Computer Physics Communications 134 (2001) 392–425

Computer Physics Communications

www.elsevier.nl/locate/cpc

Fast LEED intensity calculations for surface crystallography using Tensor LEED

V. Blum *, K. Heinz

Lehrstuhl für Festkörperphysik, Universität Erlangen-Nürnberg, Staudtstr. 7, D-91058 Erlangen, Germany

Received 16 May 2000

Abstract

The quantitative analysis of intensity spectra from low energy electron diffraction is today's most widely used technique for the extraction of detailed surface crystallographic information. The Erlangen Tensor LEED package TensErLEED provides an efficient computer code for the fast computation of LEED intensity spectra from virtually any periodic surface. For the full dynamic reference calculation, standard methods such as the muffin-tin approach and the layer stacking method are used. Amplitude changes in Tensor LEED are accessible for geometric, vibrational and chemical displacements from the reference structure. The package also contains a structural search algorithm designed for the retrieval of the global R-factor minimum between calculated and measured intensity spectra within a given portion of the parameter space using Tensor LEED.

PACS: 07.05.Tp; 61.14.Hg; 68.35.Bs; 68.35.Dv

Keywords: Electron–solid diffraction; Low energy electron diffraction (LEED); Structure optimization; Surface crystallography; Surface reconstruction; Surface relaxation; Surface structure; Surface segregation; Tensor LEED (TLEED)

PROGRAM SUMMARY

Title of program: TensErLEED

Catalogue identifier: ADNI

Program Summary URL: http://cpc.cs.qub.ac.uk/summaries/ADNI

Program obtainable from: CPC Program Library, Queen's University of Belfast, N. Ireland

Computers for which the program has been designed and others on which it has been tested: *Computers:* Digital/Compaq Alpha Workstations PWS433au, XP1000; Pentium PC

Installation: Lehrstuhl für Festkörperphysik, Universität Erlangen-Nürnberg, D-91058 Erlangen

Operating systems under which the program has been tested: Compaq Tru64 Unix, Linux

Programming language used: FORTRAN 77

Memory required to execute with typical data: 10–100 Mwords

No. of bits in a word: 8

No. of processors used: 1

* Corresponding author.
E-mail address: v.blum@fkp.physik.uni-erlangen.de (V. Blum).

0010-4655/01/$ – see front matter
PII: S0010-4655(00)00209-5

Has the code been vectorized or parallelized?: No

No. of bytes in distributed data, including test data, etc.: 201 068

Distribution format: tar gzip file

Keywords: Electron–solid diffraction, low energy electron diffraction (LEED), structure optimization, surface crystallography, surface reconstruction, surface relaxation, surface structure, surface segregation, Tensor LEED (TLEED)

Nature of physical problem
The quantitative analysis of low energy electron diffraction (LEED) intensity vs. energy $I(E)$ spectra is an important tool to obtain surface crystallographic information [1–4]. Direct methods to extract such information from LEED data work in special cases only. One generally resorts to a trial-and-error technique, comparing calculated $I(E)$-curves for different surface geometries to spectra measured from a real surface in order to retrieve the correct structural parameters by way of a *best fit*. Since experimental techniques are continuously refined and the complexity of studied systems grows dramatically, a fast means to calculate $I(E)$ spectra for many different surface structures, compare them to experimental data and identify the *best fit* is a prime necessity of the field. The TensErLEED computer code aims to satisfy this need. Examples studied with previous versions of this code include open metal surfaces [5], reconstructed alloy surfaces [6], and complex reconstructions of thin films [7–9] and semiconductor surfaces [10].

Method of solution
Standard full dynamic LEED calculations [11] are used to obtain the electron wave field diffracted from a *reference surface*. Using the Tensor LEED approximation [12–14], geometrical, vibrational [15] and chemical [16,17] parameters in a large portion of the parameter space around that *reference structure* are then varied. A search algorithm [18] allows to retrieve the *best fit* between measured data and calculated spectra reliably for typically 15 or more parameters.

