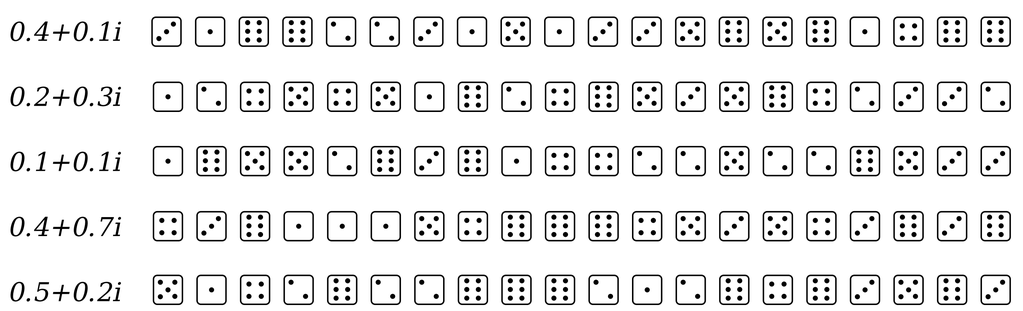
Matrix product states: advanced quantum modeling technique

Gregory Crosswhite, Department of Physics; Dave Bacon, Department of C.S.&E.

Problem

Quantum systems are very difficult to simulate due to the presence of entanglement, which prevents you from studying a system in terms of the composition of its parts. For example, a set of “classical” dice can be understood in terms of the probability distribution of each individual die, whereas in a set of “quantum” dice each separate roll of all dice has its own probability. Thus, to model quantum systems you typically need an amount of information that grows exponentially with the number of dice. The solution is to adopt a clever representation that can adequately model the entanglement properties of the system while still being of manageable size. This poster describes a solution I am investigating called “matrix product states”, as well as its higher dimensional generalization called “projected-entangled pair states”.



Part of the representation of a possible state of twenty quantum dice. Shown are the complex amplitudes associated with 5 of the 3,656,158,440,062,976 possible configurations of the system. (Probability is the absolute square of the amplitude.)

Matrix Product States (One Dimension)

Matrix product states are a good way of representing systems which have interactions in only one dimension. These can be thought of as the set of states which can be represented by a diagram of the form of the right. Each of the four sets of edges (S , S , S , and S) characterize a die at one of the four sites in the system. Entanglement is captured by the vertices, which allow each die to “communicate” with its neighbors and thus attain some correlations. The number of vertices may be increased arbitrarily to allow an arbitrary amount of entanglement to be modelled in the system.

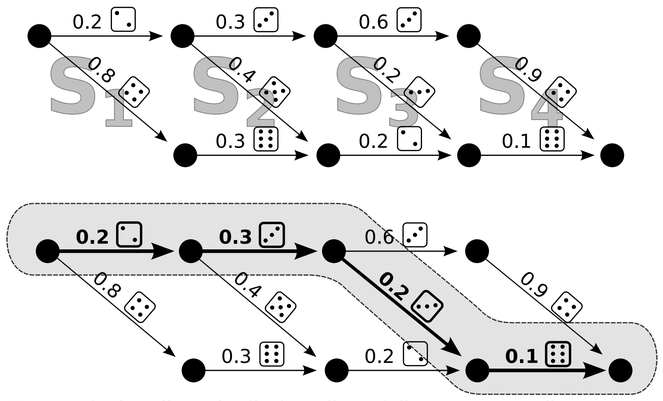
Each possible roll of the four dice is represented by a “walk” through the diagram; the amplitude of the roll is given by the product of the weights on the edges along the walk. For example, as shown on the right, the probability of rolling 2, 3, 3, 6, is (0.2)(0.3)(0.2)(0.1)=0.0012. This diagram was made sparse to keep it simple; in general each edge will have a complex-valued linear combination of many or all of the six dice, and so there may be many walks that include a given roll. In this case, the final amplitude of the roll is given by the sum of the amplitudes of all possible walks giving the roll.

