IX. SUPPLEMENTARY INFORMATION: QISKIT CODES FOR OPTIMIZATION.

For Energy estimation Z_0I , IZ_1 and Z_0Z_1 terms 58 of Hamiltonian respectively are coded then optimized 59 through scipy.optimize which contain 'powell', 'neldermead' or 'cobyla'. The QASM code for the same is as 60 follows:

```
Code for Z1(Z_0I):
```

```
# -*- coding: utf-8 -*-
  Created on Wed Feb 13 20:04:16 2019
   @author: Rahul
6
   from scipy.optimize import minimize
   from qiskit import QuantumCircuit,
9
        Classical Register \;,\;\; Quantum Register
10 import numpy as np
   from qiskit import execute
12 from qiskit import BasicAer
backend = BasicAer.get_backend('qasm_simulator')
_{14} T=8192
def Z1(theta):
        # Create a Quantum Register called "q" with
16
        3 qubits
        q = QuantumRegister(2)
17
18
        # Create a Classical Register called "c"
19
        with 3 bits
        c = ClassicalRegister(2)
20
        qc = QuantumCircuit(q, c)
21
        qc.u1(theta[0], q[0])
23
        qc.u3(theta[1],-np.pi/2,np.pi/2,q[0])
qc.u1(theta[2],q[0])
24
        qc.cx(q[0], q[1])
qc.u1(theta[3],q[1])
qc.u3(theta[4],-np.pi/2,np.pi/2,q[1])
qc.u1(theta[5],q[1])
26
27
28
29
        qc.cx(q[1], q[0])
        qc.u1(theta[6],q[0])
qc.u1(theta[7],q[1])
        qc.u3(theta[8],-np.pi/2,np.pi/2,q[0])
qc.u3(theta[9],-np.pi/2,np.pi/2,q[1])
34
        qc.u1(theta[10],q[0])
        qc.ul(theta[11],q[1])
36
37
        qc.z(q[0])
38
        \begin{array}{lll} qc.measure\left(q\left[0\right], & c\left[0\right]\right) \\ qc.measure\left(q\left[1\right], & c\left[1\right]\right) \end{array}
40
41
        #print(qc)
42
        #print(i)
43
        shots = T
                           # Number of shots to run the
44
        program (experiment); maximum is 8192 shots.
        max\_credits = 3
                                    # Maximum number of
45
        credits to spend on executions.
        job_hpc = execute(qc, backend=backend, shots
47
        =shots, max_credits=max_credits)
        result_hpc = job_hpc.result()
        counts11 = result_hpc.get_counts(qc)
49
        #print(counts11)
50
        Z=0
        if '00' in list (counts11):
             Z=Z+counts11['00']/T
```

```
\begin{array}{c} \#print\left(Z\right)\\ \text{if } \ \ '01\, ' \ \ \text{in } \ \ list\left(counts11\right): \end{array}
              Z=Z+counts11['01']/T
              #print(Z)
         if '10' in list (counts11):
              Z=Z-counts11['10']/T
              #print(Z)
         if '11' in list (counts11):
              Z=Z-counts11['11']/T
             #print(Z)
64
         return Z
   theta0 = [0, np. pi/2, 0, 0, np. pi/2, 0, 0, 0, np. pi/2, np.
        pi/2,0,0]
   {\tt res = minimize(Z1, theta0, method='powell',}\\
   options={'xtol': 1e-8', 'disp': True})
print(res, 'Z1')
\#x=Z1 ( [ 1.53637770e-03, 1.55027283e+00,
        8.09473120\,\mathrm{e}{-04},\quad 1.66340534\,\mathrm{e}{-04}, -8.32420733\,\mathrm{e}
         -03, -2.28166637e-05, -8.94140209e-04,
        5.61023344e - 04, 1.77675920e + 00, 1.58425510e
        +00, 1.09455026e-03, 8.51956374e-04])
70 #print(x)
```

Code for $Z2(IZ_1)$:

```
# -*- coding: utf-8 -*-
   Created on Wed Feb 13 20:04:17 2019
   @author: Rahul
   from scipy.optimize import minimize
   from qiskit import QuantumCircuit,
        ClassicalRegister, QuantumRegister
  import numpy as np
   from qiskit import execute
12 from qiskit import BasicAer
backend = BasicAer.get_backend('qasm_simulator')
14 T=8192
def Z2(theta):
        # Create a Quantum Register called "q" with
16
        3 qubits
        q = QuantumRegister(2)
18
        # Create a Classical Register called "c"
19
        with 3 bits
        c = ClassicalRegister(2)
        qc = QuantumCircuit(q, c)
21
        qc.ul(theta[0],q[0])
        qc.u3(theta[1],-np.pi/2,np.pi/2,q[0])
qc.u1(theta[2],q[0])
24
        qc.cx(q[0], q[1])
qc.u1(theta[3],q[1])
26
        qc.u3(theta[4],-np.pi/2,np.pi/2,q[1])
qc.u1(theta[5],q[1])
28
        qc.cx(q[1], q[0])
qc.u1(theta[6],q[0])
qc.u1(theta[7],q[1])
        qc.u3(theta[8],-np.pi/2,np.pi/2,q[0])
qc.u3(theta[9],-np.pi/2,np.pi/2,q[1])
        qc.u1(theta[10],q[0])
        qc.u1(theta[11],q[1])
36
        qc.z(q[1])
38
        \begin{array}{lll} qc.measure(q[0]\,, & c[0]) \\ qc.measure(q[1]\,, & c[1]) \end{array}
39
40
41
        #print(qc)
```

```
qc.u3(theta[4], -np.pi/2, np.pi/2, q[1])
       #print(i)
43
                                # Number of shots to run
        shots = T
                                                                        qc.ul(theta[5], q[1])
44
                                                                        qc.cx(q[1], q[0])

qc.u1(theta[6], q[0])
         the program (experiment); maximum is 8192
                                                                30
        shots.