Restrictions on the complexity of the problem
The surface is required to be periodic in two dimensions. Aspherical atomic scattering can only be included within the Tensor LEED approximation, not in the full dynamic reference calculation.

Typical running time
Running times depend very much on the actual problem. Times of 1–10 hours for systems of intermediate complexity including a structure optimization on a 500 MHz Compaq XP1000 workstation may serve as an estimate.

LONG WRITE-UP

1. Introduction

Since its beginnings in the late 1920's [19], low energy electron diffraction (LEED) has been a major driving force in the field of surface science, particularly for surface crystallography. The recent years have seen a tremendous growth of the complexity of surface structures solved, much of which is due to the experimental and theoretical progress made in low energy electron diffraction (for recent reviews see Refs. [2–4]), and, in particular, the introduction of the Tensor LEED approximation [12,14] for the calculation of LEED intensities. Examples of structures solved by this technique range from simple metal surfaces via reconstructed clean and adsorbate covered surfaces, multicomponent systems such as alloys or epitaxial systems up to complex reconstructions on semiconductor surfaces [20].

Using LEED, two steps are required in order to obtain accurate crystallographic information about a given surface. The first step is simply (yet not always simple) to provide from experiment a sufficiently large data base of LEED intensity vs. energy $I(E)$ spectra measured from the diffraction pattern of the surface under investigation. The second step is their numerical evaluation in order to retrieve the atomic configuration at the surface. Since multiple-scattering effects in LEED are strong, there exists no general method to extract the desired information directly from the experimental data. Instead, one resorts to a *best fit* strategy that involves the calculation of $I(E)$ spectra for many plausible models of the surface. By comparing these to the experimental curves, it is routinely possible to determine atomic positions near that surface (with an accuracy of a few pm in favorable cases), its chemical composition including possible substitutional disorder, and other features such as vibrational amplitudes.

When putting this approach into practice, one is simultaneously confronted with at least two necessities: first, a computational method giving easy access to $I(E)$ spectra for many different combinations of parameters, and second, a reliable search strategy to identify the *best fit* in the chosen portion of the parameter space, avoiding

possible local minima. The Erlangen Tensor LEED package TensErLEED is a collection of programs tailored to tackle both tasks. Its description in the remaining parts of this paper is organized as follows: Section 2 gives an overview of the theoretical concepts underlying LEED calculations in general, introduces the Tensor LEED approximation, and describes the search strategy for the *best fit* adopted by us. Section 3 focuses on the implementation of these concepts, separated into four different parts: the full dynamic calculation of LEED amplitudes and $I(E)$ spectra of a *reference surface*, the calculation of amplitude changes through modifications of this structure within the Tensor LEED approximation, the retrieval of $I(E)$ spectra from these, and, again, the structural search algorithm supplied with the package. Section 4 deals with the organization of the package, and, finally, Section 5 gives a brief description of the test run supplied with the package.

The programs presented here have been used and enhanced in our group for almost two decades. Of course, their basis lies partially in older work. In particular, the full dynamic section of the package is directly derived from the collection of subroutines originally published by Van Hove and Tong [11]. An early version of the programs of Rous and Pendry [21,22] served as the basis for the implementation of Tensor LEED. Although we aim to present the current version of the TensErLEED package as completely as possible, we also attempt to refer to these original authors wherever appropriate. Through the above-mentioned programs, the TensErLEED package shares some common roots with the Berkeley "Automated Tensor LEED" package [23] that is widely used in surface crystallography today. However, it should be pointed out that, otherwise, both packages represent independent trains of development.

