（注：这是大三科研其中的一份研究报告，以展示我的科研经历和学术水平）

研究报告 16
论文 4 篇

一、contents

Wang, W., Wang, J., Kolar, M., & Srebro, N. (2018). **Distributed Stochastic Multi-Task Learning with Graph Regularization**.
Xu, J., Tan, P. N., Luo, L., & Zhou, J. (2016). **GSpartan: A geospatio-temporal multi-task learning framework for multi-location prediction**. 16th SIAM International Conference on Data Mining 2016, SDM 2016, 657–665.
Li, C., Huang, S., Liu, Y., & Zhang, Z. (2018). **Distributed jointly sparse multitask learning over networks**. IEEE Transactions on Cybernetics, 48(1), 151–164.
Verma, V. K., & B, P. R. (2017). **Distributed Multi-task Learning for Sensor Network**. 1, 792–808.

二、总结

Wang, W., Wang, J., Kolar, M., & Srebro, N. (2018). **Distributed Stochastic Multi-Task Learning with Graph Regularization**.

主旨说明：
1） 对 W 的限制条件之一：两个近的点间 w 相差不大，利用图矩阵 L，变成正则项进入目标函数中
2） 使用随机梯度 SGD，减少计算次数
3） 我认为这篇论文实际上创新点一是加入了 Laplacian 图矩阵这样一个先验知识，而且作者在后面的 mini-batch SGD 中 sampling 也强调要尊重 graph，也即利用 graph 所包含的先验知识。二是对于算法的讨论，作者认为把 F（W）和 R（W）其中一个线性化可能就足够了，也能得到收敛后的 W.对于这两种方案，作者认为他们都是在不同的通信体制下使用，作者的目的在于减少交流和计算成本。利用 stochastic、prox 加快计算。

论文脉络：
1. Introduction
2. Graph-based multi-task learning
3. Distributed algorithm for ERM
   3.1 Directly solving the regularizer
   3.2 Directly optimizing the loss
4. Stochastic algorithms
   4.1 Directly solving the regularizer
   4.2 Directly optimizing the loss
5. Connection to consensus learning
6. Experiments

Details:

1. Introduction
2. Graph-based multi-task learning：提出了基于图矩阵的目标函数

$\mathbf{W}=[\mathbf{w}_1,\mathbf{w}_2,\ldots,\mathbf{w}_m]\in\mathbb{R}^{d\times m}$ 权重矩阵要满足下面的要求：

$$\Omega=\left\{\mathbf{W}:\ \|\mathbf{w}_i\|^2\le B^2,\quad \forall i=1,\ldots,m,\\ \sum_{i\neq k}\frac{a_{ik}}{2}\|\mathbf{w}_i-\mathbf{w}_k\|^2\le S^2\right\},$$

这个 Ω 的意义是：范数有界，且 related 的 predictors 不相似度小
全局的优化目标为：

$$F(\mathbf{W}):=\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}_{\mathbf{z}_i\sim\mathcal{D}_i}\left[\ell(\mathbf{w}_i,\mathbf{z}_i)\right], \tag{1}$$

其中，$F_i(\mathbf{w}_i)=\mathbb{E}_{\mathbf{z}_i\sim\mathcal{D}_i}\left[\ell(\mathbf{w}_i,\mathbf{z}_i)\right]$ 为每个 node 的优化目标。

注意，对集 Ω 的应用：$\mathbf{W}^*=\arg\min_{\mathbf{W}\in\Omega}F(\mathbf{W})$

对集 Ω 其中一个条件利用 Laplacian 图矩阵，即 L，则该条件可以转化为

$$\sum_{i\neq k}\frac{a_{ik}}{2}\|\mathbf{w}_i-\mathbf{w}_k\|^2=\sum_{i,k}\mathbf{L}_{ik}\langle\mathbf{w}_i,\mathbf{w}_k\rangle=\mathrm{tr}\left(\mathbf{W}\mathbf{L}\mathbf{W}^\top\right).$$

Regularized ERM (Empirical Risk Minimization) Objective：$\widehat{\mathbf{W}}=\arg\min_{\mathbf{W}}\ \widehat{F}(\mathbf{W})+R(\mathbf{W})$

$$\widehat{\mathbf{W}}=\arg\min_{\mathbf{W}}\underbrace{\frac{1}{m}\sum_{i=1}^{m}\widehat{F}_i(\mathbf{w}_i)}_{\widehat{F}(\mathbf{W})}\\ +\underbrace{\frac{\eta}{2m}\sum_{i=1}^{m}\|\mathbf{w}_i\|^2+\frac{\tau}{2m}\mathrm{tr}\left(\mathbf{W}\mathbf{L}\mathbf{W}^\top\right)}_{R(\mathbf{W})}, \tag{2}$$

3. Distributed algorithm for ERM

解决目标函数（2）：

首先对 $\hat{F}(W)$ 使用 gradient descent

$$\mathbf{w}_i^{t+1}=\sum_{k=1}^{m}\mu_{ki}^{t+1}\mathbf{w}_k^t-\alpha^{t+1}\nabla\widehat{F}_i(\mathbf{w}_i^t), \tag{3}$$

其中，μ_{ki}^{t+1} 为 weights for combining neighboring predictors

$$\mu_{ki}^{t+1}=\begin{cases}1-\alpha^{t+1}(\eta+\tau\sum_{k'}a_{ik'}) & :\text{if } i=k,\\ \alpha^{t+1}\tau a_{ik} & :\text{otherwise.}\end{cases} \tag{4}$$

问题：作者想要解决问题（2），原文说

The simplest approach is perhaps to perform gradient descent on $\hat{F}(W)$.

With an appropriate step-size schedule (or even a fixed stepsize if the loss is smooth), this method converges to $\hat{W}$？ 这里为什么只对 $\hat{F}(W)$ 使用了梯度下降，而没有管 R(W)?

原文：Taking steps based on the gradients amounts to considering, in each iteration, a linearization of the objective, that is of both the empirical loss $\hat{F}(W)$ and the regularizer R(W).

根据梯度采取的步骤就相当于在每次迭代中，对目标函数 $\hat{F}(W)$ 同时线性化。

即正常要对 $\widehat{F}(\mathbf{W})$ 和 $R(\mathbf{W})$ 都线性化，但是作者接下来仅对其中一个线性化，并且对另一项给出明确的处理。作者认为下面两种方法是强大的 alternatives。

3.1 Directly solving the regularizer

首先，利用变换

$$\mathbf{U}^t = \mathbf{W}^t\mathbf{M}^{\frac{1}{2}} \text{ where } \mathbf{M} = \mathbf{I} + \frac{\tau}{\eta}\mathbf{L}$$

我认为在这里使用变化 U 并没有什么特别的地方，只是为了将 R(W)中的两项结合成一项 $\|U\|^2_F$, 3.1 节的重点是将 F（W）线性化，最后得到一个计算 w 的算法

将目标函数改为，

$$\min_{\mathbf{U}} \widehat{F}(\mathbf{U}\mathbf{M}^{-\frac{1}{2}}) + \frac{\eta}{2m}\|\mathbf{U}\|_F^2. \tag{5}$$

通过 Appendix D 的叙述，可见这里对 $\widehat{F}(\mathbf{U}\mathbf{M}^{-\frac{1}{2}})$ 进行了线性化，二阶近似形式

推导的理论基础 https://blog.csdn.net/qq_38290475/article/details/81052206

$$\mathbf{U}^{t+1} = \arg\min_{\mathbf{U}} \quad \alpha^{t+1}\langle \nabla \widehat{F}^{t+1}(\mathbf{U}^t\mathbf{M}^{-\frac{1}{2}})\cdot\mathbf{M}^{-\frac{1}{2}}, \mathbf{U}-\mathbf{U}^t\rangle + \frac{1}{2}\|\mathbf{U}-\mathbf{U}^t\|_F^2, \qquad \text{for} \quad t=0,\ldots, \tag{10}$$

关于 U 使用 GD，在 W-space，上述公式简化为:

$$\mathbf{W}^{t+1} = \left(1-\alpha^{t+1}\eta\right)\mathbf{W}^t - \alpha^{t+1}\nabla\widehat{F}(\mathbf{W}^t)\cdot\mathbf{M}^{-1} \tag{6}$$

分布到每一个 node：

$$\mathbf{w}_i^{t+1} = \left(1-\alpha^{t+1}\eta\right)\mathbf{w}_i^t - \sum_{k=1}^{m}\mu_{ki}^{t+1}\nabla\widehat{F}_k(\mathbf{w}_k^t). \tag{7}$$

Note：$\mu_{ki}^{t+1} = \alpha^{t+1}(\mathbf{M}^{-1})_{ki}$ 通过通信一圈得到每个 node 的梯度，M^{-1} 提前计算

3.2 Directly optimizing the loss

由于上面的算法需要通信一圈，不太符合分布式的设置，于是得到下面的算法：

linearize the graph regularizer but fully optimize over the loss:

$$\mathbf{W}^{t+1} = \arg\min_{\mathbf{W}} \langle \nabla R(\mathbf{W}^t), \mathbf{W}-\mathbf{W}^t\rangle + \frac{1}{2m\alpha^{t+1}}\|\mathbf{W}-\mathbf{W}^t\|_F^2 + \widehat{F}(\mathbf{W}), \tag{8}$$

公式（8）也使用了近端梯度下降方法，$\|W-W^t\|^2_F$ 是在二次近似后自然而然出来的，线性化后可能有不好的性质，强凸保证

分配到 node：

下面的公式利用了近端梯度算法，

$$\mathbf{w}_i^{t+1} = \arg\min_{\mathbf{u}} \frac{1}{2\alpha^{t+1}}\left\|\mathbf{u} - (\mathbf{w}_i^t - m\alpha^{t+1}\nabla_{\mathbf{w}_i}R(\mathbf{W}^t))\right\|^2 + \widehat{F}_i(\mathbf{u})$$

根据优化的最优条件，

$$\mathbf{w}_i^{t+1} = \sum_{k=1}^{m}\mu_{ki}^{t+1}\mathbf{w}_k^t - \alpha^{t+1}\nabla\widehat{F}_i(\mathbf{w}_i^{t+1}), \tag{9}$$

Summary：

上述三种方法：

问题：

$$\widehat{\mathbf{W}} = \arg\min_{\mathbf{W}} \underbrace{\frac{1}{m}\sum_{i=1}^{m} \widehat{F}_i(\mathbf{w}_i)}_{\widehat{F}(\mathbf{W})} + \underbrace{\frac{\eta}{2m}\sum_{i=1}^{m}\|\mathbf{w}_i\|^2 + \frac{\tau}{2m}\operatorname{tr}\left(\mathbf{W}\mathbf{L}\mathbf{W}^\top\right)}_{R(\mathbf{W})}, \tag{2}$$

1） F 和 R 全部线性化：得到的是公式（3）

$$\mathbf{w}_i^{t+1} = \sum_{k=1}^{m} \mu_{ki}^{t+1}\mathbf{w}_k^t - \alpha^{t+1}\nabla\widehat{F}_i(\mathbf{w}_i^t), \tag{3}$$

Where，
$$\mu_{ki}^{t+1} = \begin{cases} 1-\alpha^{t+1}(\eta+\tau\sum_{k'} a_{ik'}) & : \text{if } i=k, \\ \alpha^{t+1}\tau a_{ik} & : \text{otherwise.} \end{cases} \tag{4}$$

2） 线性化处理 F：得到公式（7）

$$\mathbf{w}_i^{t+1} = \left(1-\alpha^{t+1}\eta\right)\mathbf{w}_i^t - \sum_{k=1}^{m}\mu_{ki}^{t+1}\nabla\widehat{F}_k(\mathbf{w}_k^t). \tag{7}$$

Where，$\mu_{ki}^{t+1} = \alpha^{t+1}(\bar{\mathbf{M}}^{-1})_{ki}$

3） 线性化处理 R：得到公式（9）

$$\mathbf{w}_i^{t+1} = \sum_{k=1}^{m} \mu_{ki}^{t+1}\mathbf{w}_k^t - \alpha^{t+1}\nabla\widehat{F}_i(\mathbf{w}_i^{t+1}), \tag{9}$$

Where,
$$\mu_{ki}^{t+1} = \begin{cases} 1-\alpha^{t+1}(\eta+\tau\sum_{k'} a_{ik'}) & : \text{if } i=k, \\ \alpha^{t+1}\tau a_{ik} & : \text{otherwise.} \end{cases} \tag{4}$$

作者认为 2）3）两种方案，每次只处理 F 或 R 其中一项，到底选择哪种要看图关系、结构和交流的形式。

作者认为公式（7）需要 using one round of global, all-to-all communication

认为公式（9）需要 using a local，peer-to-peer communication

这个地方的 global 应该指全局通信，通信一圈，而 local 应该指只和邻居通信。但是从公式没看出来？

作者说，Comparing (9) with the similar update (3) where we linearized both the regularizer and the loss, we observe that (9) is also a form of gradient method, with the gradient of loss evaluated at the “future” point.

也即（3）和（9）公式前面都是相同的，唯一的区别在于（9）的最后一项是对 w_i^{t+1} 使用梯度，而（3）是对 w_i^t 使用梯度。

作者说，The advantage of (9) is that the gradient ∇R(W) is data-independent and is obtained using only one round of local communication from each machine to its neighbors.