        max\_credits = 3
                                    # Maximum number of
                                                                        qc.ul(theta[7], q[1])
45
                                                                        \begin{array}{l} qc.u3(theta[8],-np.pi/2,np.pi/2,q[0]) \\ qc.u3(theta[9],-np.pi/2,np.pi/2,q[1]) \end{array}
        credits to spend on executions.
                                                                34
46
                                                                        qc.u1(theta[10],q[0])
       job_hpc = execute(qc, backend=backend, shots
47
                                                                35
        =shots, max_credits=max_credits)
                                                                36
                                                                        qc.ul(theta[11],q[1])
        result_hpc = job_hpc.result()
                                                                        qc.z(q[0])
48
                                                                        qc.z(q[1])
        counts22 = result_hpc.get_counts(qc)
                                                                38
49
                                                                        \begin{array}{ll} qc.measure\left(q\left[0\right], & c\left[0\right]\right) \\ qc.measure\left(q\left[1\right], & c\left[1\right]\right) \end{array}
51
                                                                40
           '00' in list (counts22):
                                                                41
            Z=Z+counts22['00']/T
                                                                42
           print(Z)
'01' in list(counts22):
                                                                        #print(qc)
54
                                                                43
                                                                44
            Z=Z-counts22['01']/T
                                                                        shots= T
                                                                                              # Number of shots to run
                                                                45
                                                                        the program (experiment); maximum is 8192
             print(Z)
           ,10, in list (counts22):
58
            Z=Z+counts22['10']/T
                                                                                                    # Maximum number of
                                                                        max\_credits = 3
59
                                                                46
        \begin{array}{c} \operatorname{print}\left(Z\right) \\ \operatorname{if} \end{array} '11' in list(counts22):
                                                                        credits to spend on executions.
60
61
            Z=Z-counts22['11']/T
                                                                        job_hpc = execute(qc, backend=backend, shots
                                                                48
            print(Z)
                                                                        =shots, max_credits=max_credits)
                                                                        result_hpc = job_hpc.result()
64
        return Z
                                                                        counts12 = result_hpc.get_counts(qc)
65
   theta0 = [0, np. pi/2, 0, 0, np. pi/2, 0, 0, 0, np. pi/2, np.
                                                                51
66
        pi / 2, 0, 0]
  res = minimize(Z2, theta0, method='nelder-mead',
    options={'xtol': 1e-8, 'disp': True})
print(res, 'Z2')
                                                                        if '00' in list (counts12):
                                                                             Z=counts12[;00;]/T
                                                                             print(Z)
                                                                        if '01' in list (counts12):
69 \#z=Z2([-5.64200351e-01, 1.61554986e+00,
                                                                56
        1.56821823e+00, 1.61219634e-03, 1.81902336e
                                                                57
                                                                             Z=Z-counts12['01']/T
        +00, 5.96777406e+00, -6.66574506e-03,
                                                                             print(Z)
        4.63725496e+00, 1.32156756e+00, 3.78827977e
                                                                            '10' in list (counts12):
                                                                             Z=Z-counts12[ '10 ']/T
        -01, 9.59049221e+00, 3.90466495e+00])
                                                                60
                                                                        print(Z)
if '11' in list(counts12):
70 #print(z)
   Code for Z3(Z_0Z_I):
                                                                             Z=Z+counts12['11']/T
                                                                63
                                                                64
                                                                             print(Z)
  # -*- coding: utf-8 -*-
                                                                        return Z
2
                                                                theta0 = [0, np. pi/2, 0, 0, np. pi/2, 0, 0, 0, np. pi/2, np.
   Created on Wed Feb 13 20:04:15 2019
3
                                                                        pi/2,0,0]
                                                                       minimize(Z3, theta0, method='nelder-mead',
options={'xtol': le-8, 'disp': True})
   @author: Rahul
5
                                                                68 print (res, 'Z3')
69 #y=Z3([-7.63609374, -0.31633082, 2.15909817,
6
   from scipy.optimize import minimize
8
                                                                        from qiskit import QuantumCircuit,
       ClassicalRegister, QuantumRegister
   import numpy as np
                                                                70 #print(y)
11 from qiskit import execute
12 from qiskit import IBMQ
backend =IBMQ.get_backend('ibmqx4')
14 T=8192
  def Z3(theta):
15
        # Create a Quantum Register called "q" with
16
       3 qubits
        q = QuantumRegister(2)
18
       # Create a Classical Register called "c"
19
        with 3 bits
        c = ClassicalRegister(2)
20
        qc = QuantumCircuit(q,c)
21
        qc.u1(theta[0],q[0])
23
       qc.u3(theta[1],-np.pi/2,np.pi/2,q[0])
qc.u1(theta[2],q[0])
24
25
26
        qc.cx(q[0], q[1])
       qc.u1(theta[3],q[1])
```

The tabulated data of convergence is as shown below.