2. Theoretical background

Conventional full dynamic LEED calculations are limited by the fact that the time to compute the electron wave diffracted from a surface scales essentially as N^3 with N the number of independent scattering centers in the unit cell. So, with growing complexity of the surface under consideration, the computational effort to calculate a single LEED spectrum rises rapidly. To make things worse, the number of free parameters in a complex structure optimization is usually also much larger than for a "simple" surface, requiring many more $I(E)$ calculations. For example, in a straightforward grid search, the effort scales exponentially with the number of free parameters.

The Tensor LEED approximation [12–14] aims to resolve the problem of scaling behaviour with system size. It is based on the idea that, while low energy electron diffraction from a surface is governed by multiple scattering, a small modification of a given *reference surface* will only cause a small change in the diffracted electron wave field and may be treated by a perturbational approach. Once the full dynamic scattering from the *reference surface* is known, the wave functions of structurally similar surfaces are deduced with computing times scaling only linearly with the number of atoms involved. In other words, the strategy is

- to perform a full dynamic LEED calculation once for a *reference surface*, saving those parts of the wave function needed later for the perturbational treatment, and then
- to use that wave function to calculate approximate amplitude changes δA for geometrically, stoichiometrically or vibrationally modified surface structures.

The remaining task is to reliably determine the structural *best fit* within the portion of the parameter space accessible via Tensor LEED. A number of optimization strategies for LEED has been proposed over the years [24–30]. Our preferred method is that introduced by Kottcke and Heinz [18]. It offers the advantage of being a global approach to structure optimization (avoiding the risk of being caught in the local optimum closest to the starting position of the search) but with improved scaling behaviour compared to other global schemes, e.g., simulated annealing [29].

In the following, we will first outline the formal justification for Tensor LEED. Then, we focus on the determination of those parts of the LEED wave field in the reference structure needed for Tensor LEED, describe the calculation of amplitude changes by structural modifications, and review the structural search algorithm adopted here. Since many related issues have been discussed at length in previous publications, we will refer to the literature where appropriate and emphasize only those aspects specific to the Erlangen Tensor LEED package. Note, however,

that our derivation of Tensor LEED deviates slightly from that of Rous and Pendry [13], clearing up a minor point in the original formulation while not affecting the actual result.

2.1. The concept of Tensor LEED

As mentioned, the Tensor LEED approximation is based on a full dynamic calculation of the LEED wave function for one suitably chosen *reference surface*. Detailed accounts of the theory behind full dynamic calculations can be found in the literature (e.g., in Refs. [1,31]) and will not be repeated. Here, some basic formulae suffice to derive the Tensor LEED approximation to a full dynamic calculation.

Our starting point is the scattering of electrons from a surface described by the muffin-tin approach, i.e. by a scattering potential

$$V = V_0 + \sum_i v_i \tag{1}$$

with v_i the individual atomic potentials forming the surface and V_0 the complex *inner potential*. Following the original treatment of Rous and Pendry [13], we write the LEED state diffracted by the surface as

$$\left|\Phi^+(\boldsymbol{k}_\parallel)\right\rangle = G^+\left|\epsilon(\boldsymbol{k}_\parallel)\right\rangle, \tag{2}$$

where $|\epsilon(\boldsymbol{k}_\parallel)\rangle$ stands for the incident state in the far past and G^+ denotes the Green function for the full surface potential. $|\epsilon(\boldsymbol{k}_\parallel)\rangle$ is chosen such that, when acted upon by the free space Green function G_0^+, it generates a plane wave of given energy E with a momentum component $\boldsymbol{k}_\parallel$ parallel to the surface. The total Green function G^+ can be expressed as a Born series in the crystal potential

$$G^+ = G_0^+ + G_0^+ V G_0^+ + G_0^+ V G_0^+ V G_0^+ + \cdots, \tag{3}$$

or, using the potential of Eq. (1) and the definition of the atomic t-matrix, $t_i = v_i + v_i G_0^+ v_i + \cdots$, as

$$G^+ = G_0^+ + \sum_i G_0^+ t_i G_0^+ + \sum_i \sum_{j,j\neq i} G_0^+ t_i G_0^+ t_j G_0^+ + \cdots. \tag{4}$$

In principle, Eq. (4) contains all we need to know to solve the LEED problem full dynamically. For convenience, the constant inner potential V_0 is included as an energy offset in the free space Green function G_0^+. Semiclassically speaking, each scattering atom in the unit cell is now represented by its t-matrix t_i, and the path between two scattering events is described by G_0^+. Note that, in Eq. (4), *successive* scattering events at the *same* atom do not occur. The atomic t-matrix already includes all such terms.

