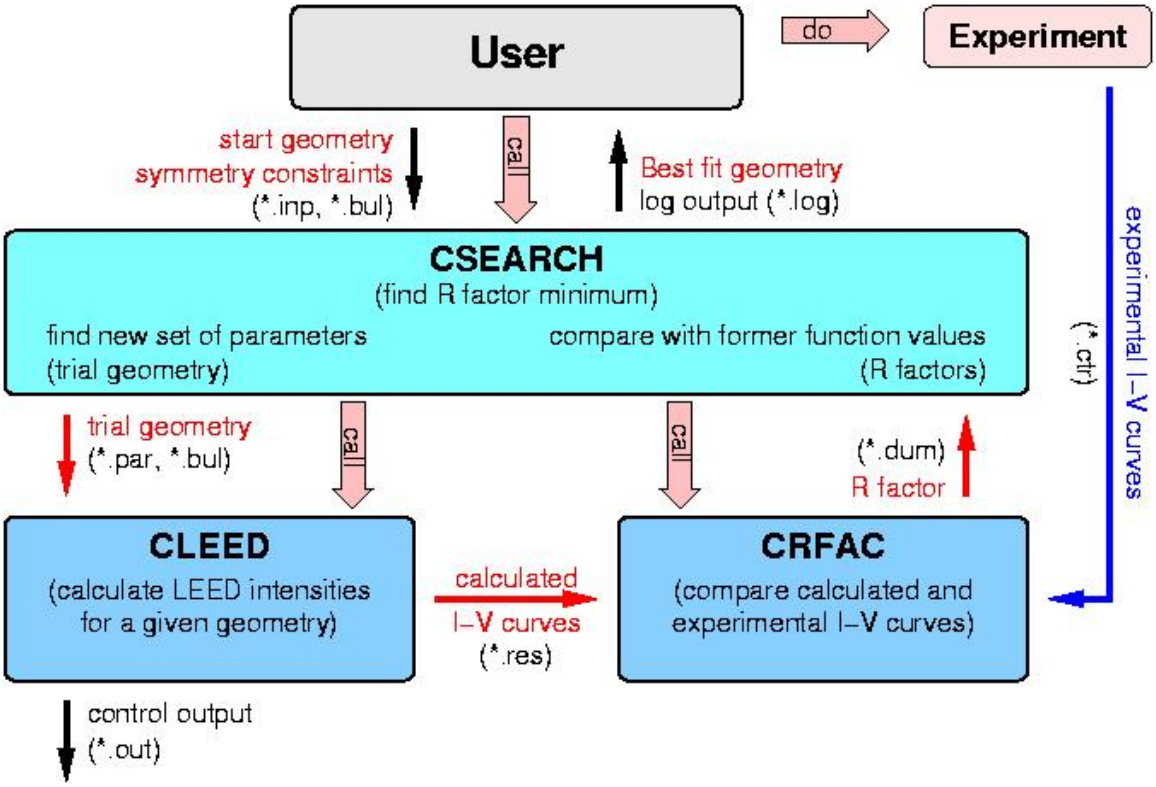
CLEED: Automated Surface Structure determination from LEED-IV curves.

In order to obtain a copy of the CLEED program package, please contact G. Held
g.held@reading.ac.uk

Program Description:
The development of the LEED-IV program package "CLEED" was started by G. Held in 1995, when he was PDRA in Prof. D. A. King's group at the University of Cambridge (hence 'C'-LEED [1]). Further development and improvement of the code took place at the Universities of Würzburg and Erlangen (Germany) and is currently part of the LEED activities in the Held group. Unlike other LEED programs currently in use, this code was written entirely in the C programming language (another reason for calling it 'C'-LEED) which provides a very natural interface with the UNIX operating system, used in the majority of medium sized workstations. The package consists of a master search program calling the actual LEED program and the R factor program as part of a function evaluation routine.
The R factor (function value) is optimized through variations of the geometrical parameters and angles of incidence (function variables) in accordance with certain user-defined symmetry constraints. Each of the three tasks, search, R factor evaluation, and LEED calculation, is performed by separate programs which communicate with each other through the operating system (through files and program calls, see Figure).



