# Open Problem: Learning sparse linear concepts by priming the features

Manfred K. Warmuth and Ehsan Amid
Google Inc.

July 15, 2023
COLT 2023 in Bangalore

# Multiplicative & additive updates

Single example $(\boldsymbol{x}, y)$ for linear regression
Square loss $(\boldsymbol{x} \cdot \boldsymbol{w} - y)^2/2$

$$w_i = w_i - \eta(\boldsymbol{w} \cdot \boldsymbol{x} - y)x_i \qquad \text{(GD)}$$

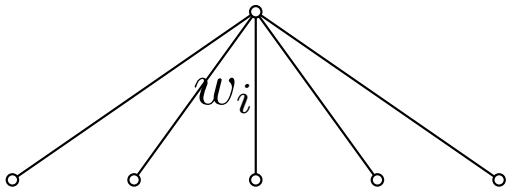
$$w_i = w_i \, \exp^{-\eta(\boldsymbol{w}\cdot\boldsymbol{x}-y)x_i} \qquad \text{(EGUnnormalized)}$$

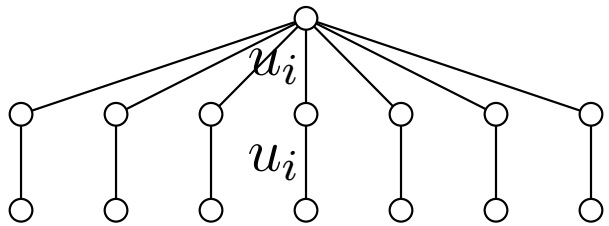
$$w_i = \frac{w_i \, \exp^{-\eta(\boldsymbol{w}\cdot\boldsymbol{x}-y)x_i}}{\sum_j w_j \, \exp^{-\eta(\boldsymbol{w}\cdot\boldsymbol{x}-y)x_j}} \qquad \text{(EG)}$$

Key: Multiplicative update much much better at learning when label $y$ is close to sparse linear

