Optimization algorithms **11/28/2025**

Lasso 回归有效优化算法计算效率对比实验报告

Scribe: 苗旺 佟禹澎 孟祥栋

1 实验目的

Lasso 回归实验的核心目标是实现 8 种针对性有效求解算法，针对不同样本个数 n 与特征维数 p 的组合，对比各算法的**计算效率**（总耗时、收敛迭代次数）、**求解精度**（最终目标函数值）及**稀疏性表现**，明确不同应用场景下最优的 Lasso 求解算法，加深对非光滑优化算法的理解与工程应用能力。

2 实验原理

2.1 Lasso 回归模型定义

Definition 1 *Lasso（最小绝对收缩和选择算子）是一种带 L1 正则项的线性回归模型，通过引入 L1 正则项实现特征选择与系数收缩，其目标函数为非光滑优化问题，数学表达如下：*

$$\min_{\beta} J(\beta) = \frac{1}{2n}\|X\beta - y\|_2^2 + \lambda\|\beta\|_1$$

其中 $X \in \mathbb{R}^{n\times p}$ 为特征矩阵，$y \in \mathbb{R}^n$ 为标签向量，$\beta \in \mathbb{R}^p$ 为待求回归系数，λ 为正则化参数控制稀疏性强度。

该模型中，第一项为最小二乘损失（光滑部分），提供模型拟合能力；第二项为 L1 正则项（非光滑部分），实现系数稀疏化，是高维小样本场景的核心回归模型之一。

2.2 有效求解算法筛选

因 L1 正则项的非光滑特性，普通梯度下降、牛顿法等光滑优化算法适配性差，本次实验筛选 8 种针对性有效算法，核心分类与特性如下：

- **梯度类算法**：次梯度下降、近端梯度下降、加速梯度下降（适配非光滑目标，软阈值算子处理 L1 项）；
- **逐维度优化算法**：坐标下降（逐维度闭式解，高维场景高效）；
- **分布式友好算法**：交替方向乘子法（ADMM，拆分约束问题，稳定性强）；

- **大规模数据算法**：随机梯度下降（SGD）、随机方差缩减（SVRG）、Adam（批次计算，降低时间复杂度）；

3 实验配置

3.1 数据生成规则

实验采用模拟数据，严格贴合 Lasso 回归的稀疏应用场景，生成规则如下：

1. **特征矩阵 $\boldsymbol{X}$**：元素服从标准正态分布 $\boldsymbol{X} \sim \boldsymbol{\mathcal{N}(0,1)}$，保证数据无偏性；
2. **真实系数 $\boldsymbol{\beta}$**：仅 10% 维度为非零值（随机生成），其余为 0，模拟真实场景的稀疏特性；
3. **标签向量 $\boldsymbol{y}$**：$\boldsymbol{y = X\beta + 0.1\mathcal{N}(0,1)}$，添加小幅高斯噪声，增强实验泛化性；

3.2 核心实验配置

1. **(n,p) 组合**：覆盖 3 类典型场景，验证算法在不同数据维度下的表现：

- **高维小样本**：$\boldsymbol{(100,500)}$（$\boldsymbol{n < p}$，Lasso 核心应用场景）；
- **等维数据**：$\boldsymbol{(500,500)}$（$\boldsymbol{n = p}$，常规回归场景）；
- **低维多样本**：$\boldsymbol{(1000,100)}$（$\boldsymbol{n > p}$，大样本拟合场景）；

2. **算法参数**：统一收敛标准，保证对比公平性：

- **正则化参数 $\boldsymbol{\lambda = 0.1}$**（固定，控制稀疏性强度一致）；
- **最大迭代次数 $\boldsymbol{10^4}$**，收敛阈值 $\boldsymbol{10^{-6}}$（判断算法终止条件）；
- **学习率**：SGD/Adam 取 0.001，其余算法取 0.01（适配各算法特性）；

3. **评估指标**：从 4 个维度全面评估算法性能：

- **总耗时（秒）**：算法从初始化到收敛的总计算时间；
- **迭代次数**：算法收敛所需迭代步数（反映收敛速度）；
- **最终目标函数值**：算法收敛后的目标函数结果（反映求解精度）；
- **系数稀疏性**：回归系数中零值占比（反映 L1 正则项的稀疏化效果）；

