

Contents

Contents	1
1 Base	3
1.1 Base.h	3
1.2 random.h	3
1.3 stl_util.h	3
1.4 probability.h	3
1.5 io_util.h	3
1.6 math_util.h	4
1.7 stat.h	4
2 ML	5
2.1 document.h	5
2.2 rbm	5
2.2.1 ais.h	5
3 Sparse Matrix	7
References	9

Chapter 1

Base

1.1 Base.h

1. void Sum(const VVReal &, VReal*);

1.2 random.h

1. void UniformSample(int t, VInt* v);
sample between 0 and size(v) uniformly t times.

1.3 stl_util.h

1. void Multiply(const VReal &src, double m, VReal* des);
2. void Multiply(const VVReal &src, double m, VVReal* des);
3. int DiffNum(const VInt &lhs, const VInt &rhs);
the number of different value between lhs and rhs
4. double Max(const T &data);
5. void Push(int num, const E &e, C* des);

1.4 probability.h

1. int SumTopN(const VInt &src, int len);
2. bool NextMultiSeq(int num, VInt* des);
产生下一个多项式分布的序列
3. bool NextBinarySeq(VInt* des);
产生下一个二进制的序列

1.5 io_util.h

1. void ReadFileToStr(const Str &file, Str* str);
2. void ReadFileToStr(const Str &file, const Str &del, VStr* data);
3. Str ReadFileToStr(const Str &file);

4. void WriteStrToFile(const Str &str, const Str &file);
5. void ReadFile(const Str &file, VInt* des);
6. void WriteFile(const Str &file, const VInt &data);

1.6 math_util.h

1. int Factorial(int n);

1.7 stat.h

1. double LogSum(double log_a, double log_b);
2. double LogPartition(const VReal &data);
输入data表示负能量值，返回对数配分函数。很多时候势函数由于很大造成溢出，采用取对数的方法可以防止溢出。
3. double LogPartition(const VInt &num, const VReal &data);
带系数
4. double Quadratic(const V &x, const V &y, const M &w);
5. double InnerProd(const V1 &x, const V2 &y);

Chapter 2

ML

2.1 document.h

2.2 rbm

2.2.1 ais.h

1. `double LogPartition(int doc_len, int word_num, const RepSoftMax &rep);`

Chapter 3

Sparse Matrix

1. class Triplet

- Triplet() : m_row(0), m_col(0), m_value(0)
- Triplet(const Index& i, const Index& j, const Scalar& v = Scalar(0))
- const Index & col () const
- const Index & row () const
- const Scalar & value () const

2. Sparse Matrix

用稀疏的方式表达矩阵，

- binaryExpr (const SparseMatrixBase< OtherDerived > &other, const CustomBinaryOp &func = CustomBinaryOp())
二值操作, Eigen有一系列的二值运算，输入一个Eigen对象，逐个元素进行 运算，返回一个相应的运算对象
- size()
返回矩阵大小，这里的大小是矩阵的真正大小，不是压缩后的大小。
- resize(int row, int col)
这里的resize和stl的resize不太一样，resize的结果是矩阵的真实大小，不是存储的大小，因此size()返回的就是resize()时候设置的
- 元素访问， SpMat有两层，外层和内层，如果是列优先，外层指的是每列的起始 位置，内层指的是低维。
innerSize()返回内部维数大小， cols()返回列数。 innerSize()在列优先矩阵里面相当于rows()
- setFromTriplets(beg, end);
SpMat很重要的一个函数，输入是三元组，运行该函数前需要首先进行 resize(), 返回一个压缩矩阵
- reserve()
预留空间，在insert前使用，预告估计要分配的元素， 可以使得insert的时间最快。
- insert(rows, cols)
返回一个引用，用于存放值，在不重新分配空间的情况下， 可以在对数时间完成，如果是排好序的，可以在常数时间完成? eigen的默认情况是，插入一个新值，矩阵不再是压缩形式，预留两个 空间。

References