SC3复现实验报告总结

> 任务1: 自行实现SC3算法(不要求SVM优化)，并在论文中给出的12个数据集上进行测试。
> 任务2: 在R语言环境下安装SC3的软件包,将问题1中的效果和该软件本身的效果进行比较。

- 已实现python版本SC3 basic算法，该方法现已开源在GITHUB。算法实现除Kmeans算法外，其他部分已验证完全与SC3 paper和open R package source code一致对齐。
  - Kmeans算法不一致原因为：R package调用的Kmeans算法基于Hartigan-Wong algorithm；python使用Elkan algorithm，两者区别在于Hartigan-Wong algorithm是先随机分配点到cluster，再进行抛出，更新，分配；而Elkan algorithm基于kmeans++做了速度优化，也就是初始化cluster时先基于距离确定不同cluster。
- 已在论文中12个数据集的5个数据上跑通并验证ARI指标结果，分别是 `Yan, Biase, Goolam, Deng, Klein` 数据。
  - Patel数据仅有logcount数据，无法在R package中跑通，因此未实验。其他数据（或cell_type）无法从hemberg-lab.github.io数据网站下载，因此未做实验验证。
  - 下表为R的SC3 package与实现的python版本实现比对定量结果；其中，括号内为评估的k值。

ARI	Yan	Biase	Goolam	Deng1	Deng2	Klein
Rcode	0.6584(6)	0.9871(5)	0.5973(6)	0.4439(9)	0.7876(9)	0.6178(12)
Python	0.6584(6)	0.9871(5)	0.5973(6)	0.3827(9)	0.6648(9)	0.7291(12)

- 说明：算法实现为SC3类，可通过以下方式调用。其中 `data_root, anno_root` 为数据csv文件和 `cell type` txt文件； `k_range` 参数默认 `None` 值，由算法评估计算，也可自行指定； `num_SVM_train` 值默认为 `0` 表示不使用SVM-mixed模型，仅基于聚类；非零值表示用于unsupervised cluster的数量，并进行SVM train，数据剩余数量被用来SVM test。
  - 具体实现及调用方式可参考源代码

```
sc3 = SC3(data_root, anno_root, k_range=None, num_SVM_train=0)
sc3.sc3_onego_run(writeroot=WRITEROOT)
```

> 任务3: 对于你实现的SC3算法进行SVM优化，并在论文中提到的大规模数据集或者在其它大规模数据集(大于5000个，比如上述地址中的Baron)中进行验证，并和SC3使用SVM优化的效果进行比较。

- 已实现SVM优化，直接通过指定 `num_SVM_train` 值即可。与R保持一致，超过5000 cell的数据默认会随机选择5000个cell进行**聚类、分配伪标签并进行SVM train**，剩余样本进行SVM test后一并合并标签输出“聚类”结果。
- 该SVM优化版本SC3算法在Baron human数据上验证，该数据维度 `8569*20125`

- R code ARI: **0.2977** [est_k: 37, times: ~2h]
- Python ARI: **0.3276** [est_k: 37, times: ~57min]

> 任务4: 注意该论文中的表述存在一些错误，其中提到对于距离矩阵使用PCA，然而其实际实现时并为将每个样本对应的向量投影到PCA的前k个主成成分所对应的向量上，而是直接使用前k个主成成分作为PCA的结果。这有一点像PCoA(主坐标分析),然而对于PCoA来说，结果并未在每个维度乘上对应的特征值。或许这种处理也和KPCA(核技巧在PCA上的应用)有所联系。写一写你对上述题干的理解，并以理论或实际有进一步表现为目标尝试给出一个改进。更进一步，你是否能尝试改变SC3的各个距离度量模块以及降维模块，以获得更好的聚类效果。