TABLE I. For Nelder Mead

Iterations	Theoretical	Experimental
1	-0.52952	-0.3658118514
2	-0.52952	-0.3687388385
3	-0.52952	-0.364456499
4	-0.52952	-0.3640980715
5	-0.52952	-0.3735459878
6	-0.52952	-0.3547868431
7	-0.52952	-0.3642996161
8	-0.52952	-0.3652836663
9	-0.52952	-0.3730938032
10	-0.52952	-0.3646086323
11	-0.52952	-0.3657974241
12	-0.52952	-0.3712501318
13	-0.52952	-0.3667064972
14	-0.52952	-0.3660282661
15	-0.52952	-0.3702449038
16	-0.52952	-0.3702913827
17	-0.52952	-0.3308180563
18	-0.52952	-0.3653252632
19	-0.52952	-0.3770401351
20	-0.52952	-0.3867578684
21	-0.52952	-0.3746037324
22	-0.52952	-0.3745729229
23	-0.52952	-0.375111539
24	-0.52952	-0.3759054584
25	-0.52952	-0.3819807294
26	-0.52952	-0.3751555337
27	-0.52952	-0.3774191599
28	-0.52952	-0.3790968918
29	-0.52952	-0.3830969824
30	-0.52952	-0.3842554535
31	-0.52952	-0.3821377446
32	-0.52952	-0.3852416368
33	-0.52952	-0.387955228
34	-0.52952	-0.3997053027
35	-0.52952	-0.3925481822
36	-0.52952	-0.3912610618
37	-0.52952	-0.3925515116
38	-0.52952	-0.39142132
39	-0.52952	-0.3948359592
40	-0.52952	-0.3993956196
41	-0.52952	-0.3929185069
42	-0.52952	-0.3984600897
43	-0.52952	-0.3981955975
44	-0.52952	-0.3995114402
45	-0.52952	-0.3982939261
46	-0.52952	-0.4039865324
47	-0.52952	-0.4040763391
48	-0.52952	-0.4061567579
49	-0.52952	-0.4068460868
50	-0.52952	-0.4075465596

T	[m]	In
Iterations	Theoretical	Experimental
51	-0.52952	-0.4042369078
52	-0.52952	-0.4167492066
53	-0.52952	-0.4100107529
54	-0.52952	-0.4121749407
55	-0.52952	-0.4123393275
56	-0.52952	-0.4153778284
57	-0.52952	-0.4201981629
58	-0.52952	-0.416993412
59	-0.52952	-0.4075684502
60	-0.52952	-0.4216554432
61	-0.52952	-0.4229979668
62	-0.52952	-0.4196147123
63	-0.52952	-0.4253503764
64	-0.52952	-0.4249917301
65	-0.52952	-0.4299162527
66	-0.52952	-0.4330937475
67	-0.52952	-0.4257880472
68	-0.52952	-0.4038551852
69	-0.52952	-0.424561212
70	-0.52952	-0.4276609296
71	-0.52952	-0.4244919108
72	-0.52952	-0.4281787243
73	-0.52952	-0.4255140557
74	-0.52952	-0.4291707672
75	-0.52952	-0.4413052833
76	-0.52952	-0.4357576628
77	-0.52952	-0.4398903991
78	-0.52952	-0.4376459041
79	-0.52952	-0.4434627664
80	-0.52952	-0.4494898276
81	-0.52952	-0.445895972
82	-0.52952	-0.4480489295
83	-0.52952	-0.448329137
84	-0.52952	-0.4582959119
85	-0.52952	-0.4535441657
86	-0.52952	-0.4562919823
87	-0.52952	-0.458321483
88	-0.52952	-0.4572042524
89	-0.52952	-0.4542379797
90	-0.52952	-0.4542379797
90	-0.52952	-0.4615284535
92	-0.52952	-0.4654706567
93	-0.52952	-0.4647630364
94	-0.52952	-0.4646672379
95	-0.52952	-0.4587078335
96	-0.52952	-0.4644073631
97	-0.52952	-0.458071464
98	-0.52952	-0.4648104928
99	-0.52952	-0.455790524
100	-0.52952	-0.4643548465

Iterations	Theoretical	Experimental
101	-0.52952	-0.4597123945
102	-0.52952	-0.4614880122
103	-0.52952	-0.4640232781
104	-0.52952	-0.4631510063
105	-0.52952	-0.4633038065
106	-0.52952	-0.4607956644
107	-0.52952	-0.4640445423
108	-0.52952	-0.4648928415
109	-0.52952	-0.464328522
109	-0.52952	-0.4654582708
110	-0.52952	-0.4653556353
111	-0.52952	-0.4637653123
112	-0.52952	-0.4627551564
113	-0.52952	-0.4616840943
114	-0.52952	-0.4639483874
115	-0.52952	-0.4658633096
116	-0.52952	-0.4652171301
117	-0.52952	-0.4662359862
118	-0.52952	-0.4657969867
119	-0.52952	-0.4654209807
120	-0.52952	-0.4644860719
121	-0.52952	-0.4655182454
122	-0.52952	-0.4639068364
123	-0.52952	-0.4650023139
124	-0.52952	-0.4634304145
125	-0.52952	-0.4670682595