确实，R(W)是跟 data 或 example 无关，而 $\widehat{F}_i(\mathbf{w}_i) = \frac{1}{n}\sum_{j=1}^{n}\ell(\mathbf{w}_i, \mathbf{z}_{ij})$ 是 data-dependent。但是后面的只需要和邻居一圈的当地通信即可得到，没太懂？

ERM: directly solving regularizer
1. $\mathbf{g}_i^{t+1} = \sum_k \mu_{ki}^{t+1} \nabla \widehat{F}_k(\mathbf{w}_k^t)$
where $\mu_{ki}^{t+1} = \alpha^{t+1}(\mathbf{M}^{-1})_{ki}$
2. $\mathbf{w}_i^{t+1} = \mathbf{w}_i^t - \mathbf{g}_i^{t+1}$
ERM: directly optimizing loss
1. $\widetilde{\mathbf{w}}_i^t = \sum_k \mu_{ki}^{t+1} \mathbf{w}_k^t$
where $\mu_{ki}^{t+1} = (\mathbf{I} - \alpha^{t+1}\eta\mathbf{M})_{ki}$
2. $\mathbf{w}_i^{t+1} = \widetilde{\mathbf{w}}_i^t - \alpha^{t+1}\nabla \widehat{F}_i^{t+1}(\mathbf{w}_i^{t+1})$

作者强调交流的信息 message 有两种，一种是 gradient 表格第一行，一种是 iterates，即 w_k^t 表格下一行。但是都要通信，通信 gradient 和 parameter 有什么区别呢？

4. Stochastic algorithms

4.1 Directly solving the regularizer

minibatch SGD with **b samples** per machine

$$\mathbf{w}_i^{t+1} = \mathbf{w}_i^t - \sum_{k=1}^m \mu_{ki}^{t+1} \nabla \widehat{F}_k^{t+1}(\mathbf{w}_k^t). \tag{10}$$

where $\widehat{F}_k^{t+1}(\mathbf{w}_k^t) = \frac{1}{b}\sum_{j=1}^b \ell(\mathbf{w}_k^t, \mathbf{z}_{kj}^{t+1})$, and $\left\{\mathbf{z}_{kj}^{t+1} : j = 1, \ldots, b\right\}$ are b samples drawn by machine k at iteration $t+1$.

4.2 Directly optimizing the loss

$$\mathbf{w}_i^{t+1} = \arg\min_{\mathbf{u}} \frac{1}{2\alpha^{t+1}} \left\|\mathbf{u} - \left(\mathbf{w}_i^t - m\alpha^{t+1}\nabla_{\mathbf{w}_i} R(\mathbf{W}^t)\right)\right\|^2 + \frac{1}{b}\sum_{j=1}^b \ell(\mathbf{u}, \mathbf{z}_{ij}^{t+1}). \tag{11}$$

5. Connection to consensus learning

在这一小节作者强调的是，当参数怎样变化时，可以把 multi-solution 变成共识 consensus 问题，共识问题指的就是所有的 nodes 最后都得到了一个模型，一个参数 W。

作者想说他们的模型对于这种 consensus 问题是兼容的。

Xu, J., Tan, P. N., Luo, L., & Zhou, J. (2016). **GSpartan: A geospatio-temporal multi-task learning framework for multi-location prediction**. 16th SIAM International Conference on Data Mining 2016, SDM 2016, 657–665.

主旨说明：提出 GSpartan 模型

1）Local models 共享一些公共的、低秩 representations

2）每个局部模型（local models）作为基模型（base models）的线性组合

3）有两种变量：response variable，即我们想要预测的变量；

predictor variables，是由全局或地区模型产生的其他输出

比如，预测每月降雨量是 response variable ;mean temperature at 2 meters, mean sea level pressure, 500 hPa geopotential height, and near surface relative humidity 是 predictor variables。

也就是说 response variable 是因变量 y；predictor variables 是自变量 X

4）实验部分通过去掉拉普拉斯图矩阵的正则化进行比较，还和一些其他方法进行了比较

5）拉普拉斯图矩阵：通过修正变差函数的逆（inverse of a modified variogram）给出

Variogram：方差图是一种空间统计中发展起来的一种度量方法，用来确定一对位置之间的空间依赖性。见引用文献[10]

$$\mathbf{A}_{i,j} = \begin{cases} 1 & \text{if } i = j \\ \frac{1}{var(\mathbf{y}_i - \mathbf{y}_j)} & \text{otherwise} \end{cases}$$

目标函数：

$$\begin{aligned} \min_{\mathbf{W},\mathbf{U},\mathbf{V}} \quad & \frac{1}{2}\sum_{s=1}^{|S|} \|\mathbf{X}_s\mathbf{w}_s - \mathbf{y}_s\|_2^2 + \lambda_1\|\mathbf{V}\|_1 + \lambda_2\|\mathbf{U}\|_1 \\ & + \frac{\lambda_3}{2}\mathrm{Tr}(\mathbf{W}(\mathbf{D}-\mathbf{A})\mathbf{W}^T) \\ \text{s.t.} \quad & \mathbf{V} \succeq 0,\ \mathbf{W} = \mathbf{UV} \end{aligned} \xrightarrow{W=UV} \begin{aligned} \min_{\mathbf{U},\mathbf{V}} \quad & \frac{1}{2}\sum_{i=1}^{|S|} \|\mathbf{X}_s\mathbf{U}\mathbf{v}_i - \mathbf{y}_s\|_2^2 + \lambda_1\|\mathbf{V}\|_1 \\ (4.1) \quad & + \lambda_2\|\mathbf{U}\|_1 + \frac{\lambda_3}{2}\mathrm{Tr}(\mathbf{U}\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\mathbf{U}^T) \\ \text{s.t.} \quad & \mathbf{V} \succeq 0 \end{aligned}$$

Note：W=UV

U：a feature representation of the base models

V：the weighted combination of the base models that form the local model at each location.

矩阵 U 是基本模型的特征表示，V 表示在一点形成局部模型的基本模型的加权组合。

A：拉普拉斯图矩阵，Task relation matrix，表示空间自相关，引入任务关系的先验知识

为了模型的可解释性，引入稀疏和非负。作者在这里希望两个稀疏，一个是每个 base model 里面的变量尽量少，即如果有 1000 个变量，最好只需要 5 个用来解释。另一个稀疏是 local 用的 base model 少一些。所以，分别在 U,V 上加 1-norm 稀疏，另外保证非负，即不引入负向的解释。

Tr（）：类似理解成 F 范数的形式，用来表征任务之间尽可能相似的 penalty term。即 A 表示空间自相关，两个点地理上相近，那么他们的参数也相近。

优化算法：

通过 BCD，交替优化 U 和 V

Input: Dataset $\mathcal{D} = \{(\mathbf{X}_1, \mathbf{y}_1), ..., (\mathbf{X}_S, \mathbf{y}_S)\}$,
Task relation matrix A, parameters λ_1, λ_2, λ_3;
Initialize: Randomly generate $\mathbf{U}$ and $\mathbf{V}$ and set
$k = 1$
Block coordinate descent:
while *not converge* **do**
  Solve U given V:
  Compute τ_U^k using Theorem 1
  Update $\mathbf{U}^k$ using Equation (4.4)
  Solve V given U:
  Compute τ_V^k using Theorem 2
  Update $\mathbf{V}^k$ using Equation (4.7)
  $k = k + 1$
end
return $\{U^k, V^k\}$

Algorithm 1: Pseudocode for GSpartan framework

这里可以看到，虽然 V≥0，是有约束的优化，但是可以看到通过分步优化，就相当于没有约束条件。

Solve U, given V:

When $\mathbf{V}$ is fixed, the objective function can be simplified as follows:

$$\min_{\mathbf{U}} \quad \frac{1}{2}\sum_{i=1}^{|S|}\|\mathbf{X}_s\mathbf{U}\mathbf{v}_i - \mathbf{y}_s\|_2^2 + \lambda_2\|\mathbf{U}\|_1 + \frac{\lambda_3}{2}\mathrm{Tr}(\mathbf{U}\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\mathbf{U}^T) \tag{4.2}$$

通过 proximal gradient descent 进一步计算，

$$\mathbf{U}^k = \operatorname*{argmin}_{\mathbf{U}}(\mathbf{U} - \hat{\mathbf{U}}^{k-1})^T\hat{\mathbf{g}}_U^k + \frac{\tau_U^{k-1}}{2}\|\mathbf{U} - \hat{\mathbf{U}}^{k-1}\|_F^2 + \lambda_2\|\mathbf{U}\|_1, \tag{4.3}$$

where

$$\hat{\mathbf{g}}_U^k = \sum_{s=1}^{|S|}\left(-\mathbf{X}_s^T\mathbf{y}_s\mathbf{v}_s^T + \mathbf{X}_s^T\mathbf{X}_s\mathbf{U}\mathbf{v}_s\mathbf{v}_s^T\right) + \lambda_3\mathbf{U}\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T$$

and

$$\hat{\mathbf{U}}^{k-1} = \mathbf{U}^{k-1} + \omega_U^{k-1}(\mathbf{U}^{k-1} - \mathbf{U}^{k-2})$$

The solution for problem (4.3) is given by

$$\mathbf{U}^k = \mathcal{S}_{\tau_U^{k-1}/\lambda_2}(\hat{\mathbf{U}}^{k-1} - \frac{\hat{\mathbf{g}}^{k-1}}{\tau_{\mathbf{U}}^{k-1}}) \tag{4.4}$$

where $\mathcal{S}_\alpha(t) = \mathrm{sign}(t)(\max(|t| - \alpha, 0)$ is a component-wise soft-thresholding function.

$$\tau_U = \sum_{s=1}^{|S|}\|\mathbf{X}_s^T\mathbf{X}_s\|\|\mathbf{v_s}\mathbf{v_s^T}\| + \lambda_3\|\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\|$$

Solve V, given U

Similarly, when $\mathbf{U}$ is fixed, the objective function becomes:

$$\min_{\mathbf{V}} \quad \frac{1}{2}\sum_{i=1}^{|S|}\|\mathbf{X}_s\mathbf{U}\mathbf{v}_i - \mathbf{y}_s\|_2^2 + \lambda_1\|\mathbf{V}\|_1 + \frac{\lambda_3}{2}\mathrm{Tr}(\mathbf{U}\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\mathbf{U}^T) \tag{4.5}$$

同样通过 proximal gradient descent，

$$\mathbf{V}^k = \operatorname*{argmin}_{\mathbf{V}}(\mathbf{V} - \hat{\mathbf{V}}^{k-1})^T\hat{\mathbf{g}}_V^k + \frac{\tau_V^{k-1}}{2}\|\mathbf{V} - \hat{\mathbf{V}}^{k-1}\|_F^2 + \lambda_1\|\mathbf{V}\|_1 \tag{4.6}$$

where

$$\hat{\mathbf{g}}_V^k = \mathbf{P} + \lambda_3\mathbf{U}^T\mathbf{U}\mathbf{V}(\mathbf{D}-\mathbf{A})$$

The s-th column of matrix $\mathbf{P}$ is given by

$$\mathbf{p}_s = -\mathbf{U}^T\mathbf{X}_s^T\mathbf{y}_s + \mathbf{U}^T\mathbf{X}_s^T\mathbf{X}_s\mathbf{U}\mathbf{v}_s$$

and

$$\hat{\mathbf{V}}^{k-1} = \mathbf{V}^{k-1} + \tau_V^{k-1}(\mathbf{V}^{k-1} - \mathbf{V}^{k-2})$$

The solution for problem (4.6) is given by

$$\mathbf{V}^k = \mathcal{S}_{\omega_V^{k-1}/\lambda_2}(\hat{\mathbf{V}}^{k-1} - \frac{\hat{\mathbf{g}}^{k-1}}{\tau_V^{k-1}}) \tag{4.7}$$

$$\tau_V = \|\mathbf{U}\|^2R^2 + \lambda_3\|\mathbf{U}^T\mathbf{U}\|\|\mathbf{D}-\mathbf{A}\|$$

推导过程：

利用了近端梯度算法。推导 U，同理可得 V

总结报告15.16：GSpartan: A geospatio-temporal multi-task learning framework for multi-location prediction

- 近端梯度算法理论基础：

$\min_x (g(x)+h(x))$　$g(x)$ 光滑，$h(x)$ 非光滑（存在不可微）

则其迭代通推公式为

$$x^k = prox_{t_k,h}\left(x^{k-1} - t_k \nabla g(x^{k-1})\right) \quad \text{——— (1)}$$

prox 算子：

$$prox_h(x) = \arg\min_u \left(h(u) + \frac{1}{2}\|u-x\|_2^2\right) \quad \text{— (2)}$$

说明：(1)式：$x^k = prox_{t_k,h}\left(x^{k-1} - t_k \nabla g(x^{k-1})\right)$

$$= \arg\min_u \left(h(u) + \frac{1}{2t_k}\left\|u - \left(x^{k-1} - t_k \nabla g(x^{k-1})\right)\right\|_2^2\right) \quad \text{— (3)}$$

$$= \arg\min_u \left(h(u) + \frac{1}{2t_k}\left\|u - x^{k-1} + t_k \nabla g(x^{k-1})\right\|_2^2\right) \quad \text{— (4)}$$

将二次项展开

$$= \arg\min_u \left(h(u) + \frac{t_k}{2}\|\nabla g(x^{k-1})\|_2^2 + \nabla g(x^{k-1})^T(u - x^{k-1}) + \frac{1}{2t_k}\|u - x^{k-1}\|_2^2\right) \cdots (5)$$

对 u 的 min，故将红框内替换一下

$$= \arg\min_u \left(h(u) + g(x^{k-1}) + \nabla g(x^{k-1})^T(u - x^{k-1}) + \frac{1}{2t_k}\|u - x^{k-1}\|_2^2\right) \cdots (6)$$

还仅一阶 Taylor 展开

$$\approx \arg\min_u \left(h(u) + g(u)\right) \quad \text{— (7)}$$

- 推导：solve U, given V

loss：$\min_U \frac{1}{2}\sum_{i=1}^{|S|}\|X_s U v_s - y_s\|_2^2 + \frac{\lambda_3}{2} Tr\left(UV(D-A)V^T U^T\right) + \lambda_2\|U\|_1$

$g(x)$ 光滑　　$h(x)$ 不可微类

迭代公式：$U^k = \arg\min_S \left(\lambda_2\|S\|_1 + \frac{1}{2t_k}\left\|S - \left(U^{k-1} - t_k \nabla g(\hat{U}^{k-1})\right)\right\|_2^2\right)$

$h(u)$

$$= \arg\min_S \left(\lambda_2\|S\|_1 + \frac{1}{2t_k}\left\|S - \hat{U}^{k-1} + t_k \nabla g(\hat{U}^{k-1})\right\|_2^2\right)$$

打开二次项

$$= \arg\min_S \left(\lambda_2\|S\|_1 + \frac{t_k}{2}\|\nabla g(\hat{U}^{k-1})\|_2^2 + \nabla g(\hat{U}^{k-1})^T(S - \hat{U}^{k-1}) + \frac{1}{2t_k}\|S - \hat{U}^{k-1}\|_F^2\right)$$

令=0

把 S 用 U 带回与论文统一写成此形式

$$= \arg\min_U \left((U - \hat{U}^{k-1})^T \hat{g}_U^k + \frac{t_U^{k-1}}{2}\|U - \hat{U}^{k-1}\|_F^2 + \lambda_2\|U\|_1\right)$$

$$\therefore \hat{g}_U^k = \nabla g(\hat{U}^{k-1}) = \nabla\left(\frac{1}{2}\sum_{s=1}^{|S|}\|X_s U v_s - y_s\|_2^2 + \frac{\lambda_3}{2} Tr\left(UV(D-A)V^T U^T\right)\right)$$

$$= \nabla\left(\frac{1}{2}\sum_{s=1}^{|S|}\left(X_s^T X_s U U^T v_s v_s^T + y_s^T y_s - 2X_s^T U v_s y_s\right) + \frac{\lambda_3}{2} Tr\left(UV(D-A)V^T U^T\right)\right)$$

$$= \sum_{s=1}^{|S|}\left(-X_s^T y_s v_s^T + X_s^T X_s U v_s v_s^T\right) + \lambda_3 \cdot UV(D-A)V^T$$

论文中 U_U^{k-1} 用来保证收敛。(4.4)中迭代公式 $U^k = S_{t_U^{k-1}\lambda_2}\left(\hat{U}^{k-1} - \frac{\hat{g}_U^{k-1}}{t_U^{k-1}}\right)$ 软阈值 ∇g

Li, C., Huang, S., Liu, Y., & Zhang, Z. (2018). **Distributed jointly sparse multitask learning over networks**. IEEE Transactions on Cybernetics, 48(1), 151–164.

