-3000
news20	**1.76**	**1.13**	**44.36**	**2.51**	**1.15**	**48.20**
url	**1.24**	0.91	**1.16**	**1.18**	**1.28**	**1.28**
yahookr	0.35	0.73	**1.17**	0.33	0.19	**2.22**
kddb	0.28	0.31	0.41	0.05	0.05	0.42
kdd12	0.15	0.22	0.37	0.29	0.36	**1.27**
criteo	0.74	0.80	0.43	0.75	0.47	0.55

Preconditioners like **`SH-3000` may incur extra costs** and not useful in terms of **running time**. Overall **`Mixed` is the best approach**. It is **more robust than** `Diag` and is often **much faster** than standard CG and `SH-3000`

Conclusion

- Applying preconditioners on a sequence of linear systems in Hessian-free Newton is difficult
- We propose methods to improve the **robustness**
- The implementation is included in **LIBLINEAR** (`https://www.csie.ntu.edu.tw/~cjlin/liblinear/`). Many users are benefiting from this development