TABLE II. For Cobyla on real Chip

2 -0.52952 -0.312141158 3 -0.52952 -0.3000890929 4 -0.52952 -0.2012450742 5 -0.52952 -0.3431368385 6 -0.52952 -0.3354519405 7 -0.52952 -0.3092321975 8 -0.52952 -0.3092321975 9 -0.52952 -0.3092321975 9 -0.52952 -0.3092321975 9 -0.52952 -0.3092321975 9 -0.52952 -0.3421126117 10 -0.52952 -0.365973851 12 -0.52952 -0.365973851 12 -0.52952 -0.377303690787 13 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 17 -0.52952 -0.3760271897 18 -0.52952 -0.3703690787 18 -0.52952 -0.3760271897 19 <td< th=""><th>Iterations</th><th>Theoretical</th><th>Experimental</th></td<>	Iterations	Theoretical	Experimental
3 -0.52952 -0.3000890929 4 -0.52952 -0.2012450742 5 -0.52952 -0.3431368385 6 -0.52952 -0.3354519405 7 -0.52952 -0.3605707275 8 -0.52952 -0.3092321975 9 -0.52952 -0.3092321975 9 -0.52952 -0.3092321975 10 -0.52952 -0.3421126117 10 -0.52952 -0.365973851 11 -0.52952 -0.365973851 12 -0.52952 -0.378163368 14 -0.52952 -0.377308508 15 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 17 -0.52952 -0.3760271897 18 -0.52952 -0.3760271897 19 -0.52952 -0.3760271897 19 -0.52952 -0.3757416573 20 <td< td=""><td>1</td><td>-0.52952</td><td>-0.3242458015</td></td<>	1	-0.52952	-0.3242458015
4 -0.52952 -0.2012450742 5 -0.52952 -0.3431368385 6 -0.52952 -0.3354519405 7 -0.52952 -0.3605707275 8 -0.52952 -0.3092321975 9 -0.52952 -0.3421126117 10 -0.52952 -0.365973851 10 -0.52952 -0.365973851 11 -0.52952 -0.365973851 12 -0.52952 -0.378163368 14 -0.52952 -0.378163368 14 -0.52952 -0.372308508 15 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 17 -0.52952 -0.3760271897 18 -0.52952 -0.377360271897 18 -0.52952 -0.3703690787 18 -0.52952 -0.3760271897 19 -0.52952 -0.3760271897 10 -0.52952 -0.3659191554 20 <	2	-0.52952	-0.312141158
5 -0.52952 -0.3431368385 6 -0.52952 -0.3354519405 7 -0.52952 -0.3605707275 8 -0.52952 -0.3092321975 9 -0.52952 -0.3421126117 10 -0.52952 -0.382317455 11 -0.52952 -0.365973851 12 -0.52952 -0.365973851 12 -0.52952 -0.378163368 14 -0.52952 -0.3772308508 15 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 17 -0.52952 -0.3760271897 18 -0.52952 -0.3760271897 19 -0.52952 -0.3760271897 10 -0.52952 -0.3760271897 11 -0.52952 -0.3760271897 12 -0.52952 -0.3760271897 13 -0.52952 -0.3757416573 20	3	-0.52952	-0.3000890929
6 -0.52952 -0.3354519405 7 -0.52952 -0.3605707275 8 -0.52952 -0.3092321975 9 -0.52952 -0.3421126117 10 -0.52952 -0.382317455 11 -0.52952 -0.365973851 12 -0.52952 -0.365973851 12 -0.52952 -0.378163368 14 -0.52952 -0.378163368 14 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 17 -0.52952 -0.3760271897 18 -0.52952 -0.3760271897 19 -0.52952 -0.3461754311 19 -0.52952 -0.3461754311 19	4	-0.52952	-0.2012450742
6 -0.52952 -0.3354519405 7 -0.52952 -0.3605707275 8 -0.52952 -0.3092321975 9 -0.52952 -0.3421126117 10 -0.52952 -0.382317455 11 -0.52952 -0.365973851 12 -0.52952 -0.365973851 12 -0.52952 -0.378163368 14 -0.52952 -0.378163368 14 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 17 -0.52952 -0.3760271897 18 -0.52952 -0.3760271897 19 -0.52952 -0.3461754311 19 -0.52952 -0.3461754311 19	5		-0.3431368385
7 -0.52952 -0.3605707275 8 -0.52952 -0.3092321975 9 -0.52952 -0.3421126117 10 -0.52952 -0.382317455 11 -0.52952 -0.365973851 12 -0.52952 -0.3070352964 13 -0.52952 -0.378163368 14 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.37760271897 16 -0.52952 -0.3703690787 17 -0.52952 -0.3703690787 18 -0.52952 -0.3703690787 18 -0.52952 -0.3760271897 19 -0.52952 -0.3760271897 10 -0.52952 -0.3760271897 11 -0.52952 -0.3760271897 12 -0.52952 -0.3757416573 14 -0.52952 -0.3805747747 23 -0.52952 -0.3807448887 24		-0.52952	-0.3354519405