主旨说明：AC-dJSMT algorithm

1） 文章考虑了多任务的两点：

ρ 的一项为惩罚 intertask similarities； η 的一项用来惩罚 joint sparsity。

如果只考虑其中一种情况，只须让对应的项的系数为 0.

2） 为了分布式，将 W 分配给每个节点 W_k：a local parameter matrix（consisting of the parameter vectors of all its neighbors），作者在这里说明了使用 W_k 最终也会收敛到 min

3） 作者认为 joint sparsity 意味着每一个 node 的参数向量 w_k^0 哪个位置是 0，哪个位置不是 0，对于所有的节点都是一样的，比如第一个分量 all nodes 全是 0，第二个分量全不是 0…这样的结构，即

$$\operatorname{supp}\left(w_1^0\right)=\cdots=\operatorname{supp}\left(w_k^0\right)=\cdots=\operatorname{supp}\left(w_N^0\right). \quad (3)$$

4) intertask similarities 的部分 c_{lk} intertask combiner，见下优化算法所述。

目标函数：

$$J_k^{loc}(w)=E\left[\left(\boldsymbol{e}_{k,i}\right)^2\right]+\rho\sum_{l\in\mathcal{N}_k}c_{lk}\|w-w_{l,i-1}\|_2^2+\eta\|W_k\|_{2,p} \quad (5)$$

Note： $\boldsymbol{e}_{k,i}=\boldsymbol{d}_{k,i}-\boldsymbol{u}_{k,i}$.另外有 $\boldsymbol{d}_{k,i}=\boldsymbol{u}_{k,i}w_k^0+\boldsymbol{v}_{k,i}$ (1)

$\sum_{l\in\mathcal{N}_k}c_{lk}=1,\quad c_{lk}=0,\ \text{if } l\notin\mathcal{N}_k.$ (6) ：intertask combiners

$\|w-w_{l,i-1}\|_2^2$：similarity-promoting term，想让 intertask 即 related tasks 邻居之间的参数相似一些

$\|W_k\|_{2,p}$：用来惩罚稀疏，sparse-promoting term，分别讨论了 L2,1-norm, reweighted L2,1-norm, L2，0-norm

解决方法：

steepest-descent method：这里只是简单的用 gradient descent 方法求解，重点在于对 L2，p-norm 范数求次梯度.

$$\nabla_w J_k^{loc}(w)=\frac{\partial J_k^{loc}(w)}{\partial w}=-E\left[\boldsymbol{e}_{k,i}\boldsymbol{u}_{k,i}^T\right]+\rho\sum_{l\in\mathcal{N}_k}c_{lk}\left(w-w_{l,i-1}\right)+\eta f_p(W_k). \quad (7)$$

Note: $f_p(W_k)$ 是关于 w 的 $L_{2,\,p}$ 范数的次梯度

迭代更新公式：

$$w_{k,i} = w_{k,i-1} + \mu E\left[\boldsymbol{e}_{k,i}\boldsymbol{u}_{k,i}^T\right] - \mu\rho \sum_{l\in\mathcal{N}_k} c_{lk}\left(w_{k,i-1} - w_{l,i-1}\right) - \mu\eta f_p\left(W_{k,i-1}\right) \quad (8)$$

用$e_{k,i}u_{k,i}^T$来代替$E\left[\boldsymbol{e}_{k,i}\boldsymbol{u}_{k,i}^T\right]$得到，

$$w_{k,i} = w_{k,i-1} + \mu e_{k,i}u_{k,i}^T - \tau \sum_{l\in\mathcal{N}_k} c_{lk}\left(w_{k,i-1} - w_{l,i-1}\right) - \gamma f_p\left(W_{k,i-1}\right) \quad (9)$$

where $\tau = \mu\rho$, and $\gamma = \mu\eta$.

优化算法：

Algorithm 1 AC-dJSMT Algorithm

Initialization: initialize $w_{k,0}$ for each node k, step size μ, regularization parameters τ and γ, constant ι, and total iterations T

for i=1: T

for each node k:

Adaptation

$$c_{lk,i} = \frac{2}{|\mathcal{N}_k^-|} \cdot \frac{1}{1+e^{\iota\|w_{k,i-1}-w_{l,i-1}\|_2^2}}$$

$$w_{k,i} = w_{k,i-1} + \mu e_{k,i}u_{k,i}^T - \tau \sum_{l\in\mathcal{N}_k} c_{lk,i}(w_{k,i-1} - w_{l,i-1}) - \gamma f_p(W_{k,i-1})$$

Communication

Transmit $w_{k,i}$ to its one-hop neighbors

end for

- 当c_{lk} is fixed，得到了 FC-dJSMT（fixed c_{lk} distributed jointly sparse multitask）

- 当c_{lk} is adaptive，得到了 AC-dJSMT（adaptive c_{lk} distributed jointly sparse multitask）

由下面的公式给出c_{lk}的取值，

$$c_{lk,i} = \begin{cases} \frac{2}{|\mathcal{N}_k^-|} \cdot \frac{1}{1+e^{\iota\|w_{k,i-1}-w_{l,i-1}\|_2^2}}, & l \in \mathcal{N}_k^- \\ 1 - \sum_{l\in\mathcal{N}_k^-} c_{lk,i}, & l = k \\ 0, & l \notin \mathcal{N}_k \end{cases} \quad (10)$$

where $\mathcal{N}_k^-$ denotes the neighbors of node k except itself, namely, $\mathcal{N}_k^- = \mathcal{N}_k \setminus \{k\}$. Here, $|\mathcal{N}_k^-|$ stands for the size of the set $\mathcal{N}_k^-$, and ι is a large constant set beforehand. The rationality of this design can be found in [41].

- 对于稀疏惩罚项$\|W\|_{2,\,p}$的范数的讨论

$$W_k = \begin{bmatrix} w_{11} & w_{12} & w_{1l} & w_{1k} \\ w_{21} & w_{22} & w_{2l} & w_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ w_{M1} & w_{M2} & w_{Ml} & w_{Mk} \end{bmatrix} = \begin{bmatrix} \check{w}_1^k \\ \check{w}_2^k \\ \vdots \\ \check{w}_M^k \end{bmatrix}$$

其中，$\check{w}_m^k = [w_{m1}, w_{m2}, w_{ml}, w_{mk}]$

A. $L_{2,\,1}$-norm

$$\|W_k\|_{2,1} = \sum_{m=1}^{M} \left\|\check{w}_m^k\right\|_2 = \sum_{m=1}^{M} \left(\sum_{l\in\mathcal{N}_k} \|w_{ml}\|_2^2\right)^{1/2}. \quad (11)$$

它的次梯度为：

$$f_1(W_k) = \mathrm{col}\left\{\frac{w_{1k}}{\|\check{w}_1^k\|_2}, \ldots, \frac{w_{Mk}}{\|\check{w}_M^k\|_2}\right\}. \quad (12)$$

其中，$\check{w}_1^k$ 就是 W_k 矩阵的第一行

B. Reweighted $L_{2,1}$-norm ： RW $l_{2,1}$ 范数

$$\|W_k\|'_{2,1} = \sum_{m=1}^{M} \log\left(1 + \varepsilon \left\|\check{w}_m^k\right\|_2\right) \quad (13)$$

它的次梯度为：

$$f'_1(W_k) = \mathrm{col}\left\{\frac{w_{1k}/\|\check{w}_1^k\|_2}{1+\varepsilon\|\check{w}_1^k\|_2}, \ldots, \frac{w_{Mk}/\|\check{w}_M^k\|_2}{1+\varepsilon\|\check{w}_M^k\|_2}\right\}. \quad (14)$$

把分子的$\|\check{w}_1^k\|_2$除下来，可以发现，$\|\check{w}_1^k\|_2$越小，$\frac{w_{1k}/\|\check{w}_1^k\|_2}{1+\varepsilon\|\check{w}_1^k\|_2}$越趋近于 0，促进稀疏（zero attractor）

C. $L_{2,0}$-norm

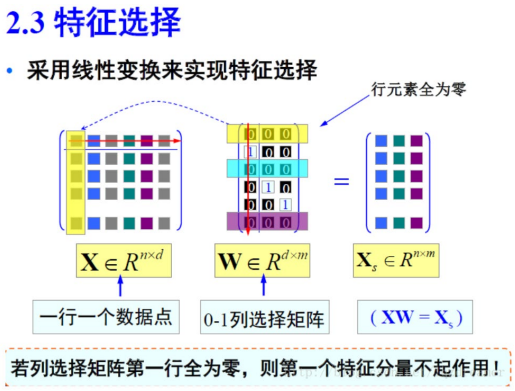
$$\|W_k\|_{2,0} = \left\|\left[\left\|\check{w}_1^k\right\|_2, \ldots, \left\|\check{w}_M^k\right\|_2\right]^T\right\|_0. \quad (15)$$

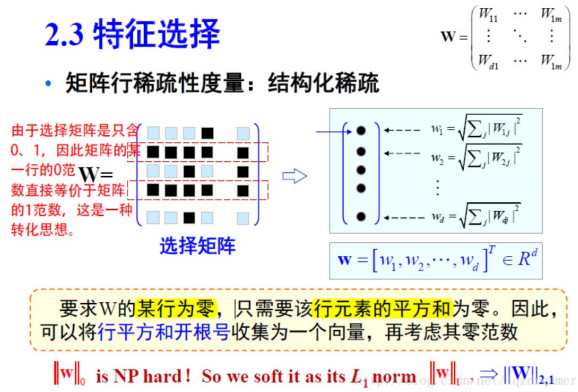
L2，0-norm 非凸：

$$\begin{aligned}\|W_k\|_{2,0} &\approx \sum_{m=1}^{M}\left(1 - \exp\left(-\epsilon\left\|\check{w}_m^k\right\|_2\right)\right) \\ &= \sum_{m=1}^{M}\left(1 - \exp\left(-\epsilon\left(\sum_{l\in N_k}\|w_{ml}\|_2^2\right)^{1/2}\right)\right)\end{aligned} \quad (16)$$

利用上式（16）L2,0-norm 的近似再将指数函数用一阶泰勒展开式展开，得到下面，它的次梯度，

$$f_0(W_k) = \begin{cases}\epsilon\dfrac{w_{mk}}{\|\check{w}_m^k\|_2} - \epsilon^2 w_{mk}, & 0 < \|\check{w}_m\|_2 \le \dfrac{1}{\epsilon} \\ 0, & \text{else.}\end{cases} \quad (17)$$





Verma, V. K., & B, P. R. (2017). **Distributed Multi-task Learning for Sensor Network**. 1, 792–808.