In order to derive the Tensor LEED approximation, we consider a modified surface with atomic t-matrices $\tilde{t}_i = t_i + \delta t_i$. The Green function $\widetilde{G}^+$ of the new surface becomes

$$\widetilde{G}^+ = G_0^+ + \sum_i G_0^+(t_i + \delta t_i)G_0^+ + \sum_i \sum_{j,j\neq i} G_0^+(t_i + \delta t_i)G_0^+(t_j + \delta t_j)G_0^+ + \cdots. \tag{5}$$

By carefully reordering all terms of this expression, one can arrange $\widetilde{G}^+$ as a power series in δt_i, i.e.

$$\widetilde{G}^+ = G^+ + \sum_i G_i^{\text{start}} \cdot \delta t_i \cdot G_i^{\text{end}} + \mathrm{O}\big((\delta t_i)^2\big), \tag{6}$$

where (using the reciprocity theorem for Green functions [32])

$$G_i^{\text{end}} = G^+ - G_0^+ t_i G_0^+ - \sum_{j,j\neq i} G_0^+ t_i G_0^+ t_j G_0^+ - \cdots \quad \text{and}$$

$$G_i^{\text{start}} = G_i^{\text{end}\,\dagger *} \tag{7}$$

sum up all scattering paths that *do not* end with (begin with) a scattering process at atom number i, respectively. The Tensor LEED approximation is obtained from Eq. (6) by keeping only the term linear in δt_i and discarding all higher orders. The quantities $G_i^{\mathrm{end}\,\dagger *}$ and G_i^{end} surrounding δt_i again reflect the fact that, in Eq. (4), there must not be any *successive* scattering events at the same atom. For the modified structure, this restriction implies that no scattering path may contain the undisplaced t-matrix t_i and then its change δt_i immediately following one another.

Eq. (6) offers some convenient computational advantages. The only quantities that depend on the modified surface are the changes in the atomic t-matrices $\delta t_i - G^+$ and G_i^{end} merely depend on the *reference surface*. So, for each particular modified structure it is only necessary to compute δt_i for all required sites i – the remaining terms need to be evaluated for the *reference surface* only. Moreover, changing the properties of different atoms simultaneously enters different terms in the sum in Eq. (6), making it possible to evaluate each term in that sum independently for each atom, effectively linearizing the effort to compute changes in the LEED state for more than one atom at a time.

Having obtained a formal expression for the Tensor LEED approximation, all we need to do is to translate it into the familiar language of LEED calculations. We are interested in calculating the plane wave amplitudes $\tilde{A}^-_{g'}$ emerging from the modified surface ($\{\boldsymbol{g}'\}$ denotes the set of reciprocal lattice vectors of the distorted surface, which may differ from that of the *reference surface*). They are usually evaluated at some position Z_0 outside the surface, the origin of the calculation, by projecting the LEED state on the wave function $\langle Z_0, \boldsymbol{k}_\parallel + \boldsymbol{g}'|$ given by

$$\langle \boldsymbol{r}|Z_0, \boldsymbol{k}_\parallel\rangle = \exp(\mathrm{i}\boldsymbol{k}_\parallel \cdot \boldsymbol{r}) \cdot \delta(z - Z_0). \tag{8}$$

