Running QUEST: A Determinant QMC Code

'QUEST' is the name of the determinant QMC code we are in the process of developing. There are still some things that we are working on, but we think it is probably better that you learn this new, improved code than one of our older legacy codes which we will gradually be retiring.

This 'hands-on' tutorial will consist of four sections (if we have time):

- 1. Running QUEST for square lattices and checking analytic limits.
- 2. Running QUEST for square lattices and looking at some physics.
- 2P. Running QUEST for square lattices and looking at **more** physics. (See 'Two paths' note below.)
- 3. Learning how to run QUEST for more general geometries.
- 4. Learning how to use 'ADEPT,' a software tool for extracting data from QUEST output files for more easy analysis and plotting.

Two paths: Sections 3,4 start getting into the nitty-gritty of how someone who really wants to use QUEST would be able to modify a geometry file to do his/her specific problem. We are concerned that might not be everyone's goal here. So, after Sec. 2, you can, alternately, move into Sec. 2P and continue looking at some physics with QUEST on the square lattice.

1. Running QUEST for square lattices and checking analytic limits

The friendly and efficient Summer School Support Team will either already have copied all the necessary files into your individual directories or will instruct you now how to do so. (If you had downloaded QUEST yourself you would need to unzip a tar file and make the executables, a 5-10 minute process explained on the QUEST website.) Note that QUEST is bundled with the required lapack and blas libraries so that we do not have any hassles with the compiler figuring out where these libraries are on the user's machine. However, that also means that the libraries might not be most fast for your particular hardware. If you seriously use QUEST you probably should link to optimized blas and lapack routines. (You can link to your own blas/lapack by modifying the file "make.inc" in the root directory of QUEST.)

Once you have the files, change to the QUEST directory:
cd QUEST-1.3.0/

We will begin by doing some runs for (two dimensional) square lattices. To do this, change directory
cd EXAMPLE/2dRecLatt

We will first check the code is correctly computing the U=0 limit. Take a brief look at the input file ueq0.in. You will see that the output file is specified to be ueq0 ('.out' will be appended to this string automatically). The lattice dimensions are $n_x = n_y = 8$, the Hamiltonian parameters are $U = 0$, $t_\uparrow = t_\downarrow = 1$, $\mu_\uparrow = \mu_\downarrow = 0$. QUEST uses the particle-hole symmetric form of the Hubbard Hamiltonian, so $\mu_\uparrow = \mu_\downarrow = 0$ corresponds to half-filling.

Important Note: Despite its name, this code only works for square lattices. Please only use $nx = ny$ when running.

Finally, notice that the inverse temperature $\beta = L\,\Delta\tau = (20)(0.1) = 2$, so that $T = 0.5$. The other input parameter we will utilize is the length of the run. In ueq0.in we have set the number of warm-up (equilibration) sweeps to $nwarm$=100 and the number of measurement sweeps to sweeps to $npass = 500$. We mentioned in describing the algorithm that DQMC solves the $U = 0$ problem exactly and so actually no equilibration or monte carlo averaging is needed. We could make $nwarm$ and $npass$ even smaller.

Let's leave the input $ntry = 0$ for now. It controls the frequency of a type of monte carlo move which is needed for $U \gtrsim 8$. We will likewise ignore all the other inputs which control the binning of the data and permitted size of accumulated round-off errors.

Run the code with the input ueq0.in
./2dRecLatt ueq0.in

This should take about 20 seconds.

Take a look at ueq0.out. At the top, the inputs to the code are echoed back, followed by some statistics on the acceptance/rejection rate of the moves. Then data for the average sign, kinetic energy, occupation, etc are listed. Let's check a few of these.

First, all the error bars are zero. This is, again, because at $U = 0$ DQMC solves the Hamiltonian exactly and hence gets the same answer for all steps in the simulation. The up and down occupations are 0.5000, which is the correct value for the particle-hole symmetric point $\mu_\uparrow = \mu_\downarrow = 0$.

To check the energy and Green's function we will use the analytic solution discussed in lecture, for which we have written a simple fortran code. The code ueq0_analytic.f should already have a compiled version. To access it, you need to change directories up out of the QUEST directory, or, much better, open up another terminal.