主旨说明：
比较中规中矩的一篇分布式+多任务学习，在 neighbors 间传递**参数 parameter**

目标函数：
全局的：

$$\mathcal{L}(\{\mathbf{w}_k\}_k) = \sum_{k=1}^{n} ||\mathbf{w}_k^\top \mathcal{X}_k - \mathbf{y}_k||_2^2 + \lambda \sum_{k=1}^{n} \sum_{s^{k'} \in \mathcal{S}_k} ||\mathbf{w}_k - \mathbf{w}_{k'}||_2^2 \tag{2}$$

分配到每一个节点的：

$$\mathcal{L}_k(\mathbf{w}_k) = ||\mathbf{w}_k^\top \mathcal{X}_k - \mathbf{y}_k||_2^2 + \lambda \sum_{s_{k'} \in \mathcal{S}_k} ||\mathbf{w}_k - \mathbf{w}_{k'}||_2^2 \tag{3}$$

得到：

$$\mathbf{w}_k = (\mathcal{X}_k^\top \mathcal{X}_k + m_k \lambda \mathcal{I})^{-1} (\mathcal{X}_k^\top \mathbf{y}_k + \lambda \sum_{s_{k'} \in \mathcal{S}_k} \mathbf{w}_{k'}) \tag{4}$$

优化算法：
第 r 次迭代：

$$\mathbf{w}_k^{(r)} = (\mathcal{X}_k^\top \mathcal{X}_k + m_k \lambda \mathcal{I})^{-1} (\mathcal{X}_k^\top \mathbf{y}_k + \lambda \sum_{s_{k'} \in \mathcal{S}_k} \mathbf{w}_{k'}^{(r-1)}) \tag{5}$$

DMTL 算法:

Algorithm 1: DISTRIBUTED MULTI-TASK LEARNING

Input: Feature Data $\{\mathcal{X}_k\}_k$; Target Data $\{\mathbf{y}_k\}_k$
Output: Model Parameters $\{\mathbf{w}_k\}_k$;
// Initialization
forall the $s_k \in \mathcal{S}$ **do**
  $\mathbf{w}_k^{(0)} = (\mathcal{X}_k^\top \mathcal{X}_k)^{-1} \mathcal{X}_k^\top \mathbf{y}_k$;
end
$r = 1$;
// Learning
forall the $s_k \in \mathcal{S}$ **do**
  while $r \leq r_{max}$ **do**
    // broadcast the parameters to neighbor sensors
    forall the $s_{k'}$, $s_k \in \mathcal{S}_{k'}$ **do**
      Send($\mathbf{w}_k^{(r-1)}$);
    end
    // collect the parameters from neighbor sensors
    forall the $s_{k'} \in \mathcal{S}_k$ **do**
      Receive($\mathbf{w}_{k'}^{(r-1)}$);
    end
    // update parameters at iteration r based on the parameters of neighbor sensors at iteration r-1
    $\mathbf{w}_k^{(r)} = (\mathcal{X}_k^\top \mathcal{X}_k + m_k \lambda \mathcal{I})^{-1} (\mathcal{X}_k^\top \mathbf{y}_k + \lambda \sum_{s_{k'} \in \mathcal{S}_k} \mathbf{w}_{k'}^{(r-1)})$;
    $r = r + 1$;
    if $||\mathbf{w}_k^{(r)} - \mathbf{w}_k^{(r-1)}||_2^2 < \theta$ **then**
      break;
    end
  end
end
return $\{\mathbf{w}_k\}_k$

推导过程：

总结报告 15.16 Distributed Multi-task learning for sensor Network

每个点(node)的分布式loss：

$$l_k(w_k) = \|w_k^T x_k - y_k\|_2^2 + \lambda \sum_{s_{k'} \in S_k} \|w_k - w_{k'}\|_2^2 \quad (1)$$

结果：

$$w_k = (x_k^T x_k + m_k \lambda I)^{-1} (x_k^T y_k + \lambda \sum_{s_{k'} \in S_k} w_{k'}) \quad (2)$$

推导：上述问题(1)为最小二乘问题，

令 $\nabla l_k(w_k) = 0$，可得(2)

先将展开(1)式

$$l_k(w_k) = w_k^T x_k^T x_k w_k + y_k^T y_k - 2 w_k^T x_k y_k^T + \lambda \left(\sum w_k^T w_k + \sum w_{k'}^T w_{k'} - 2 \sum w_k^T w_{k'} \right)$$

$$\nabla l_k(w_k) = 2 x_k^T x_k w_k - 2 x_k^T y_k + \lambda \left(\sum 2 w_k - \sum 2 w_{k'} \right) = 0$$

$$(x_k^T x_k + m_k \cdot \lambda I) w_k = (x_k^T y_k + \lambda \sum_{s_{k'} \in S_k} w_{k'})$$

$$\Rightarrow w_k = (x_k^T x_k + m_k \lambda I)^{-1} (x_k^T y_k + \lambda \sum_{s_{k'} \in S_k} w_{k'})$$

（注：这是大三科研其中的一份研究报告，以展示我的科研经历和学术水平）

研究报告 26

Primal-dual Problem

一、理论基础

Source	Details	Remark
张贤达：矩阵分析与应用 P216-P222	考虑标准形式的约束最优化问题 $$\min_{\boldsymbol{x}} f_0(\boldsymbol{x}) \quad \text{subject to } f_i(\boldsymbol{x}) \leqslant 0,\ i=1,\cdots,m;\ \boldsymbol{A}\boldsymbol{x}=\boldsymbol{b} \tag{4.3.1}$$ 或写作 $$\min_{\boldsymbol{x}} f_0(\boldsymbol{x}) \quad \text{subject to } f_i(\boldsymbol{x}) \leqslant 0,\ i=1,\cdots,m; h_i(\boldsymbol{x})=0,\ i=1,\cdots,q \tag{4.3.2}$$ (1) 只有当不等式约束函数 $f_i(\boldsymbol{x}), i=1,\cdots,m$ 均为凸函数，且等式约束函数 $h_i(\boldsymbol{x})$, $i=1,\cdots,q$ 均为仿射函数时，一个原始约束优化问题才能借助 Lagrangian 松弛方法，转换成一个凹函数的对偶无约束极大化问题。 (2) 凹函数的极大化等价于凸函数的极小化。 (3) 若原始约束优化问题的目标函数 $f_0(\boldsymbol{x})$ 不是凸函数，但不等式约束函数 $f_i(\boldsymbol{x}), i=1,\cdots,m$ 均为凸函数，并且等式约束函数 $h_i(\boldsymbol{x}), i=1,\cdots,q$ 均为仿射函数，则 Lagrangian 目标函数满足 KKT 条件的点 $\boldsymbol{x}^\star$ 和 $(\boldsymbol{\lambda}^\star,\boldsymbol{\nu}^\star)$ 一般不会分别是原始最优点和对偶最优点，即 Lagrangian 对偶无约束优化问题的最优解不是原始约束优化问题的最优解，而是 ε-次最优解，其中 $\varepsilon = f_0(\boldsymbol{x}^\star) - J_{\mathrm{D}}(\boldsymbol{\lambda}^\star,\boldsymbol{\nu}^\star)$。 (4) 若 $f_0(\boldsymbol{x})$ 和 $f_i(\boldsymbol{x})$ 均为凸函数，并且等式约束函数 $h_i(\boldsymbol{x})$ 均为仿射函数，即原始约束优化问题为凸优化问题，则 Lagrangian 目标函数满足 KKT 条件的点 $\tilde{\boldsymbol{x}}$ 和 $(\tilde{\boldsymbol{\lambda}},\tilde{\boldsymbol{\nu}})$ 分别是具有零对偶间隙的原始最优点和对偶最优点。换言之，Lagrangian 对偶无约束优化问题的最优解 $\boldsymbol{d}^\star$ 就是原始约束凸优化问题的最优解 $\boldsymbol{p}^\star$。	对于正交约束问题，$\Theta^{\mathrm{T}}\Theta$-I=0，由于它不是仿射函数（大概是说形如 Ax+b，线性函数）的形式，所以，它**不能够**借助 Lagrangian 松弛方法换成一个凹函数的对偶无约束问题。 对于 Loss（f_0（x））它可以不是凸函数。对偶之后仍然是凹函数优化问题，但是这个时候就会有 duality gap，不是原问题的最优解，而是次优解。

Yuxin Chen（2019），Princeton University：Dual and primal-dual methods PPT （接上）	**Problem formulation** $$\min_{\mathbf{x}\in X} f(K\mathbf{x}) + g(\mathbf{x}) \quad \text{(primal)}$$ Recall the convex conjugate: $f^*(\mathbf{y}) = \langle K\mathbf{x}, \mathbf{y}\rangle - f(K\mathbf{x})$, we have: $$\min_{\mathbf{x}\in X}\max_{\mathbf{y}\in Y} \langle K\mathbf{x}, \mathbf{y}\rangle + g(\mathbf{x}) - f^*(\mathbf{y}) \quad \text{(primal-dual)}$$ $$\max_{\mathbf{y}\in Y} -(f^*(\mathbf{y}) + g^*(-K^*\mathbf{y})) \quad \text{(dual)}$$ Dual 的推导： $$\text{maximize}_{\boldsymbol{\lambda}} \quad \min_{\boldsymbol{x},\boldsymbol{z}} \; f(\boldsymbol{x}) + h(\boldsymbol{z}) + \langle \boldsymbol{\lambda}, \boldsymbol{A}\boldsymbol{x} - \boldsymbol{z}\rangle$$ $$\Updownarrow \text{ decouple } \boldsymbol{x} \text{ and } \boldsymbol{z}$$ $$\text{maximize}_{\boldsymbol{\lambda}} \quad \min_{\boldsymbol{x}} \left\{\langle \boldsymbol{A}^\top\boldsymbol{\lambda}, \boldsymbol{x}\rangle + \; f(\boldsymbol{x})\right\} + \min_{\boldsymbol{z}} \{h(\boldsymbol{z}) - \langle \boldsymbol{\lambda}, \boldsymbol{z}\rangle\}$$ $$\Updownarrow$$ $$\text{maximize}_{\boldsymbol{\lambda}} \quad -f^*(-\boldsymbol{A}^\top\boldsymbol{\lambda}) - h^*(\boldsymbol{\lambda})$$ where f^* (resp. h^*) is the Fenchel conjugate of f (resp. h) Primal-dual 的推导： $$\text{minimize}_{\boldsymbol{x}} \quad f(\boldsymbol{x}) + h(\boldsymbol{A}\boldsymbol{x})$$ $$\Updownarrow \text{ add an auxiliary variable } \boldsymbol{z}$$ $$\text{minimize}_{\boldsymbol{x},\boldsymbol{z}} \quad f(\boldsymbol{x}) + h(\boldsymbol{z}) \quad \text{subject to } \boldsymbol{A}\boldsymbol{x} = \boldsymbol{z}$$ $$\Updownarrow$$ $$\text{maximize}_{\boldsymbol{\lambda}} \; \min_{\boldsymbol{x},\boldsymbol{z}} \; f(\boldsymbol{x}) + h(\boldsymbol{z}) + \langle \boldsymbol{\lambda}, \boldsymbol{A}\boldsymbol{x} - \boldsymbol{z}\rangle$$ $$\Updownarrow$$ $$\text{maximize}_{\boldsymbol{\lambda}} \; \min_{\boldsymbol{x}} \; f(\boldsymbol{x}) + \langle \boldsymbol{\lambda}, \boldsymbol{A}\boldsymbol{x}\rangle - h^*(\boldsymbol{\lambda})$$ $$\Updownarrow$$ $$\text{minimize}_{\boldsymbol{x}} \; \max_{\boldsymbol{\lambda}} \; f(\boldsymbol{x}) + \langle \boldsymbol{\lambda}, \boldsymbol{A}\boldsymbol{x}\rangle - h^*(\boldsymbol{\lambda}) \quad \text{(saddle-point problem)}$$	从推导可以看出，是否把 f（x）+<λ，Ax>结合起来写成共轭的形式，就决定了最后到底产生的是 dual 形式还是 primal dual 形式。 Dual 形式，只有 λ 一个变量，很多文献也给出了 λ 转化成 x 的公式； 而 primal-dual 形式，有 x 和 λ 两个变量，也就是一个鞍点问题。 这个 PPT 当中也给出了 dual 和 primal-dual 的 prox 算法，原始-对偶在每一次迭代中，原始变量和对偶变量都要更新，而 dual 可以只更新对偶变量，最后根据等式转换成原始变量解，（~~当然也可以 x 和 λ 都更新；delete 说的不准确~~）。 这个 PPT 中 dual 和 primal-dual methods ~~本质（delete，说的不准确）~~区别在于是否原始变量和对偶变量的迫近算子都用到了。 这段话其实是在实际操作过程中，到底选择 dual 和 primal-dual 哪种形式去做呢？这段话给出了一个指导。当原始变量和对偶变量的迫近算子都比较好算（cheap in compution），那么可以选择 primal-dual 形式。 目标函数是：

	(primal) minimize$_{\boldsymbol{x}}$ $f(\boldsymbol{x})+h(\boldsymbol{A}\boldsymbol{x})$ (dual) minimize$_{\boldsymbol{\lambda}}$ $f^*(-\boldsymbol{A}^\top\boldsymbol{\lambda})+h^*(\boldsymbol{\lambda})$ Dual formulation is useful if • the proximal operator w.r.t. h is cheap (then we can use the Moreau decomposition $\text{prox}_{h^*}(\boldsymbol{x})=\boldsymbol{x}-\text{prox}_h(\boldsymbol{x})$) • f^* is smooth (or if f is strongly convex) Can we update both primal and dual variables simultaneously and take advantage of both prox_f and prox_h?也就是这种情况利用 primal-dual problem $$\text{minimize}_{\boldsymbol{x}}\ \max_{\boldsymbol{\lambda}}\ f(\boldsymbol{x})+\langle\boldsymbol{\lambda},\boldsymbol{A}\boldsymbol{x}\rangle-h^*(\boldsymbol{\lambda}) \quad (9.3)$$	minimize$_{\boldsymbol{x}}$ $f(\boldsymbol{x})+h(\boldsymbol{A}\boldsymbol{x})$ where f and h are convex **Primal-dual algorithm** The algorithm[6] $$\min_{\mathbf{x}\in X}\max_{\mathbf{y}\in Y}\langle K\mathbf{x},\mathbf{y}\rangle+g(\mathbf{x})-f^*(\mathbf{y})$$ ▸ Choose step size $\sigma>0$ and $\tau>0$, so that $\sigma\tau L^2<1$, where $L=\|K\|$, and $\theta\in[0,1]$. ▸ Choose initialization $(\mathbf{x}^0,\mathbf{y}^0)$ ▸ For each iteration: $$\begin{cases}\mathbf{y}^{(n+1)} &= \mathbf{prox}_{f^*}(\mathbf{y}^{(n)}+\sigma K\bar{\mathbf{x}}^{(n)}) & \text{(dual proximal)}\\ \mathbf{x}^{(n+1)} &= \mathbf{prox}_{g}(\mathbf{x}^{(n)}-\tau K^*\bar{\mathbf{y}}^{(n+1)}) & \text{(primal proximal)}\\ \bar{\mathbf{x}}^{(n+1)} &= \mathbf{x}^{(n+1)}+\theta(\mathbf{x}^{(n+1)}-\mathbf{x}^{(n)}) & \text{(extrapolation)}\end{cases}$$ ▸ Essentially alternately do proximal gradient descent for $\mathbf{x}$ and $\mathbf{y}$. [6]Chambolle and Pock (2011), Pock, Cremers, Bischof, Chambolle (2009) 20 / 34
TinTin 博客：Primal-dual problem （Posted by Tintin on April 20, 2019） https://tintin.space/2019/04/20/Primal/	Primal：$$\min_{x\in R^d}\Phi(x)+\frac{1}{n}\sum_{i=1}^{n}f_i(a_i x) \quad (1)$$ 1.如何选择Primal、Primal-Dual和Dual问题? • Case1: $\Phi(x)$强凸$f(x)$光滑，可以选择Primal或Dual，不推荐Primal-Dual，因为它需要调整两个学习率. • Case2: $\Phi(x)$强凸$f(x)$不光滑，对偶目标函数是光滑非强凸，推荐Dual问题不推荐原始问题。 • Case3：原始问题中$\Phi(x)$非强凸$f(x)$光滑，推荐Primal不推荐Dual，因为对应的Dual的目标函数是不光滑强凸的，而coordinate descent要求函数光滑，如果要用需要先 • Case4：Primal和Dual的目标函数都是非强凸非光滑的，推荐Primal-Dual。 • 其它更复杂的情况，如$f(x)$是非凸，或者不是$f(a_i^T x)$的形式，推荐使用Primal而不用Dual，因为Dual问题可能会变得十分高维。	其中，$\Phi(x)$ 是 regularizer，f(x)是分布式的 loss