9 -0.52952 -0.3421126117 10 -0.52952 -0.382317455 11 -0.52952 -0.365973851 12 -0.52952 -0.3070352964 13 -0.52952 -0.378163368 14 -0.52952 -0.378308508 15 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 17 -0.52952 -0.3703690787 18 -0.52952 -0.3703690787 18 -0.52952 -0.3461754311 19 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.365919154 21 -0.52952 -0.365919154 20 -0.52952 -0.3461754311 19 -0.52952 -0.3401756666 22 -0.52952 -0.3401756666 22 -0.52952 -0.3807448887 24	7	-0.52952	-0.3605707275
9 -0.52952 -0.3421126117 10 -0.52952 -0.382317455 11 -0.52952 -0.365973851 12 -0.52952 -0.3070352964 13 -0.52952 -0.378163368 14 -0.52952 -0.372308508 15 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3703690787 17 -0.52952 -0.3703690787 18 -0.52952 -0.3703690787 18 -0.52952 -0.365919154 20 -0.52952 -0.365919154 20 -0.52952 -0.365919154 20 -0.52952 -0.365919154 20 -0.52952 -0.365919154 21 -0.52952 -0.365919154 20 -0.52952 -0.365919154 21 -0.52952 -0.3757416573 21 -0.52952 -0.380774777 23 -0.52952 -0.38074748887 24 <td< td=""><td>8</td><td>-0.52952</td><td>-0.3092321975</td></td<>	8	-0.52952	-0.3092321975
10 -0.52952 -0.382317455 11 -0.52952 -0.365973851 12 -0.52952 -0.3070352964 13 -0.52952 -0.378163368 14 -0.52952 -0.372308508 15 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3703690787 17 -0.52952 -0.3703690787 18 -0.52952 -0.3461754311 19 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.365919154 21 -0.52952 -0.3659191554 20 -0.52952 -0.3401756666 22 -0.52952 -0.3805747747 23 -0.52952 -0.3807448887 24			
11 -0.52952 -0.365973851 12 -0.52952 -0.3070352964 13 -0.52952 -0.378163368 14 -0.52952 -0.372308508 15 -0.52952 -0.3760271897 16 -0.52952 -0.3703690787 16 -0.52952 -0.3703690787 17 -0.52952 -0.3703690787 18 -0.52952 -0.3461754311 19 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3461754666 21 -0.52952 -0.3401756666 22 -0.52952 -0.380747477 23 -0.52952 -0.3807448887 24 -0.52952 -0.3807448887 25 -0.52952 -0.3807448887 26			
12 -0.52952 -0.3070352964 13 -0.52952 -0.378163368 14 -0.52952 -0.372308508 15 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 16 -0.52952 -0.3760271897 17 -0.52952 -0.3703690787 18 -0.52952 -0.3461754311 19 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3461754666 21 -0.52952 -0.3401756666 22 -0.52952 -0.3807474677 23 -0.52952 -0.3807448887 24 -0.52952 -0.3807448887 25 -0.52952 -0.3807448887 26 -0.52952 -0.3807448887 27		-0.52952	
13 -0.52952 -0.378163368 14 -0.52952 -0.372308508 15 -0.52952 -0.3760271897 16 -0.52952 -0.2783118205 17 -0.52952 -0.3703690787 18 -0.52952 -0.3461754311 19 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3461754311 19 -0.52952 -0.3461754311 20 -0.52952 -0.3401756666 21 -0.52952 -0.3401756666 22 -0.52952 -0.3805747747 23 -0.52952 -0.3638238258 24 -0.52952 -0.3789247876 25 -0.52952 -0.38057447876 26 -0.52952 -0.38779779816 27 -0.52952 -0.3779779816 28 -0.52952 -0.3805431712 29			