Run the code
./ueq0_analytic.e

Use the same inputs as in DQMC: $nx = ny = 8$, $tx = ty = 1$, $\beta = 2$, $\mu = 0$. Note there is a slight difference between the capabilities of these two codes. The analytic code considers a single spin species and allows for the x and y hopping to be different, where QUEST has two spin species and allows the up and down spin hopping to be different. QUEST can easily do different x and y hopping as well, as we shall see when we do use t he general geometry version. Of course, if $U = 0$ the up and down spins decouple, so that difference in the codes is unimportant. The analytic code will immediately give you the density, which should agree with QUEST, and also an energy (-0.747769), which should be half the QUEST kinetic energy (-0.14955385E+01), because QUEST has two spin species.

If you supply a lattice separation to the analytic code, it should give you a result which also agrees with QUEST. for example $(lx, ly) = (3, 0)$ gives $G(lx, ly) = 0.000168$ which compares with the QUEST output $0.16807029E - 03$.

Take a few minutes to modify ueq0.in and check different cases against the analytic solution. Does QUEST work when the chemical potential is nonzero?

Note that QUEST defines the kinetic energy to include all the quadratic pieces in the Hamiltonian, ie the chemical potential term as well as the hopping term. Thus if you run for $nx = ny = 8$, $tx = ty = 1$, $\beta = 2$, $\mu = 0.4$ you will immediately see agreement for the densities (0.573177), but QUEST will give you a kinetic energy -0.19351053E+01 and ueq0_analyic returns -0.738282. To see these agree, add $-\mu$ times the density to the analytic energy and double it (two spin species): 2(-0.738282-0.4*0.573177)= -1.935105.

Bonus question: Can you do a consistency check between the $(lx, ly) = (1, 0)$ Green's function and the kinetic energy? This is actually a way of evading the little calculation we just did to convert from K to $K - \mu N$.

Now let's check $t = 0$. Peak inside the input file teq0.in. You will see it is for a 4x4 lattice (of course the dimension of the lattice has no effect on the physics at $t = 0$) with $U = 4, \mu = 0$, and $\beta = 0.8$.

9) Run the code with the input teq0.in
./2dRecLatt teq0.in

This should take just a few seconds. If you open the output teq0.out you will see densities which are close to the correct half-filling value (0.496 ± 0.002 and 0.504 ± 0.002). Unlike $U = 0$, we now have error bars because for $U \neq 0$ DQMC must do a non-trivial sampling of the Hubbard-Stratonovich fields. The value of $U\langle\, n_\uparrow n_\downarrow \,\rangle = 0.334 \pm 0.002$ can be checked by running the code
./teq0_analytic.e

It should give half-filling exactly and also $\langle\, n_\uparrow n_\downarrow \,\rangle = 0.08399$. Multiplying by $U = 4$ gives a value in agreement with QUEST (0.33596). Notice a first bit of physics coming out: In the absence of interactions $\langle\, n_\uparrow n_\downarrow \,\rangle = \langle\, n_\uparrow \,\rangle \langle\, n_\downarrow \,\rangle = 0.25000$ at half-filling. $U = 4$ has reduced the double occupation by a factor of three at $\beta = 0.8$.

teq0_analytic also computes the local moment $\langle\, m^2 \,\rangle = \langle\, (n_\uparrow - n_\downarrow)^2 \,\rangle = \langle\, n_\uparrow + n_\downarrow - 2n_\uparrow n_\downarrow \,\rangle = 0.83202$. This is of course another measure of the double occupancy. The non-interacting half-filled value is $\langle\, m^2 \,\rangle = 0.5$ while the large U value is $\langle\, m^2 \,\rangle = 1.0$.

A final comment about this first $t = 0$ limit check: the magnetic structure factors $S(q_x, q_y)$ are the sums of the spin-spin correlation functions. The ferromagnetic $(q_x, q_y) = (0, 0)$ and anitferromagnetic $(q_x, q_y) = (\pi, \pi)$ values are printed out by QUEST as "XX Ferro structure factor" and "XX AF structure factor" (and their symmetry equivalent ZZ partners). In the $t = 0$ limit all intersite spin correlations vanish, and so these structure factors can be seen to give the same value as the local moment (to within statistical errors).

You should check QUEST gives the correct $t = 0$ results for non-zero μ. You might ponder how big you need to make μ to shift the density away from half-filling. What is the answer when U and β are large?

One final comment about these checks. It is a good habit not to use 'round' values like $t = 1$ when checking a code, because if your program is simply missing a factor of t somewhere the $t = 1$ check will not catch the bug. So, you should really always verify programs with 'offbeat' input parameters if possible.

