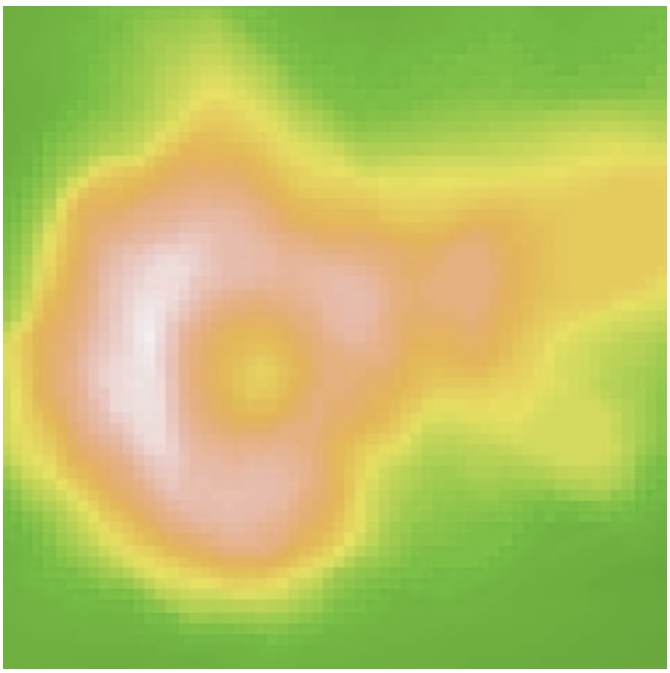
仿真实验报告

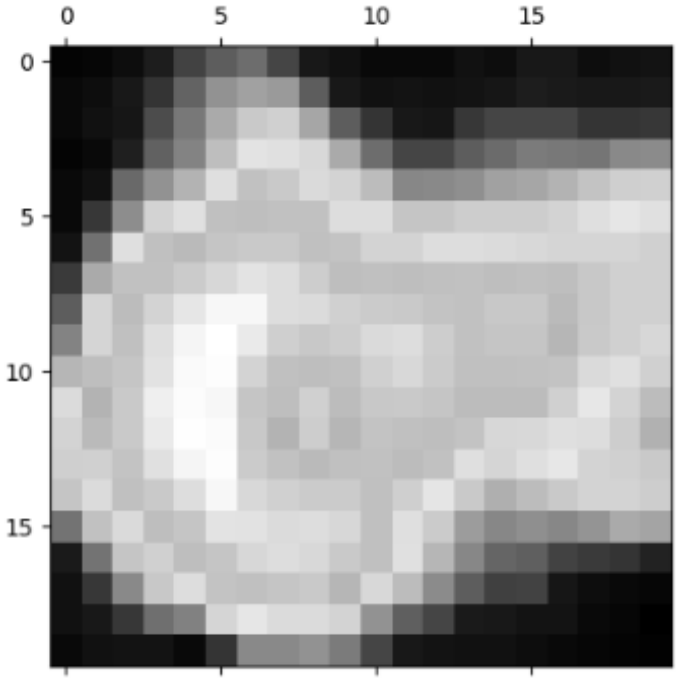
Jinwei Han, jinwei.han93@gmail.com, han550@purdue.edu

1. 模拟空间数据的 $\vec{\mu}$ 和 V

使用一张图片（代表我们希望去模拟的空间数据）生成我们想要的 $\vec{\mu}$ 和 V.