The search program uses standard optimization algorithms such as the downhill simplex method [2,3] Powell's method [2,4] simulated annealing, [2] and the genetic algorithm [5,6] which can be selected by the user in order to perform the search for the best fit geometry. While the first two algorithms are strictly downhill-oriented, i.e., will find only the nearest local R factor minimum, the latter two algorithms should in principle provide a means of locating the global R factor minimum within the given constraints, at the expense of many more search steps.
Within each search step a set of geometrical parameters and angles of incidence is chosen by the algorithm depending on the R factor values achieved before and in accordance with user-specified symmetry constraints. The parameters are written to a file serving as input for the LEED

program whose output is then fed into the R factor program in order to calculate an R factor value for this parameter set.

The R factor program offers the choice between four different R factors that can be used for the search: R1,R2 [7], RP [8] and RB [9]. The output R factor value is the optimum achieved by shifting the energy axes of experimental and theoretical I –V curves with respect to each other. This shift acts as a correction for any nonoptimum value of the optical potential in the LEED calculations, which need therefore not be optimized by the search program. This way, one dimension is eliminated from the search parameter space on which the search program operates, hence reducing the number of LEED calculations to be performed. The assignment of experimental and theoretical I-V curves is performed by the user prior to the search.

In our LEED code fully dynamical scattering theory has been implemented along the lines of the algorithms described by Pendry (layer doubling for multiple scattering be-tween sucessive layers of atoms [10]) and Van Hove/Tong (combined space method for layers with more than one atom per unit cell [11]). The extensive use of dynamical memory allocation—an intrinsic feature of the C programming language—allows the memory requirements, even for large surface unit cells, to be kept small. It also allows a very flexible input format which does not impose any restrictions on the number of bulk layers and overlayers, nor on the number of atoms therein (other than the physically available memory). The input from the optimization program to the LEED program is simply a set of atomic coordinates from which the program creates its own set of Bravais and/or composite layers. For these layers scattering matrices are calculated and used to evaluate the amplitudes for multiple scattering between the layers.

The main emphasis in the development of the code has been put on the use of symmetries in order to reduce the time needed for the giant matrix inversion which is part of the combined space method. On average a reduction by a factor of 2 could be achieved. The standard version of the codes does not make use of symmetries among beams, however a symmetrised version is also available. It does not include any linear approximations such as tensor LEED [12,13] which are in general incompatible with global search strategies such as simulated annealing or the genetic algorithm, and can lead to errors even in downhill-oriented optimizations when the search path enters regions of the parameter space where the approximation is inaccurate. It is, however, planned to implement this as an option for locally confined searches in the future version.

Recent improvements include the implementation of the “Molecular T-matrix algorithm” [14-16] and the ability to analyse data sets taken at different angles of incidence.

References:

[1] G. Held, M. P. Bessent, S. Titmuss, and D. A. King, J. Chem. Phys. 105 (1996), 22.
[2] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, “Numerical Recipes in C” Cambridge University Press, Cambridge, UK, 1988.
[3] J. A. Nedler and R. Mead, Comput. J. 7, 308 ~1965!.
[4] R. P. Brent, “Algorithms for Minimization without Derivatives” Prentice Hall.
[5] J. H. Holland, “Adaption in Natural and Artificial Systems” University of Michigan Press, 1975.
[6] D. E. Goldberg, “Genetic Algorithms in Search, Optimization and Machine Learning” Addison–Wesley, 1989.
[7] M. A. Van Hove, W. H. Weinberg, C.-M. Chan, “Low-Energy Electron Diffraction”, Springer, Berlin, Heidelberg, New York, 1986.
[8] J. B. Pendry, J. Phys. C 13, (1980) 937.
[9] G. Held, H. Pfnür, and D. Menzel, Surf. Sci. 271 (1992) 21.
[10] J. B. Pendry, “Low Energy Electron Diffraction” Academic Press, London,1974.
[11] M. A. Van Hove and S. Y. Tong, “Surface Crystallography by LEED”, Springer, Berlin, 1979.
[12] P. J. Rous and J. B. Pendry, Surf. Sci. 219 (1989) 355.
[13] P. J. Rous, M. A. Van Hove, and G. A. Somorjai, Surf. Sci. 226 (1990) 15.
[14] M. Blanco-Rey, P. de Andres, G. Held, D.A. King, Computer Physics Comm. 161 (2004) 159.
[15] M. Blanco-Rey, P. de Andres, G. Held, D.A. King, Computer Physics Comm. 161 (2004) 166.
[16] M. Blanco-Rey, P. de Andres, G. Held, D.A. King, Surface Science 579 (2005) 89.