Whenever you do a new problem with DQMC, always check the $U = 0$ and $t = 0$ limits. There is a very good chance your code is working if it passes these two tests. The reason is that $U = 0$ ensures your lattice geometry (the kinetic energy term in the Hamiltonian) is right. The $t = 0$ limit ensures you have the interactions correct. So you have checked the entire Hamiltonian. Of course it is always possible there is some sinister bug which only shows up when both t and U are nonzero, but in our experience this is rather rare. A final check against an exact diagonalization calculation on a small lattice will complete your debugging by allowing t and U both non-zero together. Even though almost all bugs are caught without this last check, it should always be done for completeness and also because diagonalization always gives some initial insight into the physics.

2. Running QUEST for square lattices and looking at some physics

We have finished Section One. Are you ready for Section Two? We will now turn to doing some physics with QUEST.

Let's first see QUEST tell us the half-filled Hubbard model has a tendency towards antiferromagnetism. Run QUEST for an 8x8 lattice at $t = 1, U = 4, \mu = 0$ by making a copy of teq0.in or ueq0.in and modifying the input values appropriately. Use $L = 40, \Delta\tau = 0.1$ so that $\beta = 4$. Do $nwarm = 500$ and $npass = 2000$. This will take about 4-5 minutes.

There are different ways to see the antiferromagnetism (AF). One is to notice the AF Structure factors are much larger than the ferromagnetic (F) ones. We get 3.7 ± 0.2 and 3.1 ± 0.3 for the XX and ZZ AF structure factors; 0.12 ± 0.07 and 0.18 ± 0.01 for the XX and ZZ F structure factors, a factor of thirty difference. Note that the error bars are rather large. We did not want us to wait too long for results, but really one would run this simulation for $npass = 10000$ so the error bars would be a factor of two smaller. As a reminder: the Hubbard model is spin-rotation invariant so XX and ZZ correlation functions should be equal to within error bars. (If you like, submit such a run in background and come back to it in a half hour and see what you get.)

You can also see AF in real space rather than momentum space by examining the XX and ZZ spin correlation functions. Moving away from the origin in the x direction these real space spin correlations are:

```
      (0,0)       (1,0)       (2,0)       (3,0)       (4,0)
XX  +0.729(2)   -0.177(4)   +0.052(7)   -0.034(4)   +0.024(4)
ZZ  +0.729(2)   -0.166(4)   +0.036(4)   -0.027(5)   +0.025(7)
```

The $(0, 0)$ value is the local moment, which is 0.500 at $U = 0$ and 1.000 for large U and β when there are neither thermal nor quantum fluctuations. Again, these error bars are a bit too large for comfort. In particular it is hard to see that the XX and ZZ correlations are equal. A longer run will fix that. Note, however, that regarding symmetries a QUEST simulation will suffer the usual problem of a monte carlo as one gets to low temperature on large lattices: the simulation may get stuck in one of the equivalent low temperature minima and take longer than the simulation time to explore the others. We are probably beginning to see this with the the differences between the XX and ZZ spin correlations, which are almost outside of error bars from each other. At low T it is often useful to run 10-20 simulations with different random number seeds to allow the simulation to sample the different possible ordering directions.

It is informative to look at the spatial decay of the Green's function:

```
      (0,0)       (1,0)       (3,0)       (2,1)       (4,1)
 G  +0.500(0)   -0.1688(9)  -0.0000(0)  -0.0369(6)  -0.0174(3)
```

Evidently the values for the up and down Green's functions $G_\uparrow(i, j)$ and $G_\downarrow(j, i)$, whose product give the spin correlator $\langle S_i^+ S_j^- \rangle = -\langle G_\uparrow(i, j) G_\downarrow(j, i) \rangle$, are somehow coordinated, because the average of their product is much larger than one might expect by just multiplying the averages of individual Green's functions. (We print out G only for those separations which are not zero by symmetry. You might want to check by using ueq0_analytic that separations like (2,0) have G exactly zero in the non-interacting limit at half-filling.)

We do not have time 'in class' to study AF carefully. As you may know, when cuprate superconductivity was first discovered and spin mechanisms of pairing were being discussed

there was a big debate about whether the ground state of the two dimensional Hubbard model (and the spin-1/2 Heisenberg model) had long range AF order. You could try to reproduce how this very important issue was settled by Hirsch and Tang using DQMC, and even do a bit better than they did. See PRL 62, 591 (1989). Hirsch and Tang did finite size scaling of the AF structure factor on lattices up to 8x8. What you would do is some longer runs on 8x8 ($npass = 20000$) and at lower $T = 0.1$. These would take about 30 times longer than the simulation you just did, so a few hours. Also 4x4 and 6x6 lattices which would be very fast because of the N^3 scaling (so 6x6 would run in $(6x6/8x8)^3 = 1/5.6$ of the time for 8x8). Running overnight you could go beyond Hirsch and Tang with 10x10 lattices.