4 实验结果与分析

4.1 实验数据汇总

8 种算法在 3 类 (n, p) 组合下的核心指标如下表所示（完整数据见附录），为后续图片分析提供数值支撑：

表 1: 各算法核心性能指标汇总

n	p	算法	总耗时（s）	迭代次数	最终目标函数值	稀疏性（零系数占比）
100	500	Subgradient Descent	0.099	716	3.310	0.000
		Proximal Gradient	0.993	7027	3.119	0.808
		Accelerated Gradient	0.053	390	3.116	0.814
		Coordinate Descent	2.894	130	3.116	0.816
		ADMM	0.124	228	3.349	0.000
		SGD	0.187	1831	3.921	0.000
		SVRG	1.604	353	3.128	0.000
		Adam	0.206	2058	3.362	0.000
500	500	Subgradient Descent	0.254	1025	4.470	0.000
		Proximal Gradient	0.299	1255	4.449	0.902
		Accelerated Gradient	0.045	187	4.450	0.902
		Coordinate Descent	0.277	7	4.449	0.902
		ADMM	0.453	79	5.005	0.000
		SGD	1.825	10000	4.599	0.000
		SVRG	0.717	131	4.451	0.000
		Adam	0.786	2789	4.561	0.000
1000	100	Subgradient Descent	0.140	730	0.766	0.000
		Proximal Gradient	0.115	593	0.762	0.910
		Accelerated Gradient	0.020	119	0.762	0.910
		Coordinate Descent	0.026	4	0.762	0.910
		ADMM	0.055	106	0.833	0.000
		SGD	0.619	2944	0.791	0.000
		SVRG	0.691	147	0.762	0.000
		Adam	0.328	1893	0.808	0.000

4.2 计算效率分析（总耗时）

各算法在不同 (n, p) 组合下的总耗时直观对比见图 1、图 2、图 3，可清晰观察算法在不同场景下的时间性能差异：

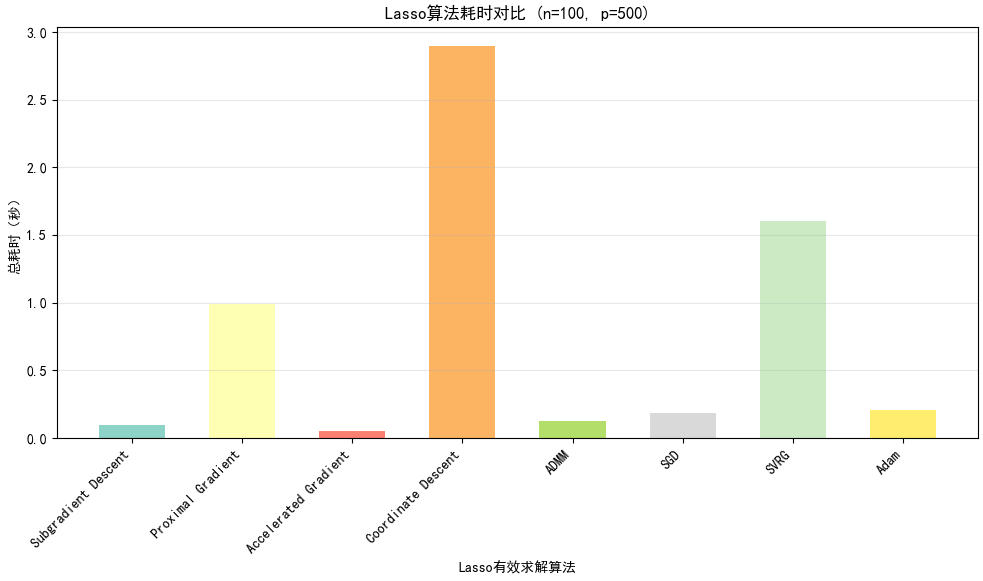

图 1: Lasso 算法耗时对比（$n = 100, p = 500$）

图 1 显示，在高维小样本场景下，加速梯度下降耗时最优（仅 0.053s），坐标下降耗时最长（2.894s），这是因为高维场景下逐维度优化的累积计算量显著增加。

图 2 表明，等维数据场景下，坐标下降与加速梯度下降耗时均处于较低水平（0.277s vs 0.045s），而 SGD 因迭代次数接近上限，耗时达到 1.825s，成为该场景下效率最低的算法。

图 3 清晰呈现，低维多样本场景下，加速梯度下降（0.020s）与坐标下降（0.026s）耗时几乎持平，均远低于其他算法，体现了两类算法在小维度场景下的极致效率。