百度文库：对偶和鞍点问题 PPT	**鞍点定理：** (1)设$(\bar{x},\bar{w},\bar{v})$是原问题的*Lagrange*函数$L(x,w,v)$的鞍点，则$\bar{x}$和$(\bar{w},\bar{v})$分别是原问题和对偶问题的最优解。(2)假设$f$是凸函数，$g_i(x)(i=1,\cdots,m)$是凹函数，$h_j(x)(j=1,\cdots,l)$是线性函数，即$h(x)=A(x)-b$，且$A$是满秩矩阵，又设存在$\hat{x}$，使$g(\hat{x})>0$，$h(\hat{x})=0$，如果$\bar{x}$是原问题的最优解，则存在$(\bar{w},\bar{v})(\bar{w}\geq 0)$，使$(\bar{x},\bar{w},\bar{v})$是*Lagrange*函数的鞍点。	**鞍点与KKT条件之间的关系** **定理：** $\min\{f(x)\mid g(x)\geq 0,h(x)=0\}$中，可行域为$S$，$\bar{x}\in S$满足*KKT*条件，即存在$\bar{w}\geq 0$，$\bar{v}$使 $\nabla f(\bar{x})-w^T\nabla g(\bar{x})-v^T\nabla h(\bar{x})=0,$ 且f为凸函数，$g_i(i\in I)$为凹函数，h_j为线性函数，则$(\bar{x},\bar{w},\bar{v})$为*Lagrange*函数$L(x,w,v)$的鞍点；反之，设$f,g_i,h_j$可微，若$(\bar{x},\bar{w},\bar{v})(\bar{w}\geq 0)$是*Lagrange*函数的鞍点，则$(\bar{x},\bar{w},\bar{v})$满足*KKT*条件。	鞍点定理其实和第一行张贤达说的是一个道理，要转换成鞍点问题，约束要满足凸函数、线性函数的条件； 鞍点与 KKT 条件之间的关系表明，满足 KKT 条件的点就是鞍点。

二、论文

Title	Objective functions	Note (The property of regularizer)
Shalev-Shwartz, S., & Zhang, T. (2013). **Stochastic Dual Coordinate Ascent methods for regularized loss minimization**. Journal of Machine Learning Research, 14(1), 567–599.	Primal: $P(w)=\left[\frac{1}{n}\sum_{i=1}^{n}\phi_i(w^\top x_i)+\frac{\lambda}{2}\|w\|^2\right].$ Dual: $\max_{\alpha\in\mathbb{R}^n} D(\alpha)$ where $D(\alpha)=\left[\frac{1}{n}\sum_{i=1}^{n}-\phi_i^*(-\alpha_i)-\frac{\lambda}{2}\left\|\frac{1}{\lambda n}\sum_{i=1}^{n}\alpha_i x_i\right\|^2\right]$	正则项：L2-norm
Yang, T. (2013). **Trading computation for communication: Distributed stochastic dual coordinate ascent**. Advances in Neural Information Processing Systems, 1–9.	Primal: $\min_{w\in\mathbb{R}^d} P(w),$ where $P(w)=\frac{1}{n}\sum_{i=1}^{n}\phi(w^\top x_i;y_i)+\lambda g(w),$ (1) $g(w)$ is a 1-strongly convex function w.r.t to $\|\cdot\|_2$. Examples include ℓ_2 norm square $1/2\|w\|_2^2$ and elastic net $1/2\|w\|_2^2+\mu\|w\|_1$. Dual: $\max_{\alpha\in\mathbb{R}^n} D(\alpha),$ where $D(\alpha)=\frac{1}{n}\sum_{i=1}^{n}-\phi_i^*(-\alpha_i)-\lambda g^*\left(\frac{1}{\lambda n}\sum_{i=1}^{n}\alpha_i x_i\right).$ (2)	正则项：g(w)要求是凸函数，但不要求光滑。比如，g（w）中含有 L1-norm 弹性网的共轭函数可以求，见文档最后

Zheng, S., Wang, J., Xia, F., Xu, W., & Zhang, T. (2017). **A general distributed dual coordinate optimization framework for regularized loss minimization.** Journal of Machine Learning Research, 18, 1–52.	Primal：$$\min_{w\in\mathbb{R}^d}\left[P(w):=\sum_{i=1}^{n}\phi_i(X_i^\top w)+\lambda n g(w)+h(w)\right], \quad (1)$$ 分布式原始：$$\min_{w;\{w_\ell\};\{u_i\}}\sum_{\ell=1}^{m}\left[\sum_{i\in S_\ell}\phi_i(u_i)+\lambda n_\ell g(w_\ell)\right]+h(w) \quad (5)$$ $$s.t\quad u_i=X_i^\top w_\ell,\ \text{for all } i\in S_\ell$$ $$w_\ell=w,\ \text{for all } \ell\in\{1,...,m\}.$$ **Proposition 1** *Define the dual objective as* 分布式对偶：$$D(\alpha,\beta):=\sum_{\ell=1}^{m}\left[\sum_{i\in S_\ell}-\phi_i^*(-\alpha_i)-\lambda n_\ell g^*\left(\frac{\sum_{i\in S_\ell}X_i\alpha_i-\beta_\ell}{\lambda n_\ell}\right)\right]-h^*\left(\sum_\ell\beta_\ell\right).$$	g(w) is a strongly convex regularizer,h(w) is another convex regularizer 比如： $g(w)=\|w\|_2^2+a\|w\|_1$, and $h(w)=b\|w\|_1$
Zhang, Y., & Xiao, L. (2017). **Stochastic primal-dual coordinate method for regularized empirical risk minimization**. Journal of Machine Learning Research, 18, 1–42.	$$\underset{x\in\mathbb{R}^d}{\text{minimize}}\quad\left\{P(x)\ \stackrel{\text{def}}{=}\ \frac{1}{n}\sum_{i=1}^{n}\phi_i(a_i^T x)+g(x)\right\}. \quad (1)$$ convex-concave saddle point problem $$\min_{x\in\mathbb{R}^d}\ \max_{y\in\mathbb{R}^n}\left\{f(x,y)\ \stackrel{\text{def}}{=}\ \frac{1}{n}\sum_{i=1}^{n}\left(y_i\langle a_i,x\rangle-\phi_i^*(y_i)\right)+g(x)\right\}. \quad (4)$$ **Stochastic Primal-Dual Coordinate (SPDC)method：** basic idea: to approach the saddle point of f(x, y) defined in (4), we alternatively maximize f with respect to y, and minimize f with respect to x. Since the dual vector y has n coordinates and each coordinate is associated with a feature vector ai ∈ R^d, maximizing f with respect to y takes O(nd) computation, which can be very expensive if n is large. We reduce the computational cost by **randomly picking a single coordinate of y at a time, and maximizing f only with respect to this coordinate**. Consequently, the computation of each iteration is O(d).	g（x）凸

Chambolle, A., & Pock, T. (2011). **A first-order primal-dual algorithm for convex problems with applications to imaging. Journal of Mathematical Imaging and Vision**, 40(1), 120–145. https://doi.org/10.1007/s10851-010-0251-1	Chambolle and Pock (2011) considered a class of convex optimization problems with the following saddle-point structure: $$\min_{x\in\mathbb{R}^d}\max_{y\in\mathbb{R}^n}\{\langle Kx,y\rangle+G(x)-F^*(y)\},\tag{27}$$ where $K\in\mathbb{R}^{m\times d}$, G and F^* are proper closed convex functions, with F^* itself being the conjugate of a convex function F. They developed the following first-order primal-dual algorithm: $$y^{(t+1)}=\arg\max_{y\in\mathbb{R}^n}\left\{\langle K\overline{x}^{(t)},y\rangle-F^*(y)-\frac{1}{2\sigma}\|y-y^{(t)}\|_2^2\right\},\tag{28}$$ $$x^{(t+1)}=\arg\min_{x\in\mathbb{R}^d}\left\{\langle K^Ty^{(t+1)},x\rangle+G(x)+\frac{1}{2\tau}\|x-x^{(t)}\|_2^2\right\},\tag{29}$$ $$\overline{x}^{(t+1)}=x^{(t+1)}+\theta(x^{(t+1)}-x^{(t)}).\tag{30}$$ When both F^* and G are strongly convex and the parameters τ, σ and θ are chosen appropriately, this algorithm obtains accelerated linear convergence rate (Chambolle and Pock, 2011, Theorem 3).	正则项要求凸。
鞍点问题和约束优化的几个一阶算法 尤燕飞（南京大学博士论文 2015）	**We consider a saddle-point problem** $$\min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}}\Phi(x,y):=\theta_1(x)-y^{\top}Ax-\theta_2(y),\tag{3.1}$$ **where $A\in\mathcal{R}^{m\times n}$, $\mathcal{X}\subseteq\mathcal{R}^n$, $\mathcal{Y}\subseteq\mathcal{R}^m$ are closed convex sets, $\theta_1:\mathcal{R}^n\to\mathcal{R}$ and $\theta_2:\mathcal{R}^m\to\mathcal{R}$ are convex, but not necessarily smooth functions. The solution set of (3.1) is assumed to be nonempty.** **Slightly extending the original PDHG scheme in [105] to the model (3.1), we obtain the scheme:** $$\begin{cases}x^{k+1} &= \arg\min\{\Phi(x,y^k)+\frac{r}{2}\|x-x^k\|^2\,|\,x\in\mathcal{X}\},\\ y^{k+1} &= \arg\max\{\Phi(x^{k+1},y)-\frac{s}{2}\|y-y^k\|^2\,|\,y\in\mathcal{Y}\},\end{cases}\tag{3.2}$$	凸，非光滑。

Tan, C. (2018). **Stochastic Primal-Dual Method for Empirical Risk Minimization with O（1）Per-Iteration Complexity**. Advances in Neural Information Processing Systems 31 (NIPS 2018).1, 1–10.	$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathcal{X}} \; f(\mathbf{x}) + h(\mathbf{A}\mathbf{x}). \quad (2)$$ It is equivalent to the following saddle-point problem $$\min_{\mathbf{x}\in\mathcal{X}} \max_{\mathbf{s}\in\mathcal{S}} \; f(\mathbf{x}) + \langle \mathbf{A}\mathbf{x}, \mathbf{s}\rangle - h^*(\mathbf{s}). \quad (3)$$ One iteration of the algorithm is $$\mathbf{s}^{k+1} \in \left(\frac{\gamma}{\lambda}\mathbf{I} + \partial h^*\right)^{-1} \left(\frac{\gamma}{\lambda}(\mathbf{I} - \lambda\mathbf{A}\mathbf{A}^\top)\mathbf{s}^k + \mathbf{A}\left(\mathbf{x}^k - \gamma\nabla f(\mathbf{x}^k)\right)\right), \quad (4a)$$ $$\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma\nabla f(\mathbf{x}^k) - \gamma\mathbf{A}^\top\mathbf{s}^{k+1}. \quad (4b)$$	有一类算法的目标函数为 f（w）+g（Aw），这里的 A 可以 data matrix.

三、summary

总的来说，对于应用 primal-dual，因为 Loss 应该就是 ERM，那么对于 regularizer 来说，都是要求是凸的，但是可以不光滑，比如 L2-norm，L1-norm,至于非凸的目前还没有看到。根据张贤达的矩阵分析与应用，应该这种正交约束难以用对偶的方法解决，但是，一些论文给出了正交约束的算法的迭代步骤。也就是当单独优化 Θ：$\min_{\theta} F(\theta) \;\; s.t. \;\; \theta^T\theta = I$，这种算法是有一些的。

对于 regularizer 来说，有的文章用核范数，谱范数来做正则，那么这两个范数如何做分布式？如何做原始对偶？他们的共轭函数的形式是什么样的？即：是否可以做出类似如下问题的分布式算法？原始对偶优化算法？

$$\min \sum f(\mathrm{w_i x_i} - \mathrm{y_i}) + \| W \|_*$$

四、response

4.1 primal 问题是长成这样：min x；

dual 问题是长成这样：max λ；

primal-dual 问题长成这样：min x max λ

1）这两种形式本质上是完全一致的，只不过是根据之后想要设计的算法来选择到底选择哪一种形式（是想同时 update x 和 λ，还是只 update λ）。

2）要说这两形式的区别就在于推导的时候是否把 f（x）+<λ，Ax>结合起来写成共轭的形式。

3）两者共存完全是因为选择了 primal-dual 形式导致的，一旦选择 primal-dual 形式就无条件满足同时 update 两者。

4.2 首先要区别**对偶范数**和**范数的共轭函数**两个的概念

• 对偶范数:

定义 向量范数||x||的对偶范数为||y||*

$$\|\boldsymbol{y}\|^* = \sup_{\|\boldsymbol{x}\| \leqslant 1} \boldsymbol{y}^{\mathrm{T}}\boldsymbol{x} \tag{4.5.19}$$

因此，对偶范数 $\|\boldsymbol{y}\|^*$ 是单位球范数 $\|\boldsymbol{x}\| \leqslant 1$ 的支撑函数。

• 范数的共轭函数：

定义

(4) 向量范数 $g(\boldsymbol{x}) = \|\boldsymbol{x}\|$ 的共轭函数 $g^*(\boldsymbol{y}) = \|\boldsymbol{y}\|^*$ 是对偶单位范数球 $\|\boldsymbol{y}\|^* \leqslant 1$ 上的指示函数

$$g^*(\boldsymbol{y}) = \sup_{\boldsymbol{x}}(\boldsymbol{y}^{\mathrm{T}}\boldsymbol{x} - \|\boldsymbol{x}\|) = \begin{cases} 0, & \|\boldsymbol{y}\|^* \leqslant 1 \\ +\infty, & \text{其他} \end{cases} \tag{4.5.20}$$

举例：

Example 3.26 *Norm.* Let $\|\cdot\|$ be a norm on $\mathbf{R}^n$, with dual norm $\|\cdot\|_*$. We will show that the conjugate of $f(x) = \|x\|$ is

$$f^*(y) = \begin{cases} 0 & \|y\|_* \leq 1 \\ \infty & \text{otherwise,} \end{cases}$$

i.e., the conjugate of a norm is the indicator function of the dual norm unit ball.