The other key element we discussed for the half-filled Hubbard model was the Mott gap. This is much more difficult to get. The reason is that one either has to compute $\rho(\mu)$ for μ nonzero, in which case there is a sign problem, or else analytically continue the Green's function $G(\tau)$ to get the density of states $A(\omega)$. Such analytic continuations are now pretty standard in the community, but are definitely a considerable step more tricky than looking at static properties.

2P. Running QUEST for square lattices and looking at more physics

For those students with a bit less interest in learning about how to run QUEST in general geometry (Hamiltonian) cases, we suggest here a few more exercises you might try which are more 'physics-oriented'. We are a little limited by suggesting computations that will take a QUEST only a few minutes to finish.

Project One: Short range correlation functions tend to be pretty well represented on small lattices (ie they have relatively minor finite size effects). Let's therefore look at the local moment and near neighbor spin correlations on 6x6 lattices for different values of U and T. We will stay at half-filling ($\mu = 0$) and use $\Delta\tau = 0.1$. If you use $nwarm = 1000$ and $npass = 10000$ the runs will not take too long (5-10 minutes for the lowest temperatures). Do a grid of runs for $\beta = 0.5, 1.0, 2.0, 3.0, 4.0, 6.0, 8.0$ and $U = 2, 4, 6, 8$. (For $U = 8$ modify the input file to set $ntry$ to 2.) Make a plot of the local moment (xx spin correlation function at dx=dy=0) as a function of $T = 1/\beta$ for different values of U.

QUEST will start printing messages about the error in updating the Green's function for larger U, β. This reflects the larger numerical errors that are accumulating. Ultimately if we increase U much further we will need to start adjusting some of the inputs to QUEST which control such errors.

You will want to modify the .in file so that QUEST prints different runs to different output files. Then take advantage of the fact that you have multiple cores at your disposal to submit runs in parallel.

You might notice that the local moment has most of its temperature evolution at pretty high T. Why is that? What energy sets the scale for moment formation? If you look very closely at your numbers, you will also see, however, that a little action takes place at lower T, and that, strangely, there is a small upwards jog in the moment for low U and a small downwards jog for large U. A few years ago there was considerable interest in these effects in the optical lattice community, especially because dynamical mean field theory overestimated the downwards jog. It turned out to be important to correct for the Trotter errors in DQMC to get to the bottom of this affair (one of the relatively few cases where the Trotter errors were important for the physics). Google 'Pomeranchuk cooling' to get more into this story.

Using the same data files, plot the near neighbor (dx=1, dy=0) spin correlation function versus T for different U. Does it behave as expected? Has the spin correlation settled down

at the same high T as the moment? What is the important energy scale for near neighbor spin correlations?

In making these plots, you will find yourself cutting and pasting the appropriate numbers out of a bunch of files. If this annoys you enough, go to Sec. 4 and learn how 'ADEPT' can do it for you.

3. Learning how to run QUEST for more general geometries

We have finished Section Two. It is time to learn how to do more general geometries with QUEST. We will do a few examples, using pre-written geometry files. As you can see, the discussion is rather lengthy, but we hope that despite being long, it is all quite simple. The ability to do general geometries was our chief motivation in rewriting our DQMC codes. Prior to this, we had separate codes for each geometry. Then, if an algorithmic improvement was discovered, it needed to be implemented separately in each of dozens of codes. Clearly this is not only inefficient, but also subject to error and challenging to verify that all the codes have been modified correctly. We will return to a second important motivation at the end of the discussion.

The best way to check general geometries is to write $U = 0$ analytic codes. (The $t = 0$ check should also be made, but of course does not require anything new.)

The general geometry version of quest can be run from the EXAMPLES/geom directory as follows: ./ggeom input

The file 'input' tells QUEST which geometry file to use with the line gfile = square.geom. It also contains the geometry independent parameters needed for the simulation: the chemical potential μ, the number of imaginary time slices L and $\Delta\tau$ and thereby $\beta = L\Delta\tau$. Also in 'input' are the length of the run $nwarm$ and $npass$.

The geometry file to which 'input' points contains the lattice structure/symmetries and the associated Hamiltonian parameters.

We will first reproduce our square lattice test. Make sure that 'input' specifies gfile = square.geom. Then go to 'square.geom' and take a look at it. It is a bit complex (because .geom files can do any geometry!). Its full structure is explained in the QUEST manual. We will go through the key points in a moment. For now, simply notice that the lattice is set to 8x8 with

```
#SUPER
8 0
0 8
```

and that the column of values of U are all set to zero in #HAMILT. (We will also discuss precisely how the hoppings are defined.)