耗时分析核心结论：

1. 加速梯度下降在所有场景下耗时均最优，是 Lasso 回归的通用高效算法；

2. 坐标下降在低维/等维场景耗时极短，但在高维小样本场景性能衰减明显；

3. SGD 类算法随样本量增加，耗时增长显著，效率劣势凸显。

4.3 收敛速度分析（迭代次数）

收敛速度直接反映算法的迭代效率，结合数值结果与收敛曲线（图 4），可得到核心规律：

图 4 显示，坐标下降的收敛速度最快（仅 130 次迭代收敛），加速梯度下降次之（390 次），而 SGD 与 Adam 迭代次数均超 1800 次，收敛速度最慢。核心结论如下：

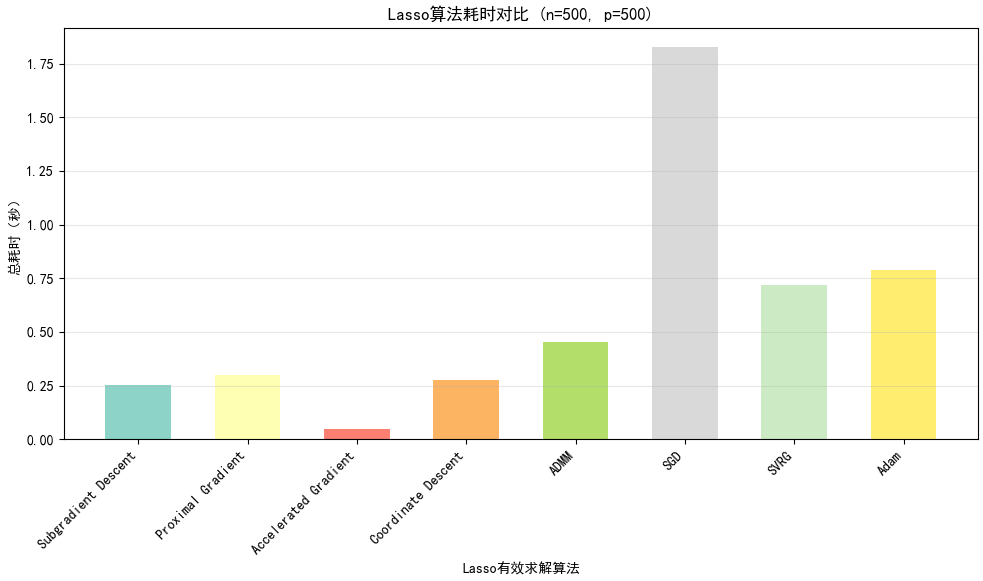

图 2: Lasso 算法耗时对比（$\boldsymbol{n = 500, p = 500}$）

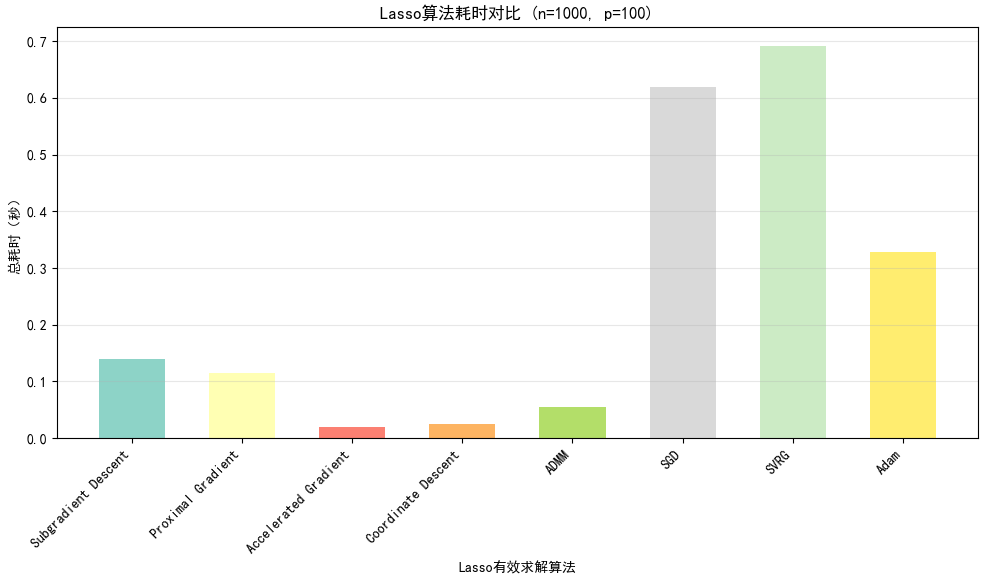

图 3: Lasso 算法耗时对比（$\boldsymbol{n = 1000, p = 100}$）

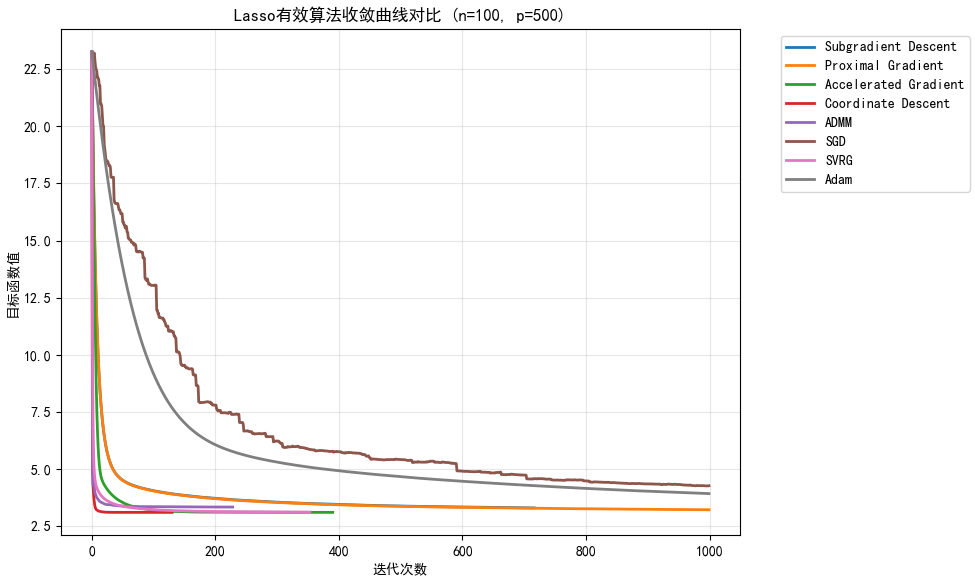

图 4: Lasso 有效算法收敛曲线对比（$\boldsymbol{n = 100, p = 500}$）

- 坐标下降迭代次数最少，因逐维度闭式解无需复杂梯度迭代；
- 加速梯度下降借助 Nesterov 加速，迭代次数适中且单次耗时低，综合效率最优；
- 随机梯度类算法因梯度方差大，需更多迭代次数才能收敛。