The mathematical representation of these diagrams is given by the equation,

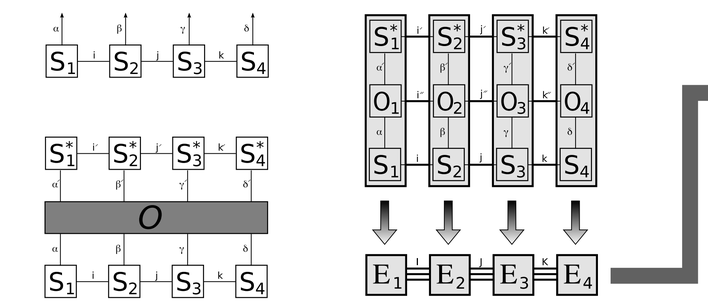
$$\psi^{\alpha\beta\gamma\delta} = \sum_{ijk} (S_1)_i^{\alpha} (S_2)_{ij}^{\beta} (S_3)_{jk}^{\gamma} (S_4)_k^{\delta}$$

where the left-hand-side denotes the amplitude of a given roll, each of the Greek indices correspond to the number rolled in one of the dice, and the Roman indices label one of the vertices in the above diagrams. This equation denotes a tensor structure which is illustrated in the right diagrams, with the boxes denoting tensors, edges denoting indices, and connected edges denoting edges summed over.

One is often interested in the calculating expectation value of the system with respect to some operator, which is illustrated in the lower-left diagram. If the operator can itself be factored into matrix product form, then the expectation value is given by a product of "E" tensors, as shown in the right diagram.

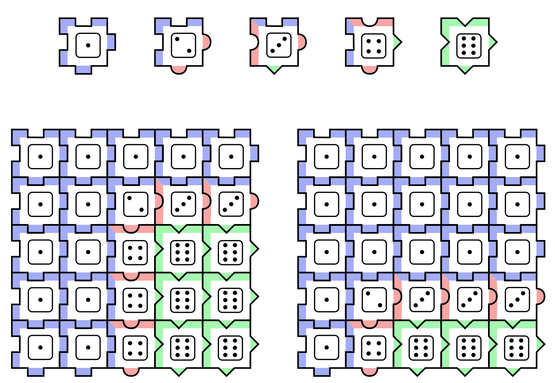


One particular allowed roll of our line of dice, corresponding to a "walk" through the above diagram.

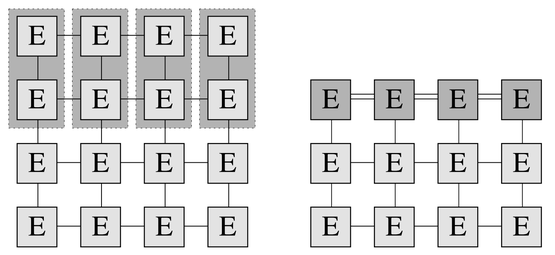


Tensor network diagram representations of a matrix product state (top-left), and the expectation value of the state with respect to an operator (bottom-left, right)

Projected Entangled-Pair States (2+ Dimensions)



Two allowed rolls of our grid of dice, assembled from the pieces above.



Diagrams representing the tensor network one needs to contract to compute expectations for a projected entangled-pair state.

For systems with interactions in two or higher dimensions, it is useful to have a form of representation that allows for “communication” in multiple directions. Projected entangled-pair states are the natural generalization of matrix product states that allow this to happen. It is possible to draw diagrams that are the analogue of matrix product state diagrams, with the edges become multi-edges that connect four vertices; this, however, produces a diagram which is too dense to be illuminating.

If the system is translationally invariant, a more useful visualization is a set of “puzzle” pieces. The different shapes/ colors correspond to the vertices in the matrix product diagrams, and can be thought of as “signals” that each site receives from above and the left and forwards down and to the right. Each piece will in general have a complex weight, but these have been omitted to save space.

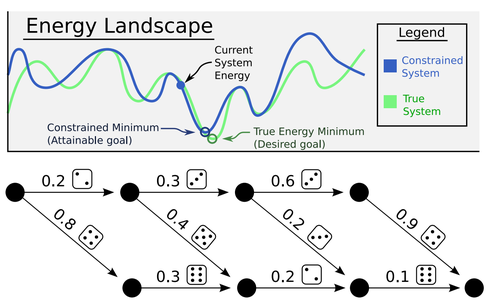
The weight of a particular roll of our “grid” of dice corresponds to the product of the weights on the pieces needed to assemble it. In this particular case, note that the only combinations with non-zero weights are those which allow at most one two to be rolled in the grid. As before, if there are multiple assemblages which obtain the same roll then the amplitude is the sum of the weight of each.