Run the code
./ggeom input

Look at the output file and verify that it has the same results that we obtained earlier with the dedicated square lattice code.

Having done a test run of ggeom, let's discuss how to use it in general. We need to explain the important elements of the .geom file. A .geom file begins by specifying the dimension of the lattice, eg for rectangular, triangular, honeycomb:

```
#NDIM
2
```

NDIM would be 3 for cubic, of course, and 1 for a linear chain.

Next the primitive lattice vectors (the Bravais lattice vectors $\vec{a}_i$ of Ashcroft and Mermin) are defined. For a square or rectangular lattice:

```
#PRIM
1.0 0.0
0.0 1.0
```

Note that we have set the lengths $|\vec{a}_i| = 1$. This is conventional in model Hamiltonians where one typically chooses t_x, t_y, and t_z different if one wants to simulate lattices which are anisotropic. One can give a full 3x3 matrix for #PRIM even if NDIM< 3. If you do that, QUEST only reads the upper left NDIM x NDIM block. This is also true of #SUPER.

The next section of the .geom file is #ORB. This is what allows QUEST to do geometries like the honeycomb lattice which require a basis. #ORB has one line for each atom in the basis. The first entry is a string which serves as a label for the atom. This label can be used in various advanced aspects of QUEST, but we will ignore it here. The next three entries in the line are the position of the atom in the unit cell. QUEST automatically assigns a number to each atom (ie to each line in the #ORB section of the .geom file) beginning with zero. The #ORB section of the .geom files we have provided contain a final entry in each line which reminds you of the assigned number. However this is not needed by QUEST. (Ie you can remove it- QUEST does not read this final entry.) Putting these comments together, most of the .geom files we have provided takes this simple form, because there is only one atom in the basis:

```
 #ORB s0  0.0  0.0  0.0   #0
```

Important: We commented above that in #PRIM and #SUPER one can give either provide NDIM dimensional objects or always use three components. QUEST is happy either way. In #ORB above (and #HAMILT and #SYMM described below) QUEST *demands* three dimensional objects. We do not go into the details for why this is in this document. However, this convention turns out to allow more simple implementation of complex geometries (like multilayer heterstructures) even if is slightly annoying to have unnecessary components for simpler (lower dimensional) systems.

Modify rectangle.geom to simulate $nx \neq ny$. Verify that you get agreement with ueq0_analytic.e in the non-interacting limit. As an example, we tried $L = 10, \Delta\tau = 0.125, \mu = 0$ with

```
#SUPER
8 0
0 6
```

Here is the comparison:

```
           QUEST      ueq0_analytic
G(1,0)   0.170087      -0.170087
G(0,1)   0.169681      -0.169681
```

We apologize for the sign convention difference between the two codes. You can see here a bit of the sensitivity of the $U = 0$ check. The Green's functions differ by only a half a percent in the x and y directions, but because the $U = 0$ problem is solved exactly in DQMC (zero error bars) one can precisely reproduce even such small effects. What this means, of course, is that even a small error in the code is likely to be revealed at $U = 0$.

You might also try different hoppings. To do this, go to the part of the .geom file that reads #HAMILT. This block of the code defines the hoppings, on-site energies, and interaction strengths. The convention for the lines in #HAMILT is the following: Each line begins with two entries which are the (automatically assigned) atom numbers from #ORB. For problems without a basis these will just be '0 0'. To include a hopping, the next three entries in the line should be the direction to the neighboring site. QUEST automatically makes $\hat{H}$ Hermitian, so each pair of connected sites requires only one line. The next two entries are the hopping values for the up and down electrons, which are allowed to be different in QUEST. The final entry is U, which should be set to zero for lines defining hopping.

To include a site energy, the neighboring site (entries three to five) should be '0.0 0.0 0.0'. On-site energies are entered in the next entries (six and seven). Again, they can be different for up and down. The eighth entry for U is left zero.

To include a U value, use '0.0 0.0 0.0' for the neighboring site inputs and insert a value in the last (eighth) entry.

If you have a geometry with a basis (more than one line in #) the different atoms can be assigned different site energies and interactions.