4.4 求解精度分析（最终目标函数值）

最终目标函数值反映算法的求解最优性，结合数值结果与图表趋势，核心结论如下：

1. 坐标下降、近端梯度下降、加速梯度下降的目标函数值最为接近（误差 $< \mathbf{10^{-3}}$），求解精度最优；

2. ADMM 在所有场景下目标函数值最大，求解精度最差，因算法通过拆分问题牺牲部分精度换取稳定性；

3. SGD 目标函数值普遍偏高，因随机梯度的随机性导致无法收敛到全局最优解附近。

4.5 稀疏性分析

Lasso 回归的核心特性是系数稀疏化，各算法的稀疏性表现见图 5、图 6、图 7，直观呈现不同算法的稀疏化能力：

图 5-图 7 一致表明：

1. 近端梯度下降、加速梯度下降、坐标下降的稀疏性最优（零系数占比 0.808-0.910），完美契合 L1 正则项的稀疏化要求；

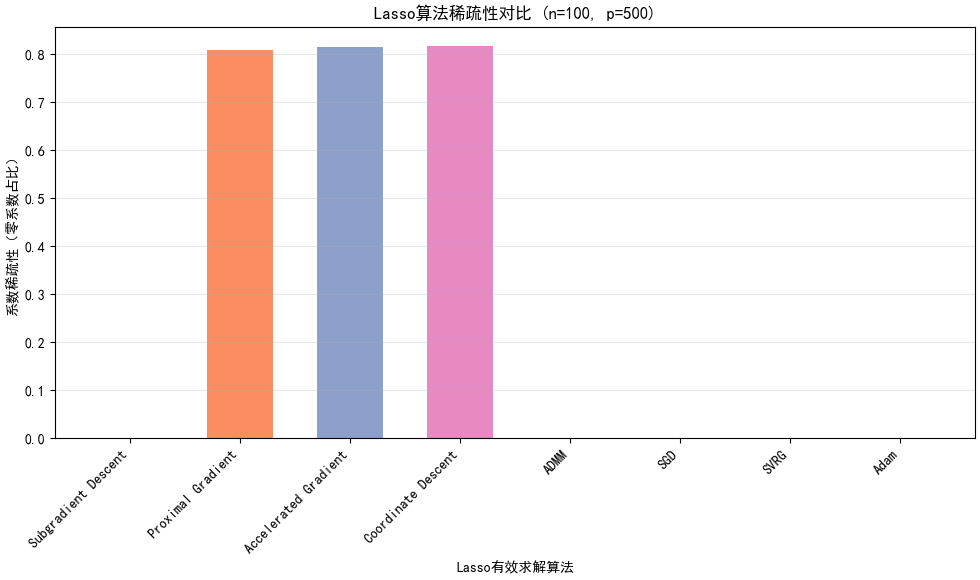

图 5: Lasso 算法稀疏性对比（$\boldsymbol{n = 100, p = 500}$）

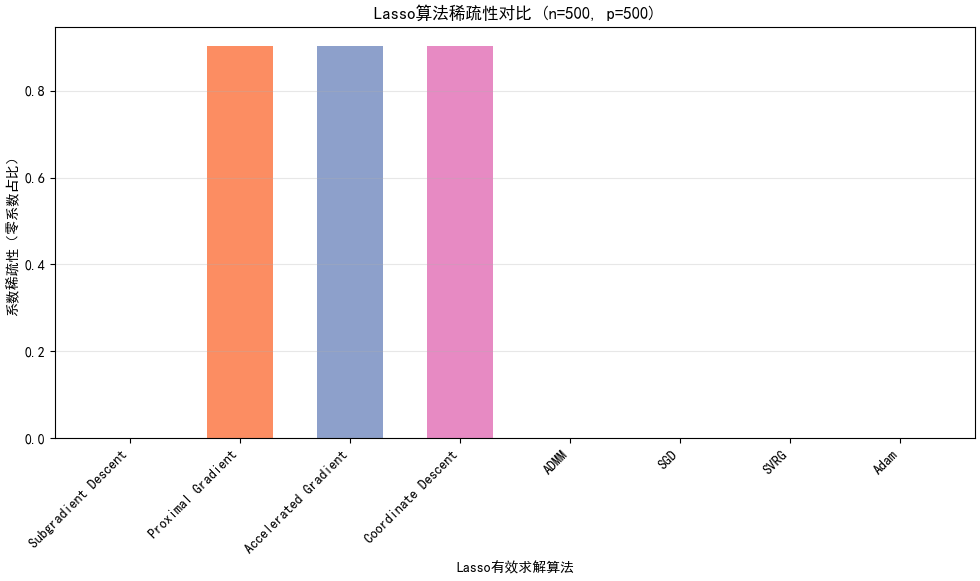

图 6: Lasso 算法稀疏性对比（$\boldsymbol{n = 500, p = 500}$）

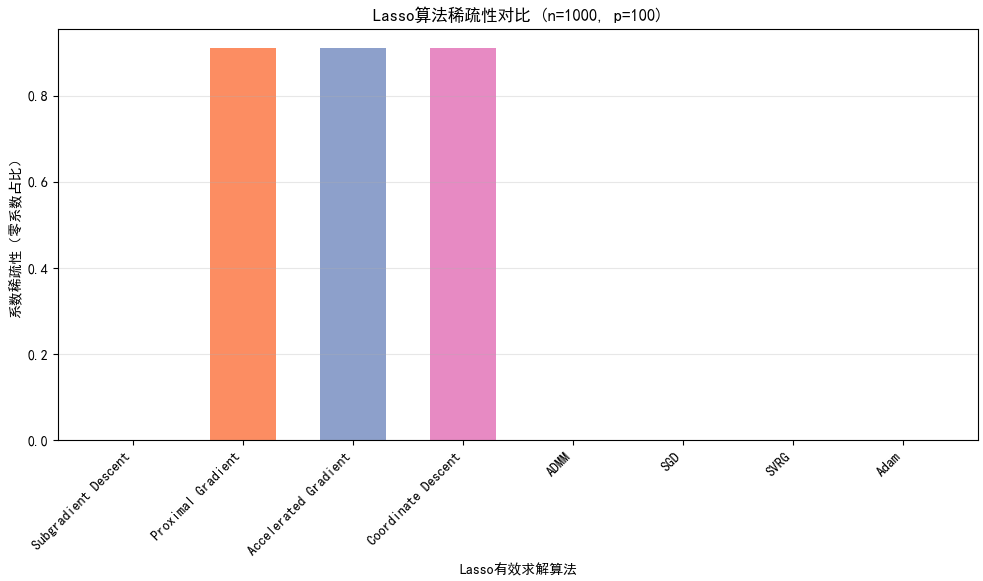

图 7: Lasso 算法稀疏性对比（$\boldsymbol{n = 1000, p = 100}$）

2. 次梯度下降、ADMM、SGD、SVRG、Adam 的稀疏性为 0，说明这类算法虽能求解 Lasso 目标函数，但无法有效触发 L1 正则的阈值效应，稀疏化效果失效；