14 -0.52952 -0.372308508 15 -0.52952 -0.3760271897 16 -0.52952 -0.2783118205 17 -0.52952 -0.3703690787 18 -0.52952 -0.3461754311 19 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3757416573 21 -0.52952 -0.3401756666 22 -0.52952 -0.3805747747 23 -0.52952 -0.3805747747 23 -0.52952 -0.3638238258 24 -0.52952 -0.3638238258 24 -0.52952 -0.3805747747 23 -0.52952 -0.389247876 25 -0.52952 -0.3807448887 26 -0.52952 -0.38913352043 27 -0.52952 -0.377797816 28 -0.52952 -0.3779779816 29 -0.52952 -0.3805431712 30 -0.52952 -0.3805431712 31			
15 -0.52952 -0.3760271897 16 -0.52952 -0.2783118205 17 -0.52952 -0.3703690787 18 -0.52952 -0.3461754311 19 -0.52952 -0.3659191554 20 -0.52952 -0.3659191554 20 -0.52952 -0.3757416573 21 -0.52952 -0.3401756666 22 -0.52952 -0.3805747747 23 -0.52952 -0.3805747747 23 -0.52952 -0.3638238258 24 -0.52952 -0.3638238258 24 -0.52952 -0.3807448887 25 -0.52952 -0.3807448887 26 -0.52952 -0.3807448887 27 -0.52952 -0.3877979816 28 -0.52952 -0.3779779816 28 -0.52952 -0.3759803137 30 -0.52952 -0.3805431712 31 -0.52952 -0.3805431712 32 -0.52952 -0.38928037149 33 <td></td> <td></td> <td></td>			
16 -0.52952 -0.2783118205 17 -0.52952 -0.3703690787 18 -0.52952 -0.3461754311 19 -0.52952 -0.3659191554 20 -0.52952 -0.3757416573 21 -0.52952 -0.3401756666 22 -0.52952 -0.3805747747 23 -0.52952 -0.3638238258 24 -0.52952 -0.3638238258 24 -0.52952 -0.3789247876 25 -0.52952 -0.3807448887 26 -0.52952 -0.3807448887 26 -0.52952 -0.3913352043 27 -0.52952 -0.3913352043 27 -0.52952 -0.3913352043 28 -0.52952 -0.3963448298 29 -0.52952 -0.3759803137 30 -0.52952 -0.3805431712 31 -0.52952 -0.3805431712 32 -0.52952 -0.385364926 34 -0.52952 -0.3848496156 36			
17 -0.52952 -0.3703690787 18 -0.52952 -0.3461754311 19 -0.52952 -0.3659191554 20 -0.52952 -0.3757416573 21 -0.52952 -0.3401756666 22 -0.52952 -0.3805747747 23 -0.52952 -0.3805747747 23 -0.52952 -0.3638238258 24 -0.52952 -0.3789247876 25 -0.52952 -0.3807448887 26 -0.52952 -0.3807448887 26 -0.52952 -0.3913352043 27 -0.52952 -0.39737979816 28 -0.52952 -0.3963448298 29 -0.52952 -0.3963448298 29 -0.52952 -0.3827794955 31 -0.52952 -0.3827794955 31 -0.52952 -0.3805431712 32 -0.52952 -0.3805431712 32 -0.52952 -0.38938037149 33 -0.52952 -0.3848496156 36 <td></td> <td></td> <td></td>			
18 -0.52952 -0.3461754311 19 -0.52952 -0.3659191554 20 -0.52952 -0.3757416573 21 -0.52952 -0.3401756666 22 -0.52952 -0.3805747747 23 -0.52952 -0.3638238258 24 -0.52952 -0.3789247876 25 -0.52952 -0.3807448887 26 -0.52952 -0.3807448887 26 -0.52952 -0.3913352043 27 -0.52952 -0.3913352043 27 -0.52952 -0.39779779816 28 -0.52952 -0.39737977916 28 -0.52952 -0.3963448298 29 -0.52952 -0.3759803137 30 -0.52952 -0.3827794955 31 -0.52952 -0.3805431712 32 -0.52952 -0.3805431712 32 -0.52952 -0.3893637149 33 -0.52952 -0.3848496156 34 -0.52952 -0.3848496156 36 <td></td> <td></td> <td></td>			
19 -0.52952 -0.3659191554 20 -0.52952 -0.3757416573 21 -0.52952 -0.3401756666 22 -0.52952 -0.3805747747 23 -0.52952 -0.3638238258 24 -0.52952 -0.3789247876 25 -0.52952 -0.3807448887 26 -0.52952 -0.3913352043 27 -0.52952 -0.3913352043 27 -0.52952 -0.377979816 28 -0.52952 -0.377979816 29 -0.52952 -0.3963448298 29 -0.52952 -0.3963448298 29 -0.52952 -0.3759803137 30 -0.52952 -0.3827794955 31 -0.52952 -0.3805431712 32 -0.52952 -0.3805431712 32 -0.52952 -0.3893807149 33 -0.52952 -0.3848496156 34 -0.52952 -0.3848496156 36 -0.52952 -0.3882537216 37			
20 -0.52952 -0.3757416573 21 -0.52952 -0.3401756666 22 -0.52952 -0.3805747747 23 -0.52952 -0.3638238258 24 -0.52952 -0.3789247876 25 -0.52952 -0.3807448887 26 -0.52952 -0.3913352043 27 -0.52952 -0.3913352043 27 -0.52952 -0.3779779816 28 -0.52952 -0.3779779816 29 -0.52952 -0.3963448298 29 -0.52952 -0.3759803137 30 -0.52952 -0.3827794955 31 -0.52952 -0.3805431712 32 -0.52952 -0.3805431712 32 -0.52952 -0.38928037149 33 -0.52952 -0.385364926 34 -0.52952 -0.3848496156 36 -0.52952 -0.3882537216 37 -0.52952 -0.3883246723 38 -0.52952 -0.3732819491 39			