Probably an example will help. Consider this #HAMILT in rectangle.geom

```
#HAMILT              tup   tdn    U
0 0   1.0   0.0   0.0   1.0   1.0   0.0
0 0   0.0   1.0   0.0   2.2   2.2   0.0
0 0   0.0   0.0   0.0   0.5   0.5   4.0
```

The first two lines are the hopping in the x and y directions. The third line is an on-site energy 0.5, and is the same for the up and down species. (The sign convention for the site energies is $+\epsilon\, n_i$ so that 0.5 in #HAMILT corresponds to -0.5 in the chemical potential. The third line also sets $U = 4$. (If you like you can also separate these into two distinct lines.) We again note that in #HAMILT it is required to supply a three component vector as the pointer to the site, even if #NDIM is two.

Note: The chemical potential defined in input is a *global* chemical potential which applies to all sites in the lattice. The on-site energies in #HAMILT allow for different chemical potentials on different atoms. Thus there is a slight redundancy in the code: Rather than having a global chemical potential one could shift all the site energies. If you don't specify chemical potentials in input, the they are set, by default, equal to zero.

Running with the rectangle geometry on an 8x8 lattice with $U = 0, \mu = -0.5, L = 10, \Delta\tau = 0.125, t_x = 1, t_y = 2.2$, We obtain

```
            QUEST      ueq0_analytic
G(1,0)    0.082363      -0.082363
G(0,1)    0.275000      -0.275000
```

Here we use the term rectangular because, although the supercell is square the hoppings $t_x \neq t_y$.

There is some redundancy in the way QUEST knows the geometry is rectangular. If the defined supercell in #SUPER has different diagonal entries, then QUEST knows not to assume the x and y directions are equivalent. Likewise, a different entry for t_x and t_y will automatically be flagged by QUEST. (Finally, the strange string label in #ORB about which we have been so silent can also be used to distinguish different types of atoms and can break symmetries. If you had a three band Cu-O model of the cuprates you could, for example, make the oxygen atoms along the x and y bonds have distinct labels and hence different entries in #HAMILT.)

But more fundamentally, QUEST knows about the symmetries of the lattice from the #SYMM block in the .geom file, which is the last one we shall discuss. In square.geom this looks like,

```
#SYMM
d  0.0 0.0 0.0 1.0 0.0 0.0
d  0.0 0.0 0.0 0.0 1.0 0.0
c4 0.0 0.0 0.0 0.0 0.0 1.0
```

QUEST supports three types of symmetry definitions. 'cn' where 'n' is an integer, tells QUEST the lattice is symmetric under rotations by $2\pi/n$. The six numbers following 'cn' specify the three cartesian coordinates of a point belonging to the axis, followed by the axis direction in cartesian coordinates. In this case 'c4' is $\pi/2$ and indicates the x and y directions are equivalent. The point belonging to the axis is the origin '0.0 0.0 0.0' and the axis is the z direction '0.0 0.0 1.0'.

The symmetry 'd' is a mirror plane symmetry. It too is followed by six numbers. The first three are the cartesian coordinates of a point in the plane, and the final three are the cartesian coordinates of the normal to the plane.

Finally, 'i' is used for inversion symmetry. It is followed by three numbers, the cartesian coordinates of the inversion point.

In rectangular.geom

```
#SYMM
d  0.0 0.0 0.0 1.0 0.0 0.0
d  0.0 0.0 0.0 0.0 1.0 0.0
```

'c4' has been removed to tell QUEST rotation by $\pi/2$ is not a symmetry. But, as mentioned earlier, the difference between t_x and t_y in HAMILT will already have alerted QUEST to this. We could leave 'c4' in and QUEST will ignore it.

In specifying #SYMM you must list all three components of the vectors even if the lattice is two dimensional.

In running the square and rectangular geometries, you may have noticed that more Green's function values were printed for rectangular, ie G(0,1) and $G(1,0)$ are both returned, whereas for square only $G(1,0)$ is listed. This is, of course, because for the square geometry the two values $G(0,1)$ and $G(1,0)$ are equal, and QUEST has automatically averaged them together. In fact, this is the second big advantage of QUEST, along with allowing for implementing code improvements more systematically mentioned at the beginning of the tutorial. It is very important in Monte Carlo to combine symmetry related measurements because it can vastly reduce your error bars. With our older codes it was relatively easy to adapt the matrix K for different geometries so that the Monte Carlo updates could proceed, but it was very

time consuming to code up the different averaging symmetries in the measurement routines each time a new geometry was considered. QUEST now handles all that automatically.

As final examples we will (quickly now) discuss the triangular and honeycomb geometries. These illustrate a few of the ideas discussed above which were trivial in the rectangular, square, cubic, and chain cases.