If $\|y\|_* > 1$, then by definition of the dual norm, there is a $z \in \mathbf{R}^n$ with $\|z\| \leq 1$ and $y^T z > 1$. Taking $x = tz$ and letting $t \to \infty$, we have

$$y^T x - \|x\| = t(y^T z - \|z\|) \to \infty,$$

which shows that $f^*(y) = \infty$. Conversely, if $\|y\|_* \leq 1$, then we have $y^T x \leq \|x\|\|y\|_*$ for all x, which implies for all x, $y^T x - \|x\| \leq 0$. Therefore $x = 0$ is the value that maximizes $y^T x - \|x\|$, with maximum value 0.

下面是三组常用的向量范数–对偶向量范数对

$$(\|\boldsymbol{x}\|_2, \|\boldsymbol{y}\|_2), \quad (\|\boldsymbol{x}\|_1, \|\boldsymbol{y}\|_\infty), \quad \left(\sqrt{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x}}, \sqrt{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{Q}^{-1}\boldsymbol{y}}\right) \quad (\boldsymbol{Q} \text{ 正定})$$

举例：带向量范数的对偶问题

Equality constrained norm minimization

Consider the problem

$$\begin{array}{ll} \text{minimize} & \|x\| \\ \text{subject to} & Ax = b, \end{array} \tag{5.12}$$

where $\|\cdot\|$ is any norm. Recall (from example 3.26 on page 93) that the conjugate of $f_0 = \|\cdot\|$ is given by

$$f_0^*(y) = \begin{cases} 0 & \|y\|_* \leq 1 \\ \infty & \text{otherwise,} \end{cases}$$

the indicator function of the dual norm unit ball.

Using the result (5.11) above, the dual function for the problem (5.12) is given by

$$g(\nu) = -b^T\nu - f_0^*(-A^T\nu) = \begin{cases} -b^T\nu & \|A^T\nu\|_* \leq 1 \\ -\infty & \text{otherwise.} \end{cases}$$

以上说的都是针对向量的范数，对于矩阵范数：

两组常用的矩阵范数–对偶矩阵范数对

$$(\|\boldsymbol{X}\|_{\mathrm{F}},\ \|\boldsymbol{Y}\|_{\mathrm{F}}),\quad \left(\|\boldsymbol{X}\|_2 = \sigma_{\max}(\boldsymbol{X}),\ \|\boldsymbol{Y}\|_* = \sum_{i=1}^{n}\sigma_i(\boldsymbol{X})\right)$$

矩阵核范数的对偶范数是谱范数。

弹性网的对偶函数：

这个例子说明弹性网的对偶函数是可解的。

We now introduce the Lasso Dual problem to see why they are useful and how they come up. We will derive the lasso dual in this section. We are going to see a couple of things in this example. The first thing is the dual trick, and the second thing is the use of conjugate. The lasso problem is:

$$min_\beta \frac{1}{2}\|y - x\beta\|_2^2 + \lambda\|\beta\|_1 \tag{13.1}$$

How many variables does the dual function take? What is the dimension? It's a little tricky for there is no constraints for the function. It's equal to the minimum of the lasso function over all x. The second term of the constraints is 0 so there is no constraints. We are going to introduce some axiliant variables here to create fake constraints, which will give us dual variables to derive the dual. This is our first dual trick. You can do this in more than one way. Let's rewrite it in the following way.

$$\Leftrightarrow min_{z,\beta} \frac{1}{2}\|y - z\|_2^2 + \lambda\|\beta\|_1 \qquad s.t. \quad X\beta = z \tag{13.2}$$

Now the Lagrangian is as follows. Our primal variables are z and β. The dual variable is u.

$$L(z, \beta, u) = \frac{1}{2}\|y - z\|_2^2 + \lambda\|\beta\|_1 + u^T(z - X\beta) \tag{13.3}$$

Let's minimize that over all z and β to get the dual function to be the function of u. So we want to minimize the Lagrangian over all z and β by breaking it up as follows:

$$min_{z,\beta}\, L(z, \beta, u) = min_z\{\frac{1}{2}\|y - z\|_2^2 + u^T z\} + min_\beta\{\lambda\|\beta\|_1 - (X^T u)^T\beta\} \tag{13.4}$$

Minimize the first part over all z and take the gradient and let it equal to 0. We will get the minimizer here to be (y-u), and we plug this in the first part. The second part may be tricky, because it involves the L1 norm which is not differentiable. We rewrite it into the conjugate function:

$$\frac{1}{2}\|y\|_2^2 - \frac{1}{2}\|y - u\|_2^2 - \lambda max_\beta\{\frac{(X^T u)^T}{\lambda}\beta - \|\beta\|_1\} \tag{13.5}$$

So we get the dual function as follows:

$$\frac{1}{2}\|y\|_2^2 - \frac{1}{2}\|y-u\|_2^2 - I\{\|X^T u\|_\infty \le \lambda\} \tag{13.6}$$

The lasso dual is to maximize (6) over all u, which is the same as (7)

$$min_u \frac{1}{2}\|y-u\|_2^2 \qquad s.t. \quad \|X^T u\|_\infty \le \lambda \tag{13.7}$$

(7) is the lasso dual. It is easy because we use the conjugate trick during derivation. So after introducing the constrain, we write down the lasso's Lagrangian. Minimize it to get the dual function, and end up with the following problem:

$$max_u \frac{1}{2}(\|y\|_2^2 - \|y-u\|_2^2) \qquad s.t. \quad \|X^T u\|_\infty \le \lambda \tag{13.8}$$

Let's think about the first point we made, which is that we can also get primal solutions from dual solutions. We can see the strong duality holds here. It does because if we go back to the primal problem, we have to look at (2). Because it is the problem whose dual we took with the fake constrain z. The strong duality holds between this problem and its dual, because it's equivalent to that problem. All we have to do it to find z and β such that $z = X\beta$. Slaters condition holds, and hence so does strong duality. The lasso's dual attains the same objective value as the lasso's primal does. But we should be careful here. If we maximize (7) over all u, we'll get $g(u^*) = f(x^8)$. It does not mean that if I minimize (6) over all u, the criteria value will equal to the f(*). (6) is just the transform version of the dual. (6) and (7) do not have the same criteria value. the optimal value of (6) is not the optimal lasso objective value.

注：

1）基本上查找了关于范数的对偶和共轭问题的资料，现在绝大部分都围绕着**向量范数**进行讨论的。对于矩阵范数的讨论，基本上只有这两个

两组常用的矩阵范数–对偶矩阵范数对

$$(\|\boldsymbol{X}\|_{\mathrm{F}}, \|\boldsymbol{Y}\|_{\mathrm{F}}), \quad \left(\|\boldsymbol{X}\|_2 = \sigma_{\max}(\boldsymbol{X}), \|\boldsymbol{Y}\|_* = \sum_{i=1}^{n} \sigma_i(\boldsymbol{X})\right)$$

而且，这仅仅是矩阵范数的对偶范数，如果想要去求其共轭函数，得出的结果也比较繁琐。并且目前还没有关于**矩阵范数**的共轭函数讨论的资料和实例，包括对于上面弹性网的实例，它也是定义在 w（共识问题），而不是 W 上的。

2）上面的论文，除了最后三篇，实际上都是 DCA（Dual Coordinate Asent）算法的延伸和改进（***方法上的改进***），比如让它 Stochastic，收敛的更快等等。所以，它们的论文当中，仅仅是对于方法上比较抽象的，比如他说 regularizer g（x）是光滑的还是非光滑，会导致什么样的效果，它还没有给出一个非常具体的应用，比如还没有明确的给出一个目标函数长什么样，然后应该怎么推导。实验部分，也仅仅是非常简单的，比如针对只有 L1-norm 或者 L2-norm 的 SVM 问题，几种方法（比如和以前的 SGD）比较一下效果怎么样之类的。（也就是现在的论文是对于 DCA 方法的改进和讨论，还没有实际应用）

3）基于 2），上面论文中给出的目标函数，可能用原始方法直接优化就十分简单，但是它可能为了讨论这种方法，就用对偶去做。至于给出一个具体的目标函数的时候是否满足它给的如下的比较抽象的条件（比如 λ 小、没有终止条件）从而用它的方法，就还得具体分析。

原问题直接用 Stochastic gradient descent(SGD)可以解，但是 SGD approach has several disadvantages. It does not have a clear stopping criterion; it tends to be too aggressive at the beginning of the optimization process, especially when λ is very small; while SGD reaches a moderate accuracy quite fast, its convergence becomes rather slow when we are interested in more accurate solutions.

So an alternative approach is dual coordinate ascent (DCA)

DCA 是更早期的文章提出的，这篇论文提出的叫 SDCA（Stochastic dual coordinate ascent），是对 DCA 的改进。

The purpose of this paper is to develop theoretical understanding of the convergence of the duality gap for SDCA.

4）有一个视频是 ICML 2017 Tutorial：**Zeyuan Allen-Zhu 朱泽园：Recent Advances in Stochastic Convex and Non-convex**，这里面讲的 primal 和 dual 和 primal-dual 的若干论文方法一个总结，我明后两天可能要好好看看，以及他列出的论文的 list。
https://www.bilibili.com/video/BV1M441177kK

（注：这是大三科研其中的一份研究报告，以展示我的科研经历和学术水平）

研究报告 27

朱泽园 tutorial：Recent advances in Stochastic Conex & Non-Convex Optimization

1. Tutorial 概述

这个 tutorial 从三个维度来讲的，即下图中的三个坐标轴

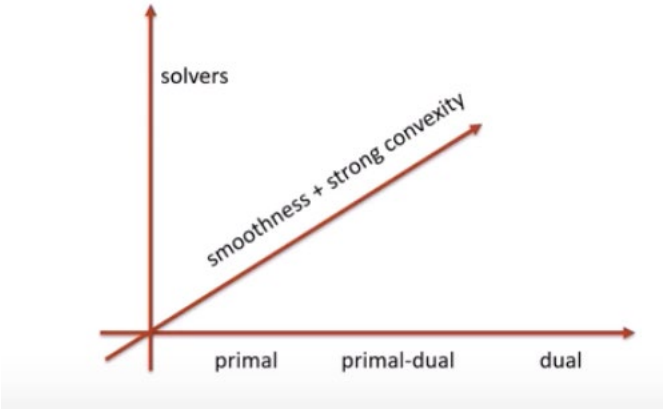
（1）primal & dual & primal-dual

（2）smoothness + strong convexity

（3）solvers

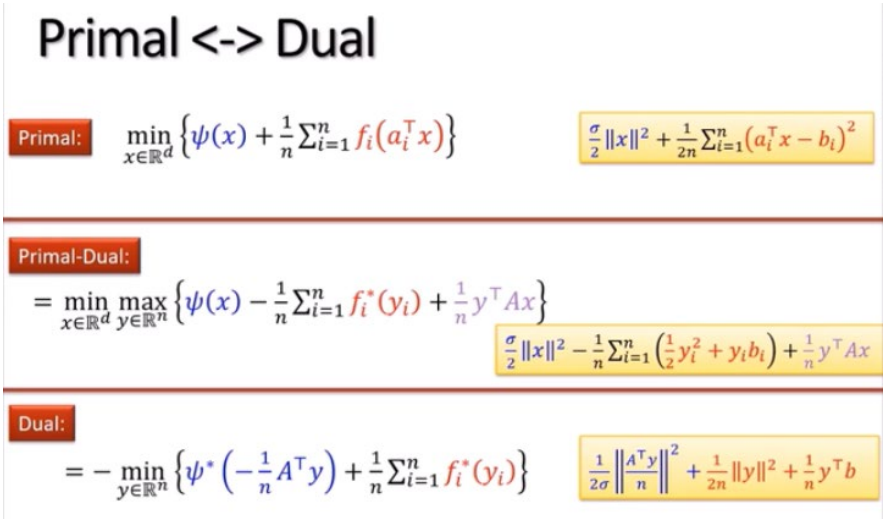
这些方法的目标函数都形如下式： 其中 $\varphi(x)$ 为 regularizer，f(x)为 loss function（ERM）

$$\min_{x\in\mathbb{R}^d}\left\{\psi(x)+\frac{1}{n}\sum_{i=1}^{n}f_i(a_i^\top x)\right\}$$



2. 下面针对这三个维度一次介绍：

（1）primal & dual & primal-dual



下面是对于 primal、dual、primal-dual 问题的推导：

Recent Advances in Stochastic Convex & Non-Convex Optimization

$\min_{x\in\mathbb{R}^d}\{\mathcal{Q}(x)+\frac{1}{n}\sum_{i=1}^n f_i(x)\}$ $\begin{cases}\mathcal{Q}(x)-\text{a convex regularizer}\\ f_i(x)-\text{a convex loss function}\end{cases}$

$\min_{x\in\mathbb{R}^d}\{\mathcal{Q}(x)+\frac{1}{n}\sum_{i=1}^n f_i(a_i^T x)\}$

$g(x)$: convex function

- Fenchel dual : $g^*(y):=\max_{x\in\mathbb{R}^d}\{y^Tx-g(x)\}$ for $y\in\mathbb{R}^d$.