$\log d$ versus $d$ dependence in the regret

# EGU approximated by GD on spindly network



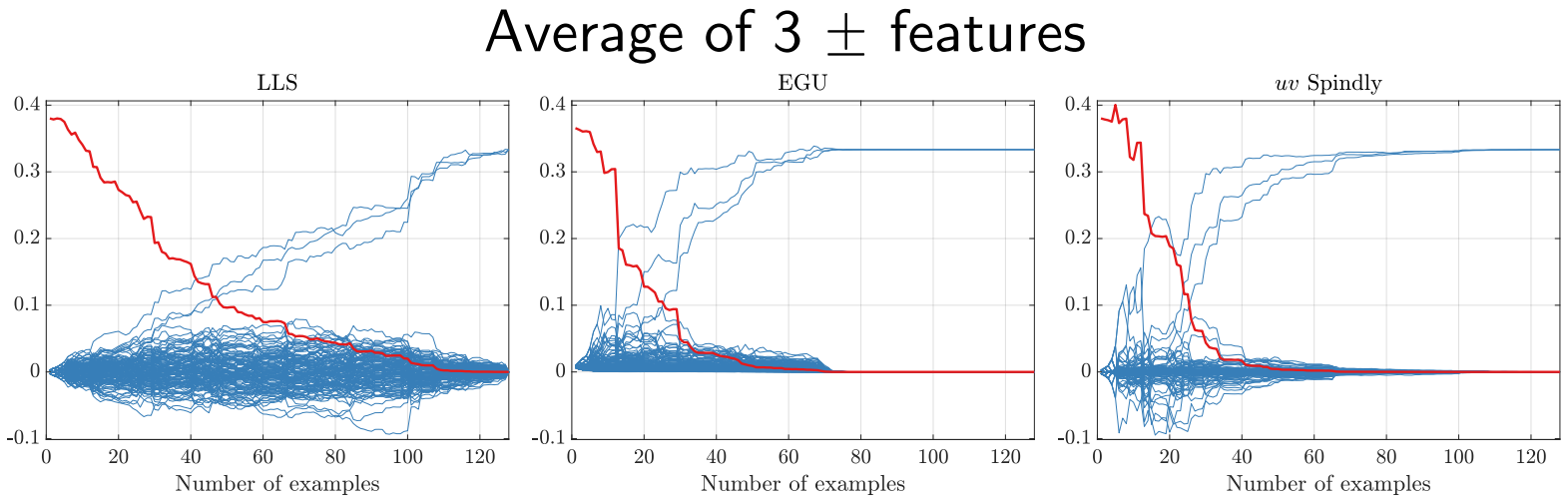

$$w_i = w_i \exp^{-\eta(\boldsymbol{w}\cdot\boldsymbol{x}-y)x_i} \quad \approx \quad u_i = u_i - \eta(\boldsymbol{u}\circ\boldsymbol{u}\cdot\boldsymbol{x} - y)x_i u_i)$$

EGU $\qquad\qquad$ $\boldsymbol{u}\circ\boldsymbol{u}$ spindly

$$\dot{\log}(\boldsymbol{w}) = -\eta\big(\boldsymbol{w}\cdot\boldsymbol{x} - y\big)\boldsymbol{x} \textbf{ equivalent } \dot{\boldsymbol{u}} = -\big(\boldsymbol{u}\circ\boldsymbol{u}\cdot\boldsymbol{x} - y\big)\boldsymbol{u}\circ\boldsymbol{x}$$

In many cases same regret bounds

# Plots




We draw a random $n \times n$ $\pm 1$ matrix $\boldsymbol{X}$, with $n = 128$. The rows are the $n$ instances and the sparse label vector $\boldsymbol{y}$ is the average of the first three column features. After seeing $t = 1 : 128$ examples, we plot weights (in blue) after applying LLS or the priming methods to all examples. For the online update, we do multiple passes over the past examples until the weight vector is consistent (EGU$\pm$ is a 2-sided version of EGU). In red, we always plot the average square loss on all 128 examples. LLS picks up the target weights too slowly.

# It is all about rotation invariance

Any neural net with a complete input layer and rotation invariant initialization can't learn sparse linear efficiently
**when trained with GD**

So on sparse targets, beaten by GD on spindly

# One big advantage of the GD family

Closed form in batch case:

$(\boldsymbol{X}, \boldsymbol{y})$, instances row-wise

$$\boldsymbol{w}_{\text{LLS}} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \, (\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y})^2 = \boldsymbol{X}^{\dagger}\boldsymbol{y} \qquad \text{(LLS)}$$

**Is there a closed form method that approximates multiplicative updates, i.e. is good at sparse targets**

# Priming

$i$th feature multiplied by the prime factor $p_i$ & apply LLS:

$$\boldsymbol{w_p} = \mathrm{diag}(\boldsymbol{p})\,(\boldsymbol{X}\,\mathrm{diag}(\boldsymbol{p}))^{\dagger}\boldsymbol{y}$$

**Choices**

1. $p_i = \mathrm{argmin}_{p_i}\,(\boldsymbol{X}(:,i)\,p_i - \boldsymbol{y})^2 = X(:,i)^{\dagger}\,\boldsymbol{y} = \frac{X(:,i)^{\top}}{||X(:,i||^2}\,\boldsymbol{y}$
   (1-dim LLS per $i$th feature)
2. $p_i = \frac{(X(:,i) - \overline{X(:,i)})\quad (\boldsymbol{y} - \overline{\boldsymbol{y}})}{\sqrt{(X(:,i) - \overline{X(:,i)}}\,\sqrt{\boldsymbol{y} - \overline{\boldsymbol{y}}}}$ (Pearson correlation coefficients)
3. $\boldsymbol{p} = \boldsymbol{w}_{\mathsf{LLS}} = \boldsymbol{X}^{\dagger}\boldsymbol{y} = (\boldsymbol{X}^{\top}\boldsymbol{X})^{\dagger}\,\boldsymbol{X}^{\top}\boldsymbol{y}$ (LLS for priming as well)