The triangular lattice has a non-trivial (ie non-diagonal) #PRIM:

```
#PRIM
1.0 0.0
0.5 0.8660254038
```

These vectors are easily recognized as one choice for the primitive vectors of a triangular lattice.

Of course #SYMM must be appropriate. The triangular it now reads

```
#SYMM
c6 0.0  0.0  0.0   0.0 0.0 1.0
```

indicating that there is a rotation by $2\pi/6$ symmetry (c6) about the origin 0.0 0.0 0.0.

In a test run we compared QUEST with a $U = 0$ code 'ueq0tri' for triangular lattices. For a 6x6 cluster with $t = 1, \beta = 1.25, \mu = 0$:

```
           QUEST       ueq0tri
G(0,0)   0.560919      0.560919
G(0,1)  -0.148547     -0.148547
G(0,2)   0.031329      0.031329
```

As a challenge, you might want to write the $U = 0$ code, patterned after the one provided for a rectangular lattice.

Notice that $G(0, 0)$ is not equal to $1/2$, even though $\mu = 0$. This is because the triangular lattice is not particle-hole symmetric.

Let's finish finally with honeycomb.geom, an example with a non-trivial #ORB.

```
s0 -0.57735027  1.0  0.0d0 #0
s0  0.57735027  1.0  0.0d0 #1
```

Recall that QUEST automatically labels the two atoms in the basis 0 and 1, and that we have reminded ourselves of that with the #0 and #1 at the ends of the lines. Can you see how if you put the locations of the atoms defined above together with the separations and labels in #HAMILT below that you will indeed get a honeycomb lattice?

```
#HAMILT                    tup  tdn  U
0 1  1.15470054  0.0 0.0   1.0  1.0 0.0
1 0  0.57735027  1.0 0.0   1.0  1.0 0.0
0 1 -0.57735027  1.0 0.0   1.0  1.0 0.0
0 0  0.00000000  0.0 0.0   0.0  0.0 0.0
1 1  0.00000000  0.0 0.0   0.0  0.0 0.0
```

Notice there are three hoppings in #HAMILT, and two separate lines (4 and 5) with which we could, if desired, assign different U values on atoms 0 and 1. We could also add two more lines if we wanted different site energies, like 0.3 and 0.7:

```
0 0   0.00000000   0.0 0.0   0.3   0.7 0.0
1 1   0.00000000   0.0 0.0   0.3   0.7 0.0
```

We have not written a $U = 0$ check for the honeycomb lattice, but we did check it works against a "legacy" DQMC code.

You will notice that the .geom files are of different lengths. Some include other items like #PHASE, #BONDS, #PAIR. We will not discuss these here. They are used to measure more extended objects, like d-wave pairing correlations where the operator being measured itself has some spatial structure which needs to be specified.

4. Learning how to use 'ADEPT'

We have finished Section Three. Our final topic is to learn to use the 'Automatic Data Extraction and Plotting Tool' ADEPT, the data analysis tool. In your top directory there is a file ADEPT.zip which includes the software and a full description (ADEPT_usersguide.pdf). There are also four youtube videos:

youtube.com/user/questadept

Journey into the ADEPT directory with cd ADEPT

Looking the Users Guide and videos over, we think they explain things better than a lecture could. If you end up using QUEST extensively, you should probably learn ADEPT. As the Users Guide explains:

1.1 About: The Automatic Data Extraction and Plotting Tool (ADEPT) is a Java application which is used to help in the analysis of data output from QUEST simulations [1]. Output files from QUEST - containing normal user-readable data - are parsed into variables in an XML-like structure which allow ADEPT to access the data in a more organized way.When the files are passed into ADEPT, it opens a Graphical User Interface (GUI) allowing the user to import, manipulate, and save data from numerous QUEST output files into a format which can be easily plotted using the user's choice of plotting software.

1.2 Motivation: ADEPT was created based on a need which arises from comparing and plotting data from multiple QUEST simulations. Previously, the required data would have to be manually extracted by the user either with a copy-paste approach or using a parsing script, each of which had difficulties associated with them.

• The copy-paste approach is obviously very labor-intensive when the user wishes to compare data from a large number of files.

• Although parsing scripts do allow for extraction of data, this approach still requires significant work from the user if any manipulation of this data is needed.

Another need that arose during development of the application was the ability to start with a large repository of data files and only compare files with certain values for certain data elements. For example, one may only want data from simulations where the interaction strenght U is equal to 4. In this case, the parsing approach would be of no help, unless the data files were themselves named using some strict labeling convention.