Note the power of this representation: even with only five pieces, we are able to capture many possible combinations of dice. It is this power which makes the representation useful for modeling entanglement in two dimensional systems.

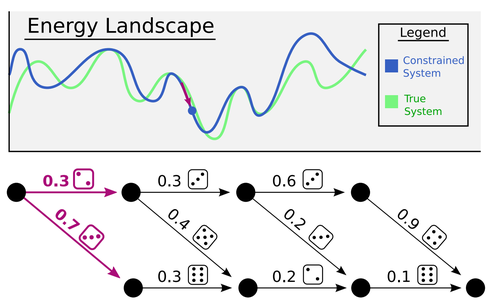
Unfortunately, this power comes at a cost. For this representation to be useful, it has to allow us to calculate expectation values with respect to operators. As in the one-dimensional case, if an operator can be expressed as a projected entangled-pair operator (which is usually the case) then calculating its expectation reduces into contracting (i.e., summing) a network of tensors. However, in the case of two dimensions this network is a grid, and performing the contraction is very expensive. As illustrated in the right of the two diagrams, whenever two rows are contracted, the number of indices on the tensors multiplies. Thus, our tensors grow exponentially large with the number of contractions. Fortunately, there is a compression technique that allows one to “shrink” the tensors to eliminate an index, but the cost of this is a loss of precision. It is currently an open question as to how far we can take this technique.

Variational Algorithm

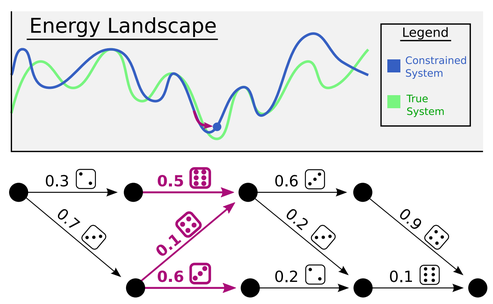
The ability to construct efficient representations of quantum systems is useless unless we can make them be good models of actual physical systems. Fortunately, in quantum physics we are typically most interested in the ground, or lowest energy state of the system. Finding the representation of this state is simply an optimization problem: we search for the energy minimizer in our constrained set of states, and hope that what we find is close to the true minimum. Fortunately, we can generally make the constrained minimum arbitrarily close to the true minimum by increasing the number of vertices (1D) / colors (2D) in our states.



At the start of the algorthim, the system has been initialized to some random matrix product state, with an energy that is neither the true minimum nor even that of the matrix product constrained system.



On the first minimization step, all the sites except site 1 are fixed. The edges at site 1 are then changed to lower the energy as much as possible, bringing the system closer to the constrained minimum.



On the next step, all the sites except site 2 are fixed, and then the edges at site 2 are changed to minimize the energy. The algorithm continues in this fashion, repeatedly sweeping through the sites from left to right and back, until convergence.

Two Possible (but mutually exclusive) Fruits

Bootstrapping to a Quantum Computer

Although these simulation techniques are not likely to work for all systems, they should give us enough information about quantum systems to help us build a robust, first-generation quantum computer. There is promise that this computer will give us even greater power in modelling quantum mechanical systems, due to its innate capacity to handle entanglement; if so, we could use this computer to help us build a more sophisticated second-generation quantum computer. Regardless, it should definitely give us greater power to solve other useful problems, as shown by Shor's Factoring algorithm and the

Quantum Computer Proven Unnecessary

It is possible, though unlikely, that classical techniques such as matrix product states will turn out to be so effective as to allow efficient classical simulation of a quantum computer. It would be very exciting if this were the case, because it would mean that there is nothing a quantum computer could do that a classical computer cannot -- for example, it would lead to an efficient classical factoring algorithm. So in short, by studying quantum mechanics we will have learned a great deal about classical computation.