Using Eq. (2), we obtain

$$\tilde{A}^-_{g'} = \left\langle Z_0, \boldsymbol{k}_\parallel + \boldsymbol{g}'|\widetilde{G}^+|\epsilon(\boldsymbol{k}_\parallel)\right\rangle, \tag{9}$$

an expression that can be split into the amplitudes leaving the *reference surface*, $A^-_{g'}$, and a sum of amplitude changes $\delta\tilde{A}^-_{i,g'}$ induced by modifying the scattering properties of individual atoms i,

$$\tilde{A}^-_{g'} = A^-_{g'} + \sum_i \delta\tilde{A}^-_{i,g'} \tag{10}$$

with

$$\delta\tilde{A}^-_{i,g'} = \left\langle Z_0, \boldsymbol{k}_\parallel + \boldsymbol{g}'|G_i^{\mathrm{end}\,\dagger *} \cdot \delta t_i \cdot G_i^{\mathrm{end}}|\epsilon(\boldsymbol{k}_\parallel)\right\rangle \tag{11}$$

by inserting Eq. (6). Since atomic t-matrices enter a LEED calculation in their angular momentum representation centered at the atomic position $\boldsymbol{r}_i$, we can rewrite (11) to obtain

$$\delta\tilde{A}^-_{i,g'} = \sum_{l,m,l',m'} T_{i,g',l,m,l',m'} \cdot \left\langle \boldsymbol{r}_i; l, m|\delta t_i|\boldsymbol{r}_i; l', m'\right\rangle,$$

with

$$T_{i,g',l,m,l',m'} = \left\langle Z_0, \boldsymbol{k}_\parallel + \boldsymbol{g}'|G_i^{\mathrm{end}\,\dagger *}|\boldsymbol{r}_i; l, m\right\rangle \cdot \left\langle \boldsymbol{r}_i; l', m'|G_i^{\mathrm{end}}|\epsilon(\boldsymbol{k}_\parallel)\right\rangle. \tag{12}$$

$T_{i,g',l,m,l',m'}$ now contains all terms that depend on the *reference surface* and is the quantity referred to as the *tensor*.

2.2. The tensor

In order to obtain the tensor $T_{i,g',l,m,l',m'}$ from a numerical calculation, it is necessary to relate the quantities

$$\left\langle \boldsymbol{r}_i; l', m'|G_i^{\mathrm{end}}|\epsilon(\boldsymbol{k}_\parallel)\right\rangle \quad \text{and} \tag{13}$$

$$\left\langle Z_0, \boldsymbol{k}_\parallel + \boldsymbol{g}'|G_i^{\mathrm{end}\,\dagger *}|\boldsymbol{r}_i; l, m\right\rangle \tag{14}$$

to a conventional LEED calculation.

As mentioned before, the operator G_i^{end} contains all scattering paths included in the total Green function G^+ except those that end with a scattering event at atom i. In Eq. (13), this operator acts upon the incident wave and is then evaluated in angular momentum components centered around atom i. In other words, Eq. (13) contains nothing else than the components of the *wave field incident upon but not scattered by atom i* of the *reference surface*. Now, as pointed out by Rous and Pendry [13], the *total* LEED wave function can be expressed in angular momentum components around atom i as

$$\langle \boldsymbol{r}|\Phi^+(\boldsymbol{k}_\parallel)\rangle = \sum_{lm} A_{i;lm}(\boldsymbol{k}_\parallel)\cdot\left[j_l(\kappa|\boldsymbol{r}-\boldsymbol{r}_i|)+t_{i,l}\cdot h_l^1(\kappa|\boldsymbol{r}-\boldsymbol{r}_i|)\right]\cdot Y_{lm}(\widehat{\boldsymbol{r}-\boldsymbol{r}_i}), \tag{15}$$

i.e. a sum of the unscattered wave plus the contribution scattered by atom i. Simply leaving away the latter part generates what we are looking for,