SOME FURTHER NOTES:

• Our data on checking 2dRecLatt for nonzero μ was obtained running on a 6x6 lattice for $\mu_\uparrow = 0.4$, $t_\uparrow = 1.0$, $\mu_\downarrow = -0.7$, $t_\downarrow = 1.5$, with $L = 12, \Delta\tau = 0.1$ (so that $\beta = 1.2$. We wanted to try different parameters for the up and down electrons to make sure the code got that right, in addition to checking nonzero chemical potential.

	QUEST	ueq0_analytic
Up spin occupancy	0.563006	0.563006
Up spin G(1,0)	-0.165515	-0.165515
Down spin occupancy	0.415216	0.415216
Down spin G(1,0)	-0.179867	-0.179867

• We checked the half-filled $U = 4$, 8x8 lattice with $nwarm = 1000$ and $npass = 10000$, as suggested in the notes. The results are:

	(0,0)	(1,0)	(2,0)	(3,0)	(4,0)	
XX	+0.729(2)	-0.177(4)	+0.052(7)	-0.034(4)	+0.024(4)	SHORT
ZZ	+0.729(2)	-0.166(4)	+0.036(4)	-0.027(5)	+0.025(7)	SHORT
XX	+0.729(2)	-0.172(2)	+0.040(2)	-0.022(1)	+0.019(1)	LONG
ZZ	+0.729(2)	-0.174(2)	+0.043(2)	-0.026(2)	+0.022(2)	LONG

The XX and ZZ AF structure factors, which were 3.7 ± 0.2 and 3.1 ± 0.3 are now 3.09 ± 0.05 and 3.32 ± 0.10.

• One can check chain.geom against ueq0_analytic.e by setting t_x or t_y to zero. For a 12 site chain with unit hopping we get these comparisons at $\beta = 1.25$ and different μ:

	G(0)		G(1)	
mu	QUEST	ueq0_analytic	QUEST	ueq0_analytic
0.0	0.500000	0.500000	-0.23153	-0.23153
1.0	0.324093	0.324093	-0.19667	-0.19667
2.0	0.166875	0.166875	-0.11660	-0.11660

To insert the chemical potential one can either modify 'input' or one can leave $\mu = 0$ in input and insert an extra line in #HAMILT of chain.geom:

```
#HAMILT            tup  tdn   U
0 0  1.0 0.0 0.0  1.0  1.0  0.0
0 0  0.0 0.0 0.0 -2.0 -2.0  0.0
0 0  0.0 0.0 0.0  0.0  0.0  0.0
```

Recall that separation $(0.0, 0.0, 0.0)$ in the third through fifth entries of a line in #HAMILT are used as a chemical potential if the non-zero entries are in the hopping (one body) arguments.

• Note to self: Verified that honeycomb.geom (almost) agreed with legacy code.

	Energy	
	QUEST	legacy
6x6	-1.23240	-1.22545
9x9	-1.23238	-1.23219

It is likely QUEST is doing a slightly different cluster/boundary conditions than the legacy code. Need to figure out how to set #SUPER (probably nondiagonal) to do same cluster as being done in legacy code.

- My results for Project One of Sec. 2P:

```
Local Moment
  beta      U=2      U=4      U=6      U=8
  0.2     0.547    0.593    0.638    0.680
  0.5     0.593    0.680    0.757    0.823
  1.0     0.614    0.722    0.815    0.885
  2.0     0.614    0.729    0.832    0.899
  3.0     0.612    0.729    0.833    0.895
  4.0     0.614    0.735    0.835    0.892
  6.0     0.618    0.746    0.832    0.886
  8.0     0.624    0.753    0.832    0.884

NN Spin correlator
  beta      U=2       U=4       U=6      U=8
  0.2    -0.0049   -0.0052   -0.0054  -0.0056
  0.5    -0.0256   -0.0280   -0.0295  -0.0309
  1.0    -0.0603   -0.0707   -0.081   -0.0840
  2.0    -0.0891   -0.1216   -0.150   -0.163
  3.0    -0.0989   -0.148    -0.216   -0.248
  4.0    -0.1032   -0.176    -0.258   -0.283
  6.0    -0.1125   -0.25     -0.28    -0.34
  8.0    -0.124    -0.24     -0.30    -0.33
```

Acknowledgements: Richard Scalettar gratefully acknowledges the help of Zhaojun Bai, Roger Lee, Andres Tomas, Chia-Chen Chang, and Simone Chiesa. The development of QUEST was supported by DOE-DE-FC0206ER25793 and NSF PHY 1005503.