BJJ Model Simulation代码说明

最后修改：George-Gate @2015/11/29

# 代码文件简介

## generateFockOperators.m

生成Fock表象下的粒子湮灭算符，在进行计算时通常需要先调用此代码初始化工作空间。本代码的注释中有说明基态的编码方式。

## generateProjectionOperators.m

生成若干投影算符。由于使用投影算符进行计算的效率较低，这个代码建议只在想偷懒时使用，大规模计算请直接读取态矢量或者密度矩阵的对应分量。

## generateEigenStates.m

生成当前哈密顿量的能量本征态。程序会直接从内存中读取矩阵H，然后对其进行特征值分解。分解后，自动剔除总粒子数不等于N0的本征态，并对剩下的本征态按照本征值大小升序排序。本征态记录在V矩阵中，本征值记录在向量D中。

## BJJGroundState\_Fock.m

生成BJJ哈密顿量的基态，并显示出来。实际上调用的是makeState()函数。

## adiabaticEvolution\_Fock.m

薛定谔方程的演化，无粒子数损失。使用二阶精度差分格式，自适应步长算法，算法说明见OneNote。

## adiabaticEvolution\_Fock\_fixStep.m

功能与adiabaticEvolution\_Fock.m基本相同，但是使用定步长的演化。

## masterEQEvolution\_Fock\_fixStep.m

Master Equation的演化，无粒子数损失。使用二阶精度的差分格式，固定步长。

## masterEQEvolutionWithLoss\_Fock.m

Master Equation的演化，有粒子数损失。使用二阶精度的差分格式，自适应步长。步长变换算法见OneNote。

## masterEQEvolutionWithLoss\_Fock\_fixStep.m

功能与masterEQEvolutionWithLoss\_Fock.m基本相同，但是使用定步长的演化。

## energySpectrum.m

计算BJJ model哈密顿量的能谱图。可以计算不同J、Ec组合下的哈密顿量的能量本征值分布，并以J为参量画出能谱图。

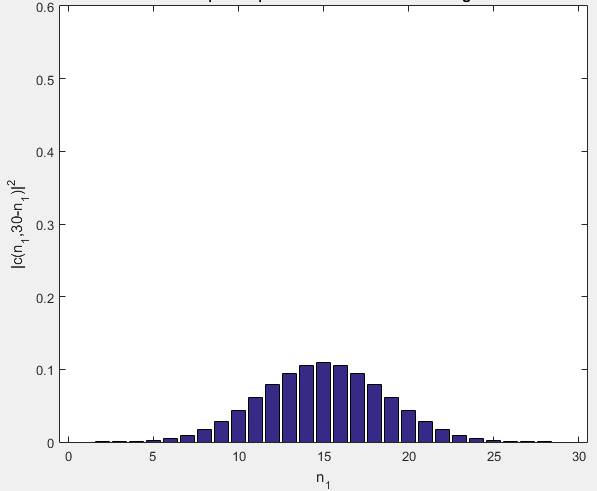
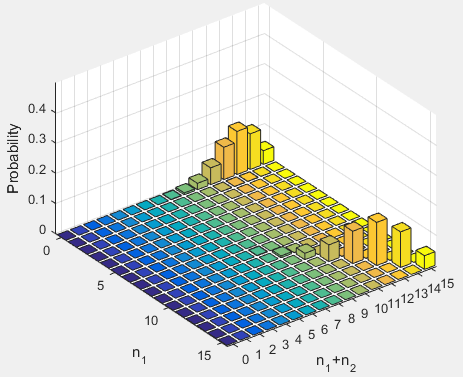
# 函数简介

## displayFockState()

将一个态矢量在Fock基上展开，并显示在Command Window中。

## plotFockState()

将一个态矢量或者密度矩阵的分量用柱状图的形式显示出来。下面是一些例子：



## makeState()

按照要求生成需要的态矢量或者密度矩阵。目前可以生成的态有SCS， ”粒子数确定的基态”以及给定哈密顿量的任意本征态。

# 全局变量说明

## [N]

类型：整数

描述：编码空间的总粒子数。注意，并不是态的总粒子数，psi的总粒子数可以<=N，也可以总粒子数不固定。比如说N=4的时候，可以有，但是中不可以出现这种总粒子数大于4的分量。

## [Dim]

类型：整数

描述：Dim=(N+1)(N+2)/2，是Fock表象下希尔伯特空间的维数。

可调用generateFockOperators.m生成。

## [k2nn]

类型：2 x Dim 整数

描述：程序中psi的下标与Fock态表示中的两个下标之间的转换关系。

psi(k)对应

可调用generateFockOperators.m生成。

## [nn2k]

类型：(N+1) x (N+1) 整数

描述：程序中psi的下标与Fock态表示中的两个下标之间的转换关系。

对应psi( nn2k(**n1+1**, **n2+1**) )

注意：下标加了1！

可调用generateFockOperators.m生成。

## [a1]

类型：Dim x Dim 复数

描述：Fock表象下Mode 1的湮灭算符。可调用generateFockOperators.m生成。

## [a2]

类型：Dim x Dim 复数

描述：Fock表象下Mode 2的湮灭算符。可调用generateFockOperators.m生成。

## [proj]

类型：(N+1) x (N+1) cell, 每个cell元素为Dim x Dim的矩阵

描述：Fock态的投影算符。。的元素为空。

可调用generateProjectionOperators.m生成。

## [projNOON]

类型：N x 1 cell, 每个cell元素为Dim x Dim的矩阵

描述：NOON态的投影算符。。

可调用generateProjectionOperators.m生成。