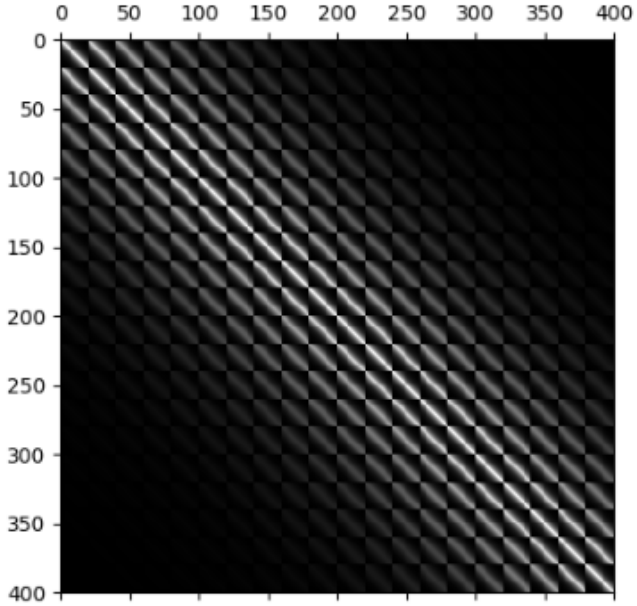


首先，为了得到 $\vec{\mu}$，我们需要做灰度化和采样(M=20)处理，(实际是20维的向量，下图是平面表示，符合直观)。



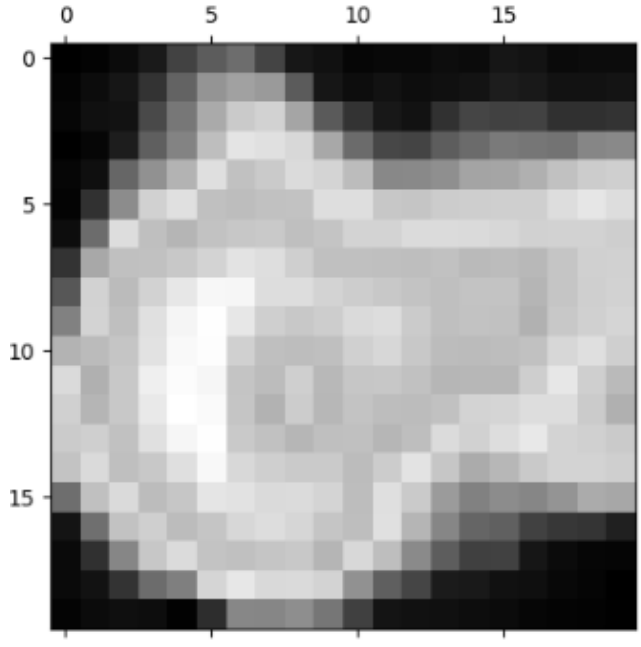
上图中，每个格子的灰度是 $\mu(s_i)$， $\vec{\mu}$ 的维度等于M*M=N=400， $\vec{\mu} \in R^N$。

然后，我们会基于空间关系生成V, $V = (\sigma^2 \exp \frac{-||s_i - s_j||_2}{\phi})$，$V$矩阵的宽高是$N * N = 400 * 400$。这里我们将空间控制在0-1的grid上（不要被图片的刻度迷惑，那只是直接print的矩阵，坐标是index）。



最后，我们有了$\vec{\mu}$和V，就可以生成模拟数据：

$$Y \sim N(\vec{\mu}, V)$$



上图就是我们重新按照模型生成的仿真数据，和平面化的$\vec{\mu}$非常相似，但由于随机生成，还是有略微的差异。

2. 模型参数

现在我们有了基于模型仿真出来的数据。

我们会将数据点分成两部分，一部分用做训练集，一部分用作测试集。在整体上进行k折验证，得到最终的模型fitness（使用一个类似R^2的概念，去衡量模型的预测能力）。

我们会分别将两个模型（传统的和近似的）分别去施加在上述数据集上。

涉及到的模型参数：

参数	取值	参数意义	备注
M	20	0-1方格横竖分割出的数量	规模可调
N	M*M	总的小格子数量，400	
phi	0.16	covariance function $\text{cov}(s_i, s_j) = e^{-\frac{d_{ij}}{\phi}}$	可调整，似乎有比0.16更好的值
sigma2	1	covariance function中使用	
k_fold_splits	10	将整体数据分成10份去进行k-fold验证，具体是用10%的数据去预测90%的剩余数据，还是用90%的数据去预测10%的剩余数据取决于k_fold_train_by_small参数	
k_fold_train_by_small	TRUE	用10%的数据去预测90%的剩余数据	
neighbor_relative_ratio	0.2	生成近似V^-1y矩阵时，用测试集中最近（欧式距离）的20%的测试集	可调整

关于K-fold验证，在这里我用了用10%的数据去预测90%的剩余数据（如果反过来，我们的近似算法性能相比会很差）

3. 算法分析（每fold）

定义1/k_fold_splits （训练集的占比）为 $\alpha = 0.1$
定义neighbor_relative_ratio（参与到近似计算的点占训练集的比例）为 $\beta = 0.2$

全体数据规模：N
预测集规模：αN

	传统算法（1）	近似算法（2）
V矩阵规模	$(\alpha N)^2$	$(\alpha\beta N)^2$
V矩阵取逆次数	1	αN

所以，我们的算法性能是用提升V矩阵求逆次数的代价换来的V矩阵的规模减少。
如果α接近1的话，我们的矩阵规模的降低带来的优势，会引入N倍的取逆次数的增加，所以不划算。
（比如我们用90%的数据去预测10%的数据，$\alpha = 0.9$，M=20时，这时耗时是12s跟133s。此时近似算法不占优势）

4. 实验

基于上述实验数据和参数，我们对不同M进行测试，对预测能力和性能做评估。

4.1 实验1 （参数设定M=20）

- 模型1: 使用全体预测集中的点

  R^2 =0.6650

  V矩阵的Conditional number=23.1（40维）

- 模型2: 使用近邻的点去近似（近邻策略是固定比例的近邻预测集）

  R^2 = 0.6653

  40次V矩阵的计算，平均Conditional number=12.2（8维）

在这组参数设定下，预测能力差不太多。由于低维数据，运算时长基本一致（都小于1s）。我们用更高维数据来验证性能问题。

4.2 实验2 （参数设定M=60）

- 模型1: 使用全体预测集中的点

  R^2 = 0.9681

  V矩阵的Conditional number大约700（360维）

  耗时：252s

- 模型2: 使用近邻的点去近似（近邻策略是固定比例的近邻预测集）

  R^2 = 0.9678

  360次V矩阵的计算，平均Conditional number大约350（72维）

  耗时：198s

R^2反映的预测能力差不多，但是新算法耗时确实是有一定幅度的减少（如果减少beta，减少alpha，这个耗时还能大幅度降低）

5. 代码仓库

https://github.com/jeyhan/spatial_stat.git