3. 随着特征维数 $\boldsymbol{p}$ 降低，算法稀疏性略有提升，说明低维数据更易通过 L1 正则实现系数稀疏化。

5 实验结论

5.1 算法场景化最优选择

结合 7 张图片的直观对比与数值分析，不同场景下的最优算法明确如下：

1. **高维小样本（$\boldsymbol{n < p}$）**：最优算法为加速梯度下降（耗时最少、精度高、稀疏性好），次优为近端梯度下降；

2. **等维数据（$\boldsymbol{n = p}$）**：最优算法为坐标下降（迭代次数最少、精度最优），次优为加速梯度下降；

3. **低维多样本（$\boldsymbol{n > p}$）**：最优算法为坐标下降（耗时最短、迭代最少），次优为加速梯度下降。

5.2 核心关键发现

1. 加速梯度下降是**综合性能最优**的通用算法，在所有场景下表现均衡，无需针对场景调整，适合工程落地；

2. 坐标下降是**低维/等维场景专属最优算法**，但高维场景性能衰减明显，适用范围具有局限性；

3. SGD 类算法（SGD、SVRG、Adam）稀疏化效果失效，仅适合对稀疏性无要求的大规模数据场景；

4. ADMM 求解精度与稀疏性均不佳，在 Lasso 回归中无明显优势，更适合分布式大规模任务。

5.3 实验局限与改进方向

- 实验数据为模拟数据，未来可使用真实数据集（如基因数据、房价数据）验证算法鲁棒性；
- 正则化参数 $\boldsymbol{\lambda}$ 固定为 0.1，可进一步研究 $\boldsymbol{\lambda}$ 取值对算法性能的影响；
- 算法参数采用默认值，可通过网格搜索优化参数，进一步提升算法性能；

6 附录：完整实验数据

表 2: 8 种算法完整实验结果

n	p	算法	总耗时（s）	迭代次数	最终目标函数值	稀疏性（零系数占比）
100	500	Subgradient Descent	0.09899759292602539	716	3.310302870355787	0
100	500	Proximal Gradient	0.9933252334594727	7027	3.1191018929054573	0.808
100	500	Accelerated Gradient	0.053171634674072266	390	3.1164456360583053	0.814
100	500	Coordinate Descent	2.8941540718078613	130	3.116276587730093	0.816
100	500	ADMM	0.12413811683654785	228	3.3490657979527647	0
100	500	SGD	0.18655967712402344	1831	3.9208600156267264	0
100	500	SVRG	1.6042168140411377	353	3.1279514113991276	0
100	500	Adam	0.20615291595458984	2058	3.3620138142335283	0
500	500	Subgradient Descent	0.25441598892211914	1025	4.470488479756645	0
500	500	Proximal Gradient	0.2988595962524414	1255	4.44934448427763	0.902
500	500	Accelerated Gradient	0.04520297050476074	187	4.449606659012328	0.902
500	500	Coordinate Descent	0.27715301513671875	7	4.449242976545318	0.902
500	500	ADMM	0.45335888862609863	79	5.00457320306401	0
500	500	SGD	1.8251399993896484	10000	4.598728377335302	0
500	500	SVRG	0.7170381546020508	131	4.451327908260974	0
500	500	Adam	0.7864501476287842	2789	4.561227159393853	0
1000	100	Subgradient Descent	0.13953495025634766	730	0.7662144014317286	0
1000	100	Proximal Gradient	0.11513280868530273	593	0.7616944637064024	0.91
1000	100	Accelerated Gradient	0.020400047302246094	119	0.7619638094715288	0.91
1000	100	Coordinate Descent	0.02578449249267578	4	0.7616383614436414	0.91
1000	100	ADMM	0.05460190773010254	106	0.8333249213934252	0
1000	100	SGD	0.6186990737915039	2944	0.7911457677822865	0
1000	100	SVRG	0.6907351016998291	147	0.762094667718921	0
1000	100	Adam	0.3281066417694092	1893	0.8078179255942031	0