$$\langle \boldsymbol{r}|G_i^{\text{end}}|\epsilon(\boldsymbol{k}_\parallel)\rangle = \sum_{lm} A_{i;lm}(\boldsymbol{k}_\parallel)\cdot j_l\big(\kappa|\boldsymbol{r}-\boldsymbol{r}_i|\big)\cdot Y_{lm}(\widehat{\boldsymbol{r}-\boldsymbol{r}_i}), \tag{16}$$

leading to

$$\langle \boldsymbol{r}_i;l,m|G_i^{\text{end}}|\epsilon(\boldsymbol{k}_\parallel)\rangle = A_{i;lm}(\boldsymbol{k}_\parallel). \tag{17}$$

The evaluation of the second part of the tensor, Eq. (14), yields a similar result. Following the lines of Rous and Pendry [13], we notice that

$$\langle Z_0,\boldsymbol{k}_\parallel+\boldsymbol{g}'|G_i^{\text{end}\dagger *}|\boldsymbol{r}_i;l,m\rangle = (-1)^m\langle \boldsymbol{r}_i;l\ -m|G_i^{\text{end}}|Z_0,-(\boldsymbol{k}_\parallel+\boldsymbol{g}')\rangle, \tag{18}$$

and since (Ref. [13])

$$\begin{aligned}\langle \boldsymbol{r}|G_0^+|Z_0,-(\boldsymbol{k}_\parallel+\boldsymbol{g}')\rangle &= (1/2\mathrm{i}\kappa k_{g'z}^+\Omega)\cdot\exp\big[\mathrm{i}\boldsymbol{K}_{g'}^+\cdot(\boldsymbol{r}-Z_0)\big]\\ &= (1/2\mathrm{i}\kappa k_{g'z}^+\Omega)\cdot\langle \boldsymbol{r}|G_0^+|\epsilon(-(\boldsymbol{k}_\parallel+\boldsymbol{g}'))\rangle\end{aligned} \tag{19}$$

(with κ the modulus of the momentum of the "free" electron and Ω the area of the surface unit mesh for proper normalization), we see that Eq. (14) leads to a result very similar to that of Eq. (17) except that the "incident" beam of the LEED state now has the lateral momentum component $-(\boldsymbol{k}_\parallel+\boldsymbol{g}')$ instead of $\boldsymbol{k}_\parallel$:

$$\langle Z_0,\boldsymbol{k}_\parallel+\boldsymbol{g}'|G_i^{\text{end}\dagger}|\boldsymbol{r}_i;l,m\rangle = (1/2\mathrm{i}\kappa k_{g'z}^+\Omega)\cdot A_{i;lm}\big(-(\boldsymbol{k}_\parallel+\boldsymbol{g}')\big). \tag{20}$$

Combining Eqs. (17) and (20), we see that the tensor may be obtained from conventional LEED calculations with different directions of incidence:

$$T_{i,g',l,m,l',m'} = \frac{1}{2\mathrm{i}\kappa k_{g'z}^+\Omega}(-1)^m\cdot A_{i;lm}\big(-(\boldsymbol{k}_\parallel+\boldsymbol{g}')\big)\cdot A_{i;l'm'}(\boldsymbol{k}_\parallel). \tag{21}$$