  $g(x)=\frac{1}{p}\|x\|_p^p$ $g^*(y)=\frac{1}{q}\|y\|_q^q$ $\frac{1}{p}+\frac{1}{q}=1$

- dual theorem $g^{**}(x)=g(x)$

- properties : ① $g^*(y)$ is (proper) convex

  ② $g(x)$ is L-smooth $\Leftrightarrow$ $g^*(y)$ is $\frac{1}{L}$-SC

  def $f(x)$ is L-smooth if $\nabla^2 f(x)\le L\cdot I$

  $f(x)$ is σ-SC if $\nabla^2 f(x)\succeq\sigma\cdot I$

  ③ $g(x)$ is σ-SC $\Leftrightarrow$ $g^*(y)$ is $\frac{1}{\sigma}$-smooth

SC: Strongly Convex

Primal $\Leftrightarrow$ Dual

- Primal $\min_{x\in\mathbb{R}^d}\{\mathcal{Q}(x)+\frac{1}{n}\sum_{i=1}^n \underbrace{f_i(a_i^Tx)}_{\frac{1}{L}\text{-SC}}\}$ χ: 数据

  $=\min_{x\in\mathbb{R}^d}\max_{y\in\mathbb{R}^n}\{\mathcal{Q}(x)+\frac{1}{n}\sum_{i=1}^n(y_i\cdot a_i^Tx-f_i^*(y_i))\}$ A: input data matrix

  a_i 第 i 行

- Dual $=\min_{x\in\mathbb{R}^d}\max_{y\in\mathbb{R}^n}\{\underbrace{\mathcal{Q}(x)}_{\sigma\text{-SC}}-\frac{1}{n}\sum_{i=1}^n f_i^*(y_i)+\frac{1}{n}y^TAx\}$

  $=-\min_{y\in\mathbb{R}^n}\max_{x\in\mathbb{R}^d}\{-\mathcal{Q}(x)+\frac{1}{n}\sum_{i=1}^n f^*(y_i)-\frac{1}{n}y^TAx\}$

Dual $=-\min_{y\in\mathbb{R}^n}\{\underbrace{\mathcal{Q}^*(-\frac{1}{n}A^Ty)}_{\frac{1}{\sigma}\text{-smooth}}+\frac{1}{n}\sum_{i=1}^n\underbrace{f^*(y_i)}_{\frac{1}{L}\text{-SC}}\}$

Smooth - SC

（2）smoothness + strong convexity

根据共轭函数的性质：

- $g(x)$ is L-smooth $\Leftrightarrow$ $g^*(y)$ is $\frac{1}{L}$ - strongly convex
- $g(x)$ is σ-SC $\Leftrightarrow$ $g^*(y)$ is $\frac{1}{\sigma}$ -smooth

那么,实际上目标函数总共可以分为四个 cases：

$$\min_{x\in\mathbb{R}^d}\left\{\psi(x)+\frac{1}{n}\sum_{i=1}^n f_i(a_i^\top x)\right\}$$

$\psi(\cdot)$ is σ-SC　　$f_i(\cdot)$ is L-smooth

$\sigma>0,\ L<+\infty$ (SC and smooth)
$\sigma>0,\ L=+\infty$ (SC and non-smooth)
$\sigma=0,\ L<+\infty$ (non-SC and smooth)
$\sigma=0,\ L=+\infty$ (non-SC and non-smooth)

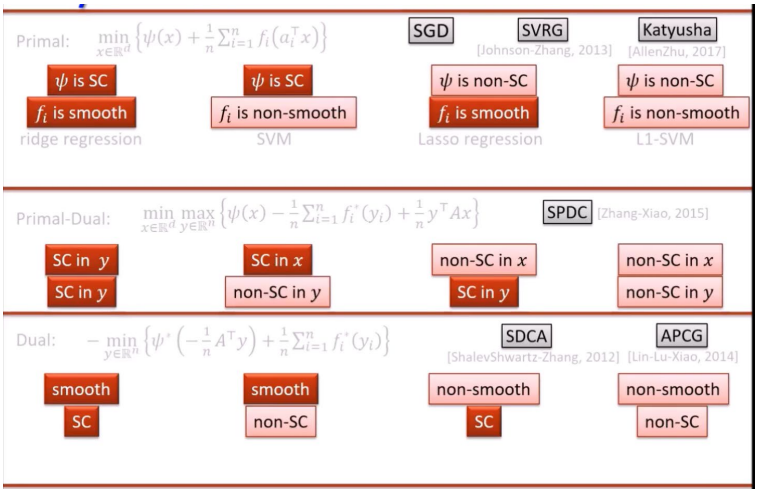
下面这四种 cases 都是针对原问题说的，他们的对偶问题根据上面两条共轭的性质可以得到。

Case1: $\varphi(x)$ 强凸，f（x）光滑,也就是 σ>0,L<+∞；

Case2: $\varphi(x)$ 强凸，f（x）不光滑。σ>0,L=+∞；

Case3: $\varphi(x)$ 非强凸，f（x）光滑。σ=0,L<+∞；

Case4: $\varphi(x)$ 非强凸，f（x）非光滑。σ=0,L=+∞.

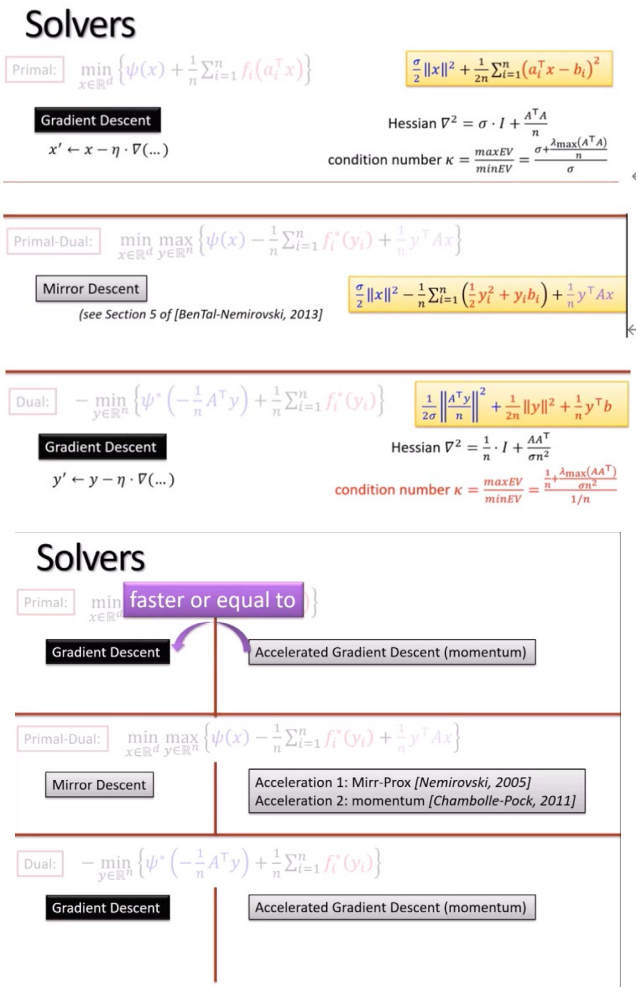


一些具体的例子：

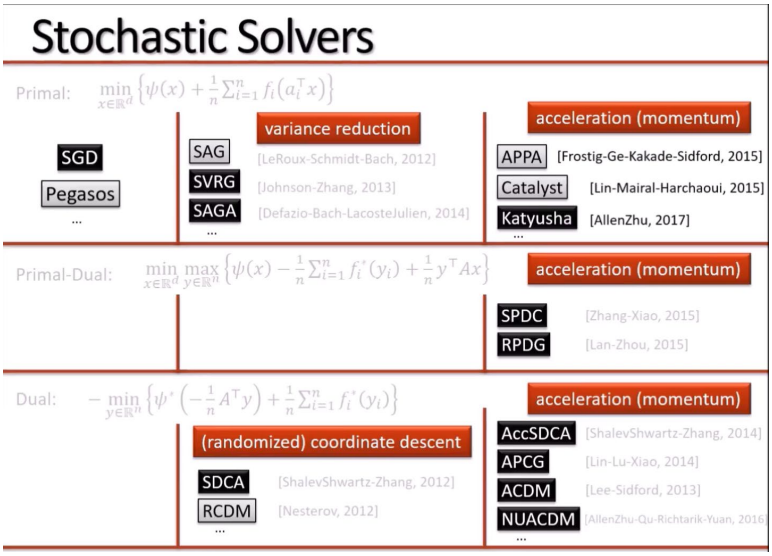
Problem	$\psi(x)$	$f_i(a_i^\top x)$	σ	L
ridge regression	$\frac{\sigma}{2}\|x\|^2$	$\frac{1}{2}(a_i^\top x-b_i)^2$	σ	$\|a_i\|^2$
Lasso regression	$\lambda\|x\|_1$	$\frac{1}{2}(a_i^\top x-b_i)^2$	0	$\|a_i\|^2$
logistic regression	$\lambda\|x\|_1$	$\log\left(1+e^{-b_i\cdot a_i^\top x}\right)$	0	$\frac{1}{4}\|a_i\|^2$
SVM	$\frac{\sigma}{2}\|x\|^2$	$\max\{0,1-b_i\cdot a_i^\top x\}$	σ	∞
L1-SVM	$\lambda\|x\|_1$	$\max\{0,1-b_i\cdot a_i^\top x\}$	0	∞

（3）solvers

basic solvers：gradient descent（针对 primal-dual 问题叫 mirror descent）



上面这张 slide 是 Gradient descent 的加速版本，也就是下面要引出的 stochastic solvers

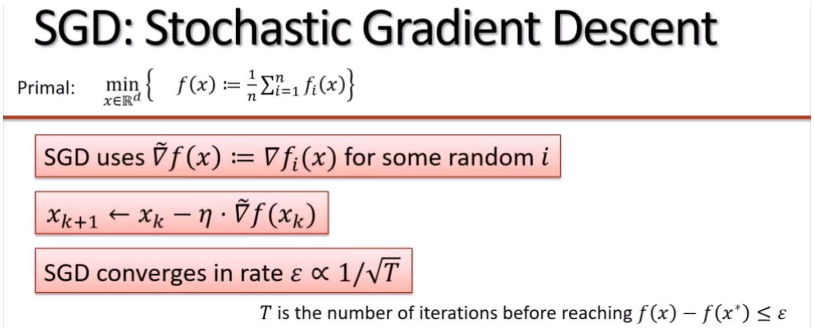


可以看到，原始、对偶、原始对偶问题，针对每一个问题都给出了三栏不同收敛速率的方法。每一行从左到右收敛速率都是越来越快的。

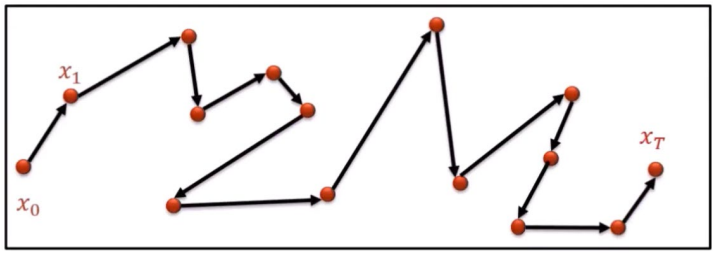
作者从中选择介绍了 SGD，SVRG，SDCA，APCG，SPDC，Katyusha 这六种方法。

Primal methods：	SGD，SVRG，Katyusha
Primal-dual methods：	SPDC
Dual methods:	SDCA，APCG

1） SGD



下面这个示意图说的就是 SGD 的过程，每次走的都不是真正的负梯度方向。



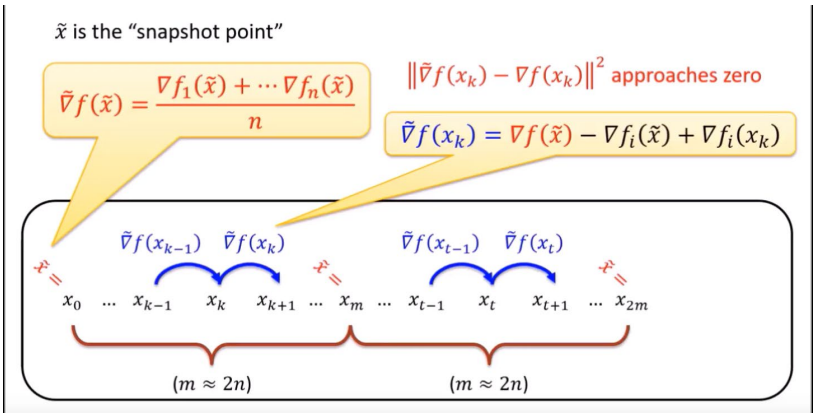
SGD 存在的问题：就是 SGD converges slowly because $\|\tilde{\nabla} f(x) - \nabla f(x)\|$ can be “large”

所以，自然引出 Can we design some other $\tilde{\nabla} f(x)$?

