apparent from the high quality of the fits in the following sections.

3.4.4 Parallelization

A more technical part of this work was the parallelization of the single-band version of the QMC code. This effort was necessary in order to use all available computational resources efficiently and to reach the high accuracy necessary for computing the first-order metal-insulator phase transition line. We chose to implement only coarse-grained parallelism where the same impurity model is (approximately) solved independently in parallel processes. The self-consistency conditions (including the lattice Dyson equation) are evaluated by a master process after the Green functions obtained in each solution have been averaged. While this approach practically limits the number of processors that can be efficiently used to about 16 or 32 (otherwise the relative cost for warm-ups is too high), it has the advantage that the communication overhead is negligible and that its implementation using the Message Passing Interface (MPI) is straightforward. Note that it is a nontrivial task to ensure independent random-number streams for the parallel runs. We used the Scalable Parallel Random Number Generator (SPRNG) library, version 1.0 (Ceperley, Mascagni, Mitas, Saied, and Srinivasan, 1998) for this purpose.

In order for the new code to be executable in serial environments without retaining separate serial versions, we instrumented the program with compiler switches so that both MPI extensions and the random number generator (as well as the maximum value of Λ) can be chosen at compile time. While the code originally already ran on a range of architectures (Cray T90, IBM AIX, Fujitsu VPP), manual setup and knowledge of the suitable compiler options was required for each new compilation. In order not to further complicate matters with the inclusion of new architectures (Intel Linux, Cray T3E) in serial or parallel environments, we created makefiles and an autodetection script. This setup toolbox automatically executes the necessary compilers with appropriate options and includes libraries depending on the detected architecture. Furthermore, the whole code (including tools and required precompiled libraries) can be packed automatically into a single self-extracting file; this feature simplifies code distribution significantly. All of these improvements were later integrated into the multi-band versions of the QMC code by Held and Keller.

Ironically, the most demanding QMC calculations (for $\Lambda \gtrsim 300$) were run in serial mode. The reason is that problems of this size can only be treated efficiently on vector supercomputers with very fast memory access. In principle, the VPP as a vector parallel computer with clock speed memory access would have been ideal for the purpose. Despite an investment of some hundred hours of development time, however, we could not get the MPI version of our program to run on this machine. As the same code worked on all other architectures that we had tested it on, we suspect a bug in the MPI implementation for the VPP.

energy functional. Therefore, the iteration process cannot stick in unphysical regions; it always drifts towards a valid solution. Numerical problems in this respect can only arise near points in the extended T, U, Δ phase space where new minima of the free energy functional appear (as a functional of the hybridization function Δ).