In brief, the strategy to obtain the spherical-wave amplitudes $A_{i;lm}(\boldsymbol{k}_\parallel)$ is to perform a conventional LEED calculation (required anyways to calculate the LEED amplitudes $A_{g'}^-$ leaving the surface) and utilize the quantities constructed along the way. Standard LEED calculations are performed in two different representations. Multiple scattering within a layer j (possibly composed of several sublayers i) is evaluated in angular momentum space, the resulting layer diffraction matrices are then transformed to plane wave representation, and combined to give $A_{g'}^-$. For a given incident beam with component $\boldsymbol{k}_\parallel$, we require the plane-wave amplitudes $B_{j;g'}^-$ and $B_{j;g'}^+$ incident on a particular crystal layer j. Since we prefer the self consistent layer stacking method [1] over the renormalized forward scattering (RFS) [31] formalism originally employed by Rous and Pendry [21], $B_{j;g'}^-$ and $B_{j;g'}^+$ do not immediately result from the full dynamic treatment of interlayer scattering. Instead, in a conventional layer stacking step, one combines the reflection and transmission matrices $\boldsymbol{R}_{j;g_1,g_2}^{\pm}$ and $\boldsymbol{T}_{j;g_1,g_2}^{\pm}$ of a layer with the effective reflection matrix $\boldsymbol{R}_{j+1;g_1,g_2}^{\text{eff},-}$ of an underlying stack of layers to obtain the reflection matrix of that stack with layer j on top, $\boldsymbol{R}_{j;g_1,g_2}^{\text{eff},-}$. The formalism can easily be modified to additionally provide two more effective diffraction

matrices, $\boldsymbol{T}^{\mathrm{eff},-}_{j;g_1,g_2}$ and $\boldsymbol{T}^{\mathrm{eff},+}_{j;g_1,g_2}$, which describe the *propagation of plane wave amplitudes* $B^{+}_{j;g'}$ *incident on layer* j *from above into the space between layer* j *and the underlying slab*:

$$B^{-}_{j;g'} = \boldsymbol{T}^{\mathrm{eff},-}_{j;g_1,g_2} \cdot B^{+}_{j;g'}, \tag{22}$$

$$B^{+}_{j+1;g'} = \boldsymbol{T}^{\mathrm{eff},+}_{j;g_1,g_2} \cdot B^{+}_{j;g'}. \tag{23}$$

Once all layer stacking steps have been performed, the required plane wave amplitudes can be extracted for all layers j in question, beginning with known amplitudes $B^{+}_{1;g'}$ incident on the topmost layer ($j = 1$) and moving into the crystal recursively by way of Eqs. (22) and (23).

For each layer j, the only remaining task is to retrieve the spherical wave amplitudes $A_{i;lm}(\boldsymbol{k}_\parallel)$ incident on sublayer i of that layer. The mathematics needed to perform this step for a Bravais layer (only one sublayer) is outlined in chapter IVC of Pendry's book [31], and its application to Tensor LEED has been discussed in Refs. [13,21]. In order to convert $B^{-}_{j;g'}$ and $B^{+}_{j;g'}$, one employs the $\boldsymbol{X}$ matrix which contains all information on intralayer multiple scattering in the angular momentum representation *just before* the final scattering process of each scattering path occurs:

$$A_{i;lm}(\boldsymbol{k}_\parallel) = \sum_{l',m'} A^{0}_{i;lm}(\boldsymbol{k}_\parallel) \cdot (1 - \boldsymbol{X})^{-1}_{l'm',lm} \quad \text{with} \tag{24}$$

$$A^{0}_{i;lm}(\boldsymbol{k}_\parallel) = 4\pi \mathrm{i}^{l} \sum_{g} \left(B^{+}_{j;g} Y^{*}_{lm}(\boldsymbol{K}^{+}_{g}) + B^{-}_{j;g} Y^{*}_{lm}(\boldsymbol{K}^{-}_{g}) \right). \tag{25}$$

Obtaining $\boldsymbol{X}$ is a part of every conventional LEED calculation.