2） SVRG（Stochastic Variance Reduction Gradient）

SVRG converges in rate $\varepsilon \propto 1/T$

算法的思想：见下图可以看到 $\tilde{\nabla} f(x_k)$ is design as $\tilde{\nabla} f(x_k) = \nabla f(\tilde{x}) - \nabla f_i(\tilde{x}) + \nabla f_i(x_k)$



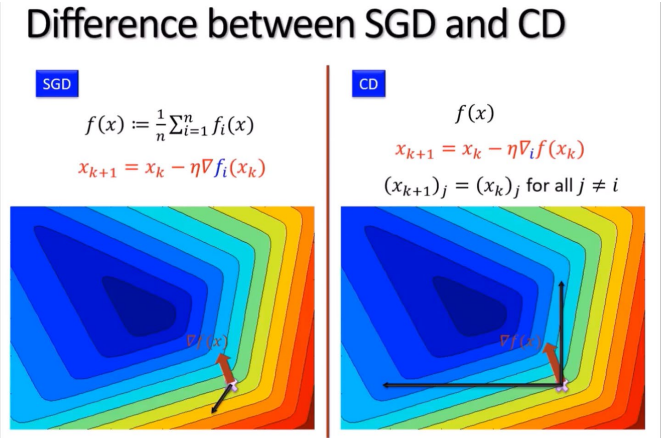
注：这里的每一个大括号 m 表示一个 epoch。

3） SDCA（Stochastic Dual Coordinate Ascent）

首先介绍了 SGD 和 CD（Coordinate Descent）的区别，

注意图中的红色公式，代表了它们的差别。

下面的彩图表示 SGD 可能走的是和负梯度完全相反的错误的方向，但是 CD 不会是错误的方向，会永远朝着和梯度大致一样的方向，沿着两个坐标轴交替走。

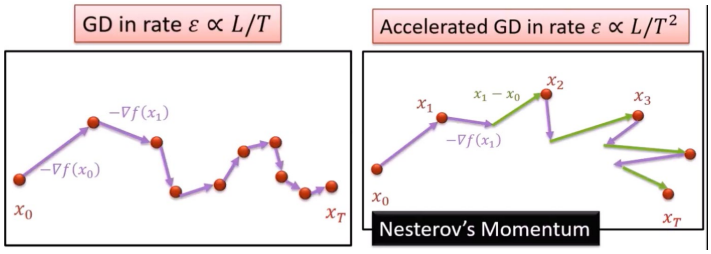


Remark：

1. SGD is harder to design than CD。
2. computation of $\nabla_i g(y)$ is of the same complexity as computation of $\nabla f_i(a_i^\top x)$

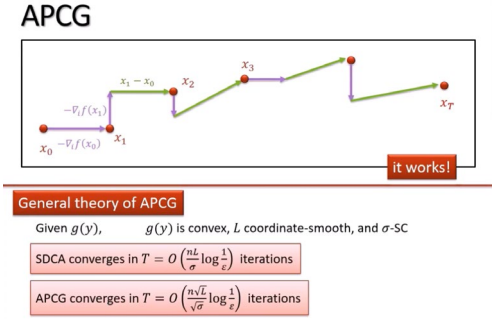
4) APCG（Accelerated Proximal Coordinate Gradient）

首先介绍了通过 Nesterov 's Momentum 实现 acceleration,实现加速的一般原理如下图。



从上图可以看出区别。看右图可以发现，x0 点和 GD 是一样的走法，但从 x1 点开始，x1 不仅走了该点的负梯度方向，还走了 x1-x0 这条绿线，由图即可看出实现了 accelerate.

APCG methods 就采用了 Nesterov 's Momentum 的思想，但是这个方法他每次走的不是 full gradient+Δx，而是 coordinate gradient+Δx.



APCG 的收敛速率是比 SDCA 还要快的。

5) SPDC（Stochastic Primal Dual Coordinate）

可以看到解决介绍的唯一一个解决 Primal-dual 问题的方法，之前的方法是：

因为 Primal-dual problem 中是同时优化原始变量 x 和对偶变量 y 的。

所以，之前的方法对 x 施加 full gradient descent，而对 y 施加 coordinate descent.

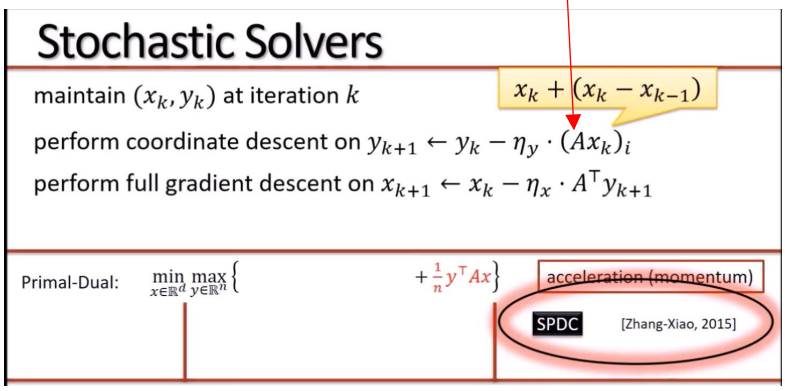
这里作者为了叙述简便，把原始对偶问题简化为下式，

Primal-Dual: $\min_{x\in\mathbb{R}^d}\max_{y\in\mathbb{R}^n}\left\{ \quad +\frac{1}{n}y^\top Ax\right\}$

所以实际上应用 gradient 就是对上面的式子求导，只不过，对 x 施加 full gradient descent，而对 y 施加 coordinate descent.

但是 SPDC 的贡献是应用了 Nesterov 's Momentum 实现了加速。

也就是把 y_{k+1} 更新中的 x_k 换成$x_k+(x_k-x_{k-1})$，这里也包括了 Δx.

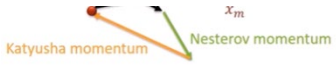


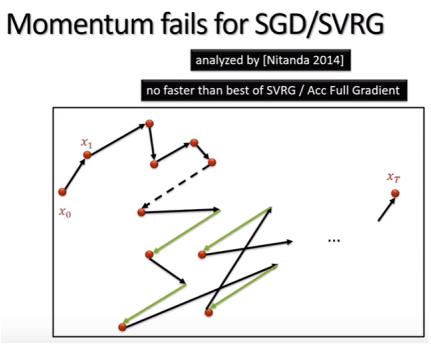
6）Katyusha

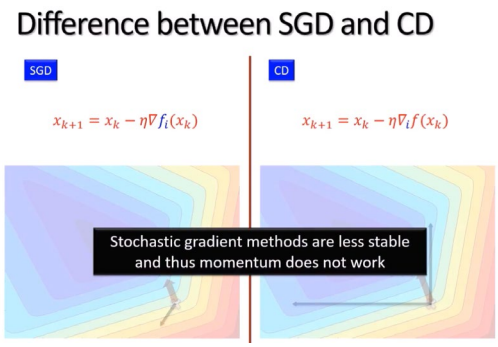
这个是作者朱泽园提出的方法，他的启发是，既然 Dual 问题的 APCG method 和 Primal-dual 问题的 SPDC method 都能应用 Momentum？那么针对 Primal 问题能不能也应用 Momentum 呢？

作者给出的答案是，可以。但是因为 Nesterov 's Momentum 不能应用到 SGD/SVRG 这两个 primal methods (因为如果 stochastic 走的不是正确的 full gradient 方向，那么如果再应用 Nesterov 's Momentum，每次再多走一个 Δx，那么这个 mistake 就会被 double.下面两个图解释了这个问题，并说明了为什么 SGD 不能用但是 CD 可以用 Momentum)，所以作者应用的除了 Nesterov 's Momentum，还有一个叫做 Katyusha Momentum

示意图：

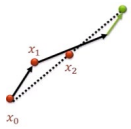




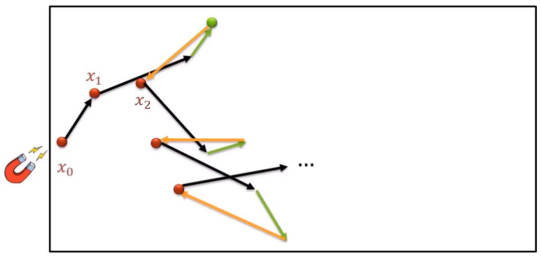


思想：

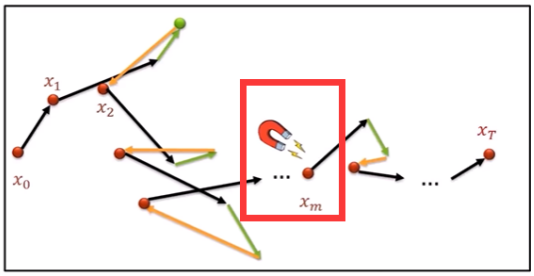
Step1：黄线是创新的地方，也就是每次走的时候，除了走梯度、Δx、再走一个

$\frac{1}{2}\Delta$（绿点-x_0）  （形象地说，就相当于 x0 点有一个磁铁在说 don‘t forget come back home）

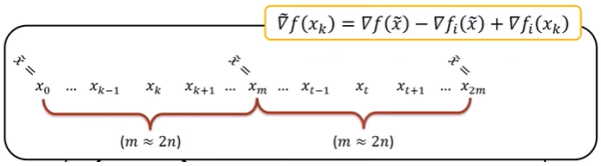
Katyusha Method



Step2：下一个 epoch，磁铁换了位置，成了 x_m 点

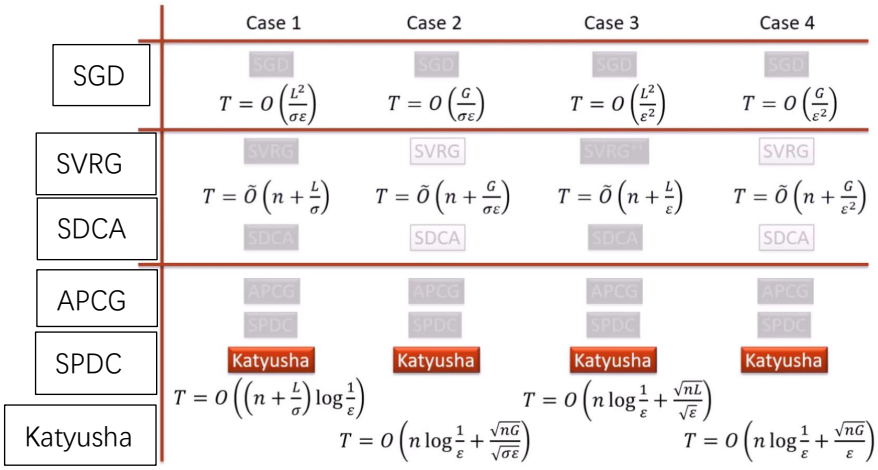


注：每一个 epoch



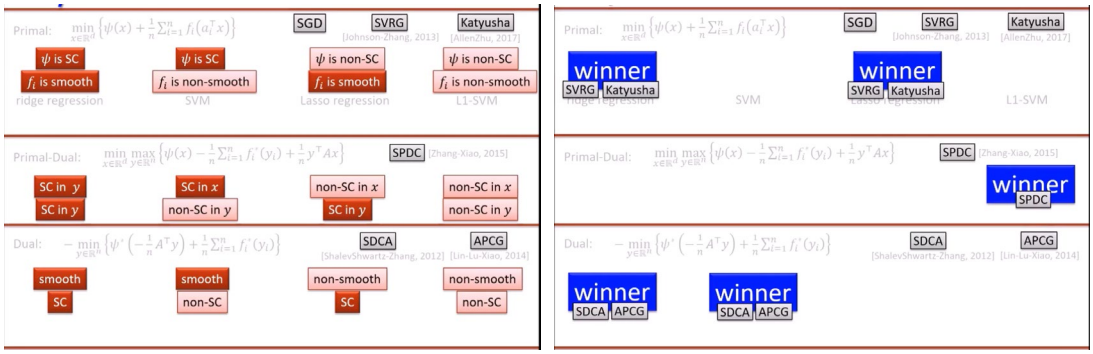
综合以上，

以上这些方法的收敛速率？

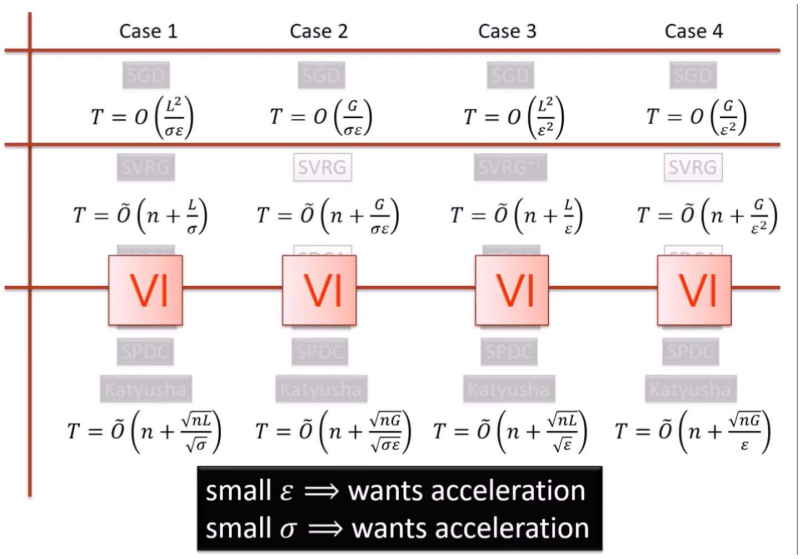


作者最后提出了两个问题？

问题 1：怎么样根据 smooth & strongly convex 去选择这些方法？



问题 2：什么时候需要加速？



At last, my opinion:

因为针对原始对偶问题 min max 形式，SPDC 这个方法一般可以用来解决非凸非光滑的 Primal-dual 形式的问题。

针对原始问题：$\min_{x\in\mathbb{R}^d}\left\{\psi(x) + \frac{1}{n}\sum_{i=1}^{n} f_i(a_i^\top x)\right\}$，其中 $\varphi(x)$ 为 regularizer，f(x)为 loss function（ERM）。在 case 1 和 case 2 的情况，作者推荐使用 primal 的 SVRG 或 Katyusha 方法，或者 dual 的

SDCA 和 APCG，我们如果想用对偶的方法的话，最好选择 Dual 形式的 SDCA 或者 APCG 及其变形形式。

Case1: $\varphi(x)$ 强凸，f（x）光滑,也就是 σ>0,L<+∞；

Case2: $\varphi(x)$ 强凸，f（x）不光滑。σ>0,L=+∞；