- **比较于PCoA** SC3先对输入矩阵X_{m*n}进行距离度量获得dissimilaity matrix M_{n*n}；然后计算协方差矩阵T_{n*n}，等价于PCoA中获得离差矩阵；然后对T_{n*n}进行特征分解获取前k个特征值对应特征向量，得到矩阵$C = U^T$ (假设U_{d*d}每一列为特征值对应的特征向量)；而PCoA分析中$C = \Lambda^{\frac{1}{2}} \cdot U^T$才为坐标矩阵；因此，这里没有乘各自特征值的平方根。
  此外，有一点值得注意，原SC3实现中dissimilaity matrix送入PCA之前经过了一次数据标准化，导致原本dissimilaity matrix的数据对称性被破坏。也就不满足PCoA的输入形式。
  因此，从这两个角度理解下来，**SC3做法不等价于PCoA。**
- **比较于KPCA** SC3先对输入矩阵X_{m*n}进行距离度量获得dissimilaity matrix M_{n*n}；然后对M_{n*n}标准化后得到$\bar{M}_{n*n}$，我们假设由X到$\bar{M}$映射函数为$\phi(x)$；然后SC3算法对$\bar{M}$进行PCA分析并获取其前k个主成分，
  也就是，获取了$\bar{M}_{n*n}$的协方差矩阵T_{n*n}的前k个特征值对应的特征向量矩阵；
  我们形式化为：
  $T_{n*n} = \frac{1}{n} \bar{M}_{n*n} \cdot \bar{M}^T_{n*n}$ **[6]**，
  将$n \cdot T_{n*n} = \bar{M}_{n*n} \cdot \bar{M}^T_{n*n}$可等价于KPCA中的核矩阵$K_{n*n}$，
  则，以上做法形式上等价于对X进行KPCA。
  另外，$n \cdot T_{n*n}$核矩阵为协方差矩阵，因此一定为半正定矩阵，满足核函数mercer定理；因此**核函数形式上等价。**
  但这里$\bar{M}_{n*n}$已知，由X经$\phi(x)$映射，并非向高维空间映射，因此，我认为并**不严格意义等价。**
- 上述表述的分析推导过程
- 因此，我思考以下问题：
  1. SC3做法既然等价于对X进行KPCA，且高维空间映射不满足；那能否先把dist matrix当作kernel matrix，进而直接做特征分解？此时相当于dist matrix不再做协方差矩阵。
  2. 或者，能否修改dist matrix，即，改变为其他的kernel function（RBFkernel等）得到不同的kernel matrix?
  3. 既然等价于KPCA，映射到高维空间实现非线性；而且Laplacian矩阵特征分解其实是spectral cluster，也是非线性方法；那是否可以采用其他非线性映射方法，如LLE或tSNE或UMAP?
- 上述问题经过实验验证，有些并不可行（实验结果见下表）。
  - 直接对dist matrix进行特征分解获取特征向量并不理想，考虑是否是计算的欧式距离矩阵和pearson矩阵不满足mercer定理？这个尚未推导；

    - 改变kernel function，采用RBF kernel和linear kernel获取dist matrix，获得了不错的结果，目前RBFkernel在多个数据上结果都优于先前。
    - 采用UMAP算法对dissimilarity matrix进行降维，结果也不理想；考虑是不是优化未收敛的原因，或者是超参数有调整的必要？尚未做更多验证。

ARI (est_k)	Yan	Biase	Goolam	Deng1	Deng2	Klein	Baron
Rcode	0.6584(6)	0.9871(5)	0.5973(6)	0.4439(9)	0.7876(9)	0.6178(12)	0.2977(37)
Python(base)	0.6584(6)	0.9871(5)	0.5973(6)	0.3827(9)	0.6648(9)	0.7291(12)	0.3276(37)
-laplacian	0.7212(6)	0.9839(5)	0.5973(6)	0.4333(9)	0.7745(9)	0.8224(12)	0.4179(37)
-laplacian +UMAP*	0.7902(6)	0.8302(5)	0.5927(6)	0.3606(9)	0.6616(9)	0.5548(12)	-
-laplacian -spearman	0.7212(6)	0.9839(5)	0.5973(6)	0.4440(9)	0.7888(9)	0.8247(12)	0.3757(37)
-laplacian -spearman +rbf	0.7212(6)	0.9871(5)	0.5927(6)	0.4721(9)	0.8366(9)	0.8167(12)	0.4228(37)
-laplacian -spearman +linear	0.7212(6)	0.9807(5)	0.5973(6)	0.4343(9)	0.7722(9)	0.8372(12)	0.4208(37)
-laplacian -spearman +rbf pca>decom (similarity)	-	-	-	-	0.6870(9)	0.3402(12)	-

- 对于SC3算法也有一些其他实验，在以下实验结果中体现并验证。
    - 欧式距离归一化以及从信息角度度量样本进而采用JS散度距离。

ARI (est_k)	Yan	Biase	Goolam	Deng1	Deng2	Klein	Baron
Python(base)	0.6584(6)	0.9871(5)	0.5973(6)	0.3827(9)	0.6648(9)	0.7291(12)	0.3276(37)
normdists	0.6584(6)	0.9871(5)	0.5973(6)	0.3647(9)	0.6557(9)	0.6870(12)	-
-laplacian +normdists	0.6584(6)	0.9394(5)	0.5860(6)	0.4504(9)	0.7975(9)	0.6141(12)	0.2983(37)
-laplacian +JS dist	0.7080(6)	0.9839(5)	0.5973(6)	0.3023(9)	0.5858(9)	0.8070(12)	-