# Priming

$i$th feature multiplied by the prime factor $p_i$ & apply LLS:

$$\boldsymbol{w_p} = \mathrm{diag}(\boldsymbol{p})\,(\boldsymbol{X}\,\mathrm{diag}(\boldsymbol{p}))^{\dagger}\boldsymbol{y}$$

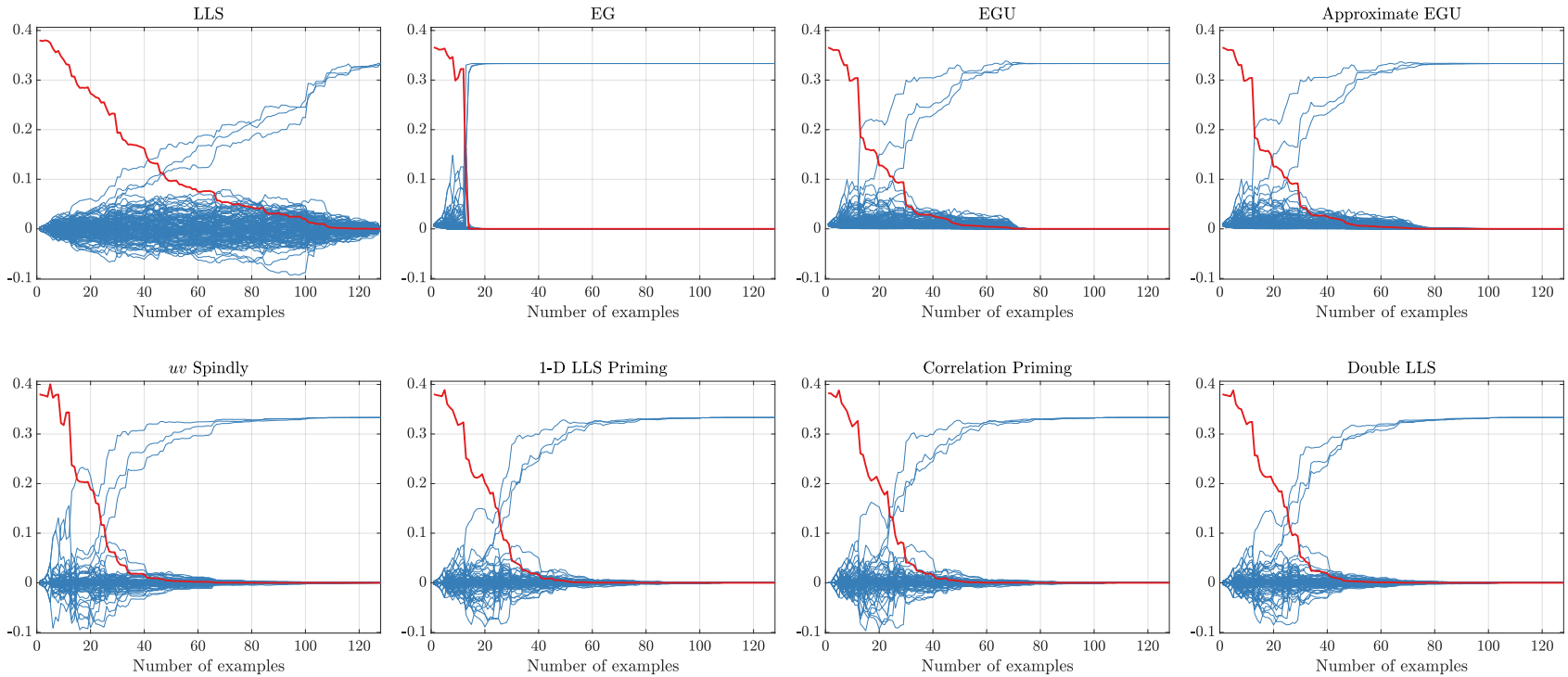
**Choices**

1. $p_i = \mathrm{argmin}_{p_i}\,(\boldsymbol{X}(:,i)\,p_i - \boldsymbol{y})^2 = X(:,i)^{\dagger}\,\boldsymbol{y} = \frac{X(:,i)^{\top}}{||X(:,i||^2}\,\boldsymbol{y}$
   (1-dim LLS per $i$th feature)
2. $p_i = \frac{(X(:,i) - \overline{X(:,i)})\quad(\boldsymbol{y}-\overline{\boldsymbol{y}})}{\sqrt{(X(:,i)-\overline{X(:,i)}}\,\sqrt{\boldsymbol{y}-\overline{\boldsymbol{y}}}}$ (Pearson correlation coefficients)
3. $\boldsymbol{p} = \boldsymbol{w}_{\mathsf{LLS}} = \boldsymbol{X}^{\dagger}\boldsymbol{y} = (\boldsymbol{X}^{\top}\boldsymbol{X})^{\dagger}\,\boldsymbol{X}^{\top}\boldsymbol{y}$ (LLS for priming as well)

**Open**

- Are there competitive regret bound for any of the priming methods?
- What is the optimal priming function for sparse linear problems?
- Are there priming methods for learning sparse disjunctions?

# Average of 3 $\pm$ features




LLS picks up the target weights too slowly. All three priming methods perform similarly to the multiplicative updates

# Kernelization

It does not help for learning sparse concepts

