My Thesis

Joe Science

Department of Physics and Astronomy

University of Leeds

A thesis submitted for the degree of

Doctor of Philosophy

30th January 3000

This thesis is dedicated to...

Acknowledgements

My thanks to...

Abstract

My abstract in here...

Abbreviations

k_B	Boltzmann's constant
$k_B T$	Thermal energy
...	...

Contents

List of Figures

Chapter 1

Introduction

1.1 Introduction

Defining and distinguishing phases of matter has been a continuing effort by physicists for many years.

1.2 Free Fermion Systems

1.2.1 Dirac Fermions

The basic constituent of fermionic systems is the Dirac fermion. In the language of quantum field theory, a Dirac fermion can be represented by a second quantised field operator a, called the annihilation operator, and its conjugate partner $a_i^\dagger$, called the creation operator. They obey the anticommutation relations $\{a, a^\dagger\} = 1$ and $\{a^{(\dagger)}, a^{(\dagger)}\} = 0$. These operators act on a number state, also known as a Fock state, in the following way

$$\begin{aligned} a\left|0\right\rangle &= 0 & a^\dagger\left|0\right\rangle &= \left|1\right\rangle \\ a\left|1\right\rangle &= \left|0\right\rangle & a^\dagger\left|1\right\rangle &= 0 \end{aligned} \tag{1.1}$$

which follow from the commutation relations. A general state of a system consisting of a single Dirac fermion can be written in the form

$$\left|\psi\right\rangle = \alpha_0\left|0\right\rangle + \alpha_1\left|1\right\rangle, \tag{1.2}$$

where $\alpha_0, \alpha_1 \in \mathbb{C}$.

Given $N \in \mathbb{N}^+$ Dirac fermions, they can be represented by a set of second quantised fermionic field operators $\{a_i\}$ and their conjugate partners $\{a_i^\dagger\}$, where $i = 1, ..., N$. They obey the following commutation relations

$$\{a_i, a_j^\dagger\} = \delta_{ij} \qquad\qquad \{a_i^{(\dagger)}, a_j^{(\dagger)}\} \tag{1.3}$$

where δ_{ij} Kronercker delta function. These operators act on a tensor product of Fock states. A general state of the fermionic system, $|\psi\rangle$ can be written as

$$|\psi\rangle = \sum_{n_i=0,1} \left(\alpha_{n_1,\ldots,n_N} \bigotimes_{i=1}^{N} |n_i\rangle \right), \tag{1.4}$$

where $\alpha_{n_1,\ldots,n_N} \in \mathbb{C}$ and

$$\bigotimes_{i=1}^{N} |n_i\rangle = \left(\bigotimes_{i=1}^{N} \left(a_i^\dagger\right)^{n_i} \right) \left(\bigotimes_{i=1}^{N} |0\rangle \right). \tag{1.5}$$

1.2.2 Quadratic Hamiltonians

Chapter 2

Decomposition of the Chern Number

2.1 Introduction

2.1.1 Background

Appendix A

Code samples

A.1 Random Number Generator

The Bayes Durham Shuffle ensures that the psuedo random numbers used in the simulation are further shuffled, ensuring minimal correlation between subsequent random outputs (Press *et al.*, 1992).

```
#define IM1 2147483563
#define IM2 2147483399
#define AM (1.0/IM1)
#define IMM1 (IM1-1)
#define IA1 40014
#define IA2 40692
#define IQ1 53668
#define IQ2 52774
#define IR1 12211
#define IR2 3791
#define NTAB 32
#define NDIV (1+IMM1/NTAB)
#define EPS 1.2e-7
#define RNMX (1.0 - EPS)

double ran2(long *idum)
{
```

```
/*-----------------------------------------------------*/
/* Minimum Standard Random Number Generator          */
/* Taken from Numerical recipies in C                */
/* Based on Park and Miller with Bays Durham Shuffle */
/* Coupled Schrage methods for extra periodicity     */
/* Always call with negative number to initialise    */
/*-----------------------------------------------------*/

int j;
long k;
static long idum2=123456789;
static long iy=0;
static long iv[NTAB];
double temp;

if (*idum <=0)
{
  if (-(*idum) < 1)
  {
    *idum = 1;
  }else
  {
    *idum = -(*idum);
  }
  idum2=(*idum);
  for (j=NTAB+7;j>=0;j--)
  {
    k = (*idum)/IQ1;
    *idum = IA1 *(*idum-k*IQ1) - IR1*k;
    if (*idum < 0)
    {
      *idum += IM1;
    }
    if (j < NTAB)
```

```
      {
        iv[j] = *idum;
      }
    }
    iy = iv[0];
  }
  k = (*idum)/IQ1;
  *idum = IA1*(*idum-k*IQ1) - IR1*k;
  if (*idum < 0)
  {
    *idum += IM1;
  }
  k = (idum2)/IQ2;
  idum2 = IA2*(idum2-k*IQ2) - IR2*k;
  if (idum2 < 0)
  {
    idum2 += IM2;
  }
  j = iy/NDIV;
  iy=iv[j] - idum2;
  iv[j] = *idum;
  if (iy < 1)
  {
    iy += IMM1;
  }
  if ((temp=AM*iy) > RNMX)
  {
    return RNMX;
  }else
  {
    return temp;
  }
}
```

References

Nielsen, S., Lopez, C., Srinivas, G. & Klein, M. (2004). Coarse grain models and the computer simulation of soft materials. *J. Phys. Condens. Matter*, **16**, R481–R512.

Press, W. *et al.* (1992). *Numerical recipes in C*. Cambridge University Press Cambridge. 4