

笔记 III: TeraStructure 包

Jonathan Chow

2022-09-30

目录

1	TeraStructure 包	1
2	snpsamplinge.cc	2
2.1	算法思路	2
2.2	准备工作	3
2.3	求解子问题	3
2.4	更新全局变量	4
2.5	计算停止条件	5

1 TeraStructure 包

TeraStructure 包通过 C++ 实现使用随机变分推断算法 (SVI) 拟合 PSD 模型。

主要文件如下。之后我们只具体分析 `snpsamplinge.cc`。

模型拟合与参数更新: `snpsamplinge.hh`、`snpsamplinge.cc`。

环境设置: `env.hh`。

利用动态数组自定义的矩阵操作: `matrix.hh`。

读取 PLINK 数据: `snp.cc`。

多线程: `thread.cc`。

日志: `log.cc`。

用命令行控制训练的主函数: `main.cc`。

2 snpsamplinge.cc

2.1 算法思路

我们使用 C++ 的科学计算库 GSL。三个常用的 C++ 矩阵库 Eigen, Armadillo, GSL 在 Rcpp 中都有对应的版本 RcppEigen, RcppArmadillo, RcppGSL。我们回顾算法思路。

全局参数: 个体的种群分布参数 θ , θ 变分族 (Dirichlet 分布) 的变分参数 γ , 对应隐变量变分族的变分参数 α 。

局部参数: 种群的基因分布参数 β , β 变分族 (Beta 分布) 的变分参数 λ , 对应隐变量变分族的变分参数 η 。

(1) 初始化。抽样得到验证集, 初始化全局参数。

(2) 迭代。分为三步进行。

(2.1) 初始化。抽样得到子问题, 初始化局部参数。

(2.2) 迭代更新局部参数。为了方便起见固定迭代次数为 100。

(2.3) 更新全局参数。选取随机梯度方向。

(3) 停止条件。每隔适当的迭代次数计算一次。分为两步进行。

(3.1) 迭代计算验证集上的局部参数。为了方便起见固定迭代次数为 2000。

(3.2) 计算验证集上的对数似然函数。直到变化足够小或者连续下降多次。

2.2 准备工作

我们对子问题和验证集分别抽样。再次强调子问题是用来更新局部参数的，验证集是用来计算停止条件的。

```
void
SNPSamplingE::set_test_sample()

void
SNPSamplingE::set_validation_sample()
```

初始化 gamma 和 lambda。

```
void
SNPSamplingE::init_gamma()

void
SNPSamplingE::init_lambda()
```

初始化多线程。

```
int
SNPSamplingE::start_threads()
```

2.3 求解子问题

利用 eta，更新 lambda 和 beta。

```
void
SNPSamplingE::update_lambda(uint32_t loc)

void
SNPSamplingE::estimate_beta(uint32_t loc)
```

```
void
SNPSamplingE::optimize_lambda(uint32_t loc)

void
SNPSamplingE::compute_and_save_beta()
```

2.4 更新全局变量

计算辅助变量 phimom 和 phidad，它们是 alpha 和 eta 的结合，然后可以直接用于变分参数的更新。定义这两个变量是为了忠实地还原算法从而保证代码可读性。

```
inline void
SNPSamplingE::update_phimom(uint32_t n, uint32_t loc)

inline void
SNPSamplingE::update_phidad(uint32_t n, uint32_t loc)
```

设置随机梯度方向 rho，它是关于给定参数 tao 和 kappa 还有迭代次数的函数。

```
void
SNPSamplingE::update_rho_indiv(uint32_t n)
```

利用 phimom 和 phidad 还有 rho，更新全局参数。

```
void
PhiRunnerE::estimate_theta(const IndivsList &indivs)

void
PhiRunnerE::update_lambda_t(const IndivsList &indivs)
```

2.5 计算停止条件

先迭代 2000 次以计算验证集上的局部参数。然后计算验证集上的对数似然函数，判断是否停止。

```
double  
SNPSamplingE::compute_likelihood(bool first, bool validation)
```