**Open**
Can kernels be primed efficiently?

**Dream**
If priming has regret bounds that grow with $\log d$
then learn sparse concepts in feature space

# Spindly connected to $L_1$!

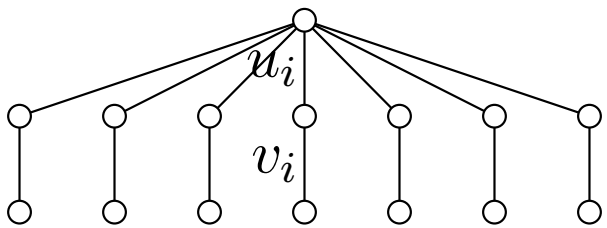
The Hoff $\boldsymbol{u} \circ \boldsymbol{v}$ Lemma

$$\begin{aligned} &\underset{\boldsymbol{w}}{\operatorname{argmin}}\, h(\boldsymbol{w}) + {}^{1}\!/\!_{\eta}\, ||\boldsymbol{w}||_1 \\ &= \underset{\boldsymbol{u},\boldsymbol{v}}{\operatorname{argmin}}\; h(\boldsymbol{u} \circ \boldsymbol{v}) + {}^{1}\!/\!_{\eta}\, ({}^{1}\!/\!_{2}\boldsymbol{u}^2 + {}^{1}\!/\!_{2}\boldsymbol{v}^2) \end{aligned}$$

- A one iteration batch update
- Assumes initializations are $\boldsymbol{w}, \boldsymbol{u}, \boldsymbol{v} = \mathbf{0}$

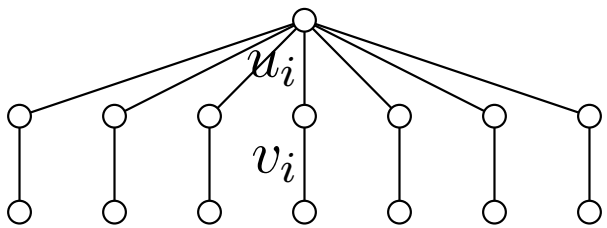
# Big picture

- Multiplicative = spindly in continuous case
- Spindly updates seem to have an $L_1$ motivation as well
- The are regret bounds for some spindlified updates
- Priming can be viewed as layerwise spindly



# Big picture

- Multiplicative = spindly in continuous case
- Spindly updates seem to have an $L_1$ motivation as well
- The are regret bounds for some spindlified updates
- Priming can be viewed as layerwise spindly




**How many hacks can we get away with and still learn sparse concepts?**