21 -0.52952 -0.3401756666 22 -0.52952 -0.3805747747 23 -0.52952 -0.3638238258 24 -0.52952 -0.3789247876 25 -0.52952 -0.3807448887 26 -0.52952 -0.3913352043 27 -0.52952 -0.397379779816 28 -0.52952 -0.3963448298 29 -0.52952 -0.3963448298 29 -0.52952 -0.3963448298 29 -0.52952 -0.3759803137 30 -0.52952 -0.3827794955 31 -0.52952 -0.3805431712 32 -0.52952 -0.3805431712 32 -0.52952 -0.38928037149 33 -0.52952 -0.385364926 34 -0.52952 -0.3848496156 36 -0.52952 -0.3882537216 37 -0.52952 -0.3883246723 38 -0.52952 -0.3732819491 39 -0.52952 -0.38917883664 40 </td <td></td> <td></td> <td></td>			
$\begin{array}{cccccccccccccccccccccccccccccccccccc$			
23 -0.52952 -0.3638238258 24 -0.52952 -0.3789247876 25 -0.52952 -0.3807448887 26 -0.52952 -0.3913352043 27 -0.52952 -0.3779779816 28 -0.52952 -0.3963448298 29 -0.52952 -0.39759803137 30 -0.52952 -0.3827794955 31 -0.52952 -0.382794955 32 -0.52952 -0.3805431712 32 -0.52952 -0.38928037149 33 -0.52952 -0.385364926 34 -0.52952 -0.3818327581 35 -0.52952 -0.3848496156 36 -0.52952 -0.3892537216 37 -0.52952 -0.3883246723 38 -0.52952 -0.3732819491 39 -0.52952 -0.3917883664 40 -0.52952 -0.3891585442 41 -0.52952 -0.3891585442 42 -0.52952 -0.3859512631 43			
$\begin{array}{cccccccccccccccccccccccccccccccccccc$			
25 -0.52952 -0.3807448887 26 -0.52952 -0.3913352043 27 -0.52952 -0.3779779816 28 -0.52952 -0.3963448298 29 -0.52952 -0.3759803137 30 -0.52952 -0.3827794955 31 -0.52952 -0.3805431712 32 -0.52952 -0.38928037149 33 -0.52952 -0.38928037149 33 -0.52952 -0.385364926 34 -0.52952 -0.3848496156 36 -0.52952 -0.3848496156 36 -0.52952 -0.3892537216 37 -0.52952 -0.3883246723 38 -0.52952 -0.3732819491 39 -0.52952 -0.3917883664 40 -0.52952 -0.3891585442 41 -0.52952 -0.3891585442 42 -0.52952 -0.3859512631 43 -0.52952 -0.3852036904 44 -0.52952 -0.3852036904 45 <td></td> <td></td> <td></td>			
$\begin{array}{cccccccccccccccccccccccccccccccccccc$			
$\begin{array}{cccccccccccccccccccccccccccccccccccc$			
28 -0.52952 -0.3963448298 29 -0.52952 -0.3759803137 30 -0.52952 -0.3827794955 31 -0.52952 -0.3805431712 32 -0.52952 -0.3928037149 33 -0.52952 -0.3928037149 34 -0.52952 -0.385364926 34 -0.52952 -0.3918327581 35 -0.52952 -0.3848496156 36 -0.52952 -0.3892537216 37 -0.52952 -0.3883246723 38 -0.52952 -0.3732819491 39 -0.52952 -0.3917883664 40 -0.52952 -0.3917883664 41 -0.52952 -0.3891585442 42 -0.52952 -0.3891585442 42 -0.52952 -0.3859512631 43 -0.52952 -0.3852036904 44 -0.52952 -0.3852036904 45 -0.52952 -0.3859246681 46 -0.52952 -0.3859063958 47			
29 -0.52952 -0.3759803137 30 -0.52952 -0.3827794955 31 -0.52952 -0.3805431712 32 -0.52952 -0.3928037149 33 -0.52952 -0.3928037149 34 -0.52952 -0.385364926 34 -0.52952 -0.3918327581 35 -0.52952 -0.3848496156 36 -0.52952 -0.3892537216 37 -0.52952 -0.3883246723 38 -0.52952 -0.3732819491 39 -0.52952 -0.3917883664 40 -0.52952 -0.3917883664 41 -0.52952 -0.3891585442 42 -0.52952 -0.3891585442 42 -0.52952 -0.3859512631 43 -0.52952 -0.3859512631 43 -0.52952 -0.3852036904 45 -0.52952 -0.3852036904 45 -0.52952 -0.3859246281 47 -0.52952 -0.3890963958			
30 -0.52952 -0.3827794955 31 -0.52952 -0.3805431712 32 -0.52952 -0.3928037149 33 -0.52952 -0.3928037149 34 -0.52952 -0.385364926 34 -0.52952 -0.3918327581 35 -0.52952 -0.3848496156 36 -0.52952 -0.3892537216 37 -0.52952 -0.3883246723 38 -0.52952 -0.3732819491 39 -0.52952 -0.3917883664 40 -0.52952 -0.3917883664 41 -0.52952 -0.3891585442 42 -0.52952 -0.3891585442 42 -0.52952 -0.3859512631 43 -0.52952 -0.3852036904 44 -0.52952 -0.3852036904 45 -0.52952 -0.3859246281 46 -0.52952 -0.3859246281 47 -0.52952 -0.3890963958			
31 -0.52952 -0.3805431712 32 -0.52952 -0.3928037149 33 -0.52952 -0.385364926 34 -0.52952 -0.3818327581 35 -0.52952 -0.3848496156 36 -0.52952 -0.3892537216 37 -0.52952 -0.3883246723 38 -0.52952 -0.3732819491 39 -0.52952 -0.3917883664 40 -0.52952 -0.3926158442 41 -0.52952 -0.38915855442 42 -0.52952 -0.3891585442 42 -0.52952 -0.3847456284 44 -0.52952 -0.3852036904 45 -0.52952 -0.3852036904 45 -0.52952 -0.3859246281 47 -0.52952 -0.3890963958			
$\begin{array}{cccccccccccccccccccccccccccccccccccc$			
41 -0.52952 -0.3891585442 42 -0.52952 -0.3859512631 43 -0.52952 -0.3947456284 44 -0.52952 -0.3852036904 45 -0.52952 -0.3942924663 46 -0.52952 -0.3859246281 47 -0.52952 -0.3890963958			
42 -0.52952 -0.3859512631 43 -0.52952 -0.3947456284 44 -0.52952 -0.3852036904 45 -0.52952 -0.3942924663 46 -0.52952 -0.3859246281 47 -0.52952 -0.3890963958			
43 -0.52952 -0.3947456284 44 -0.52952 -0.3852036904 45 -0.52952 -0.3942924663 46 -0.52952 -0.3859246281 47 -0.52952 -0.3890963958			
44 -0.52952 -0.3852036904 45 -0.52952 -0.3942924663 46 -0.52952 -0.3859246281 47 -0.52952 -0.3890963958			
45 -0.52952 -0.3942924663 46 -0.52952 -0.3859246281 47 -0.52952 -0.3890963958			
46 -0.52952 -0.3859246281 47 -0.52952 -0.3890963958			
47 -0.52952 -0.3890963958			
$ 48$ $ -0.52952$ $ -0.4008\overline{480231}$			
			-0.4008480231
			-0.4013480612
50 -0.52952 -0.3944512178	50	-0.52952	-0.3944512178

Iterations	Theoretical	Experimental
51	-0.52952	-0.3884351222
52	-0.52952	-0.3884351222
53	-0.52952	-0.3884351222
54	-0.52952	-0.3907311105
55	-0.52952	-0.3980456027
56	-0.52952	-0.3906448114
57	-0.52952	-0.4059641886
58	-0.52952	-0.388795591
59	-0.52952	-0.3965149438
60	-0.52952	-0.4046238846
61	-0.52952	-0.4025665527
62	-0.52952	-0.3938826373
63	-0.52952	-0.3762519993
64	-0.52952	-0.395794006
65	-0.52952	-0.3963359515
66	-0.52952	-0.4041835107
67	-0.52952	-0.4079657663
68	-0.52952	-0.3994302984
69	-0.52952	-0.3969173201
70	-0.52952	-0.4045947654
71	-0.52952	-0.3955795003
72	-0.52952	-0.396464158
73	-0.52952	-0.3954576879
74	-0.52952	-0.3922859202
75	-0.52952	-0.3903440066
76	-0.52952	-0.3997641322
77	-0.52952	-0.3953649947
78	-0.52952	-0.3912133919
79	-0.52952	-0.4025246453
80	-0.52952	-0.4028140875
81	-0.52952	-0.3936148616
82	-0.52952	-0.4070786244
83	-0.52952	-0.3972980299
84	-0.52952	-0.401852009
85	-0.52952	-0.4013366986
86	-0.52952	-0.4078261972
87	-0.52952	-0.3935246526
88	-0.52952	-0.4035197528
89	-0.52952	-0.4009673514
90	-0.52952	-0.4068310896
91	-0.52952	-0.4011399496
92	-0.52952	-0.407983523
93	-0.52952	-0.4008302665
94	-0.52952	-0.4089342389
95	-0.52952	-0.404329474
96	-0.52952	-0.4098480158
97	-0.52952	-0.4068971478
98	-0.52952	-0.3970568892
99	-0.52952	-0.4050681684
100	-0.52952	-0.4176714243
	1 3.3_30_	

Iterations	Theoretical	Experimental
101	-0.52952	-0.4127125846
102	-0.52952	-0.4028407225
103	-0.52952	-0.397501173
104	-0.52952	-0.4120374641
105	-0.52952	-0.4085077118
106	-0.52952	-0.4095649677
107	-0.52952	-0.4097617167
108	-0.52952	-0.4122036682
109	-0.52952	-0.4182641555