For a composite layer (more than one sublayer i), the procedure is slightly different. In the standard matrix-inversion formalism based on Beeby's idea [11,33,34], the scattering matrices $\overline{\boldsymbol{T}}^{+}_{i;lm}(\boldsymbol{k}_\parallel + \boldsymbol{g})$ and $\overline{\boldsymbol{T}}^{-}_{i;lm}(\boldsymbol{k}_\parallel + \boldsymbol{g})$ defined by Eqs. (33) and (34) of Ref. [34] already contain all scattering paths that end at sublayer i, for incidence towards $+z$ or $-z$, respectively, with a lateral momentum $\boldsymbol{k}_\parallel + \boldsymbol{g}$. One simply needs to remove the final scattering process at atom i (i.e. divide by the atomic t-matrix t_i) in order to obtain

$$A_{i;lm}(\boldsymbol{k}_\parallel) = \frac{2\kappa \mathrm{i}^{l}}{t_{i;l}} \cdot \sum_{g} \left(B^{+}_{j;g} \cdot \overline{\boldsymbol{T}}^{+}_{i;lm}(\boldsymbol{k}_\parallel + \boldsymbol{g}) + B^{-}_{j;g} \cdot \overline{\boldsymbol{T}}^{-}_{i;lm}(\boldsymbol{k}_\parallel + \boldsymbol{g}) \right). \tag{26}$$

Again, the calculation of $\overline{\boldsymbol{T}}^{+}_{i;lm}(\boldsymbol{k}_\parallel + \boldsymbol{g})$ and $\overline{\boldsymbol{T}}^{-}_{i;lm}(\boldsymbol{k}_\parallel + \boldsymbol{g})$ is directly part of the full dynamic calculation of layer diffraction matrices [11,34] so that only little extra numerical effort arises.

2.3. Perturbing the reference structure

Having obtained the tensor, the next step is to express perturbations of the reference structure in terms of individual atomic t-matrices. For the most important application of LEED, the determination of surface geometry, changes of individual atomic positions are translated into a modified t-matrix by evaluating the scattering properties of that atom from a different spatial origin. A displacement of the scatterer by an amount $\delta \boldsymbol{r}_i$ is expressed in terms of a translational operator $\tau(\delta \boldsymbol{r}_i)$, leading to

$$\delta t_i = \tilde{t_i} - t_i = \tau(\delta \boldsymbol{r}_i) \cdot t_i \cdot \tau(-\delta \boldsymbol{r}_i) - t_i. \tag{27}$$

Details of the computational procedure in angular momentum space have been given by Rous and Pendry [13,21].

Of course, the application of Tensor LEED is not restricted to geometrical quantities. Another example is the treatment of substitutional disorder in the crystal lattice which, through the well-established average t-matrix approximation [35], directly affects the t-matrix of the scatterer in question. In general, if a lattice site i is occupied statistically by two chemical elements A and B, the t-matrix for that site is given by

$$t_i = x_i \cdot t_A + (1 - x_i) \cdot t_B, \tag{28}$$

where x_i is the probability of finding element A on site i and t_A and t_B are the t-matrices of two elements A and B. In chemical Tensor LEED [16,17] this means

$$\delta t_i = \tilde{x}_i \cdot t_A + (1 - \tilde{x}_i) \cdot t_B - t_i = \delta x_i \cdot (t_A - t_B) \tag{29}$$

when the occupation probability of site i by element A is changed to $\tilde{x}_i = x_i + \delta x_i$.

Of course, two different elements occupying the same lattice site i near a surface need not necessarily reside in exactly the same position. In that case, the modified t-matrix is obtained by averaging t_A and t_B when displaced from their original position according to Eq. (27), yielding

$$\delta t_i = \tilde{x}_i \cdot \tau(\delta \boldsymbol{r}_{i,A}) t_A \tau(-\delta \boldsymbol{r}_{i,A}) + (1 - \tilde{x}_i) \cdot \tau(\delta \boldsymbol{r}_{i,B}) t_B \tau(-\delta \boldsymbol{r}_{i,B}) - t_i. \tag{30}$$

In contrast to the previous case (Eq. (29)), the resulting t-matrix is now intrinsically aspherical (as in Eq. (27)) and would be difficult to handle in a full dynamic calculation. Using Tensor LEED, the additional computational effort is drastically reduced. In fact, it is possible to rewrite Eq. (10) so that

$$\tilde{A}^-_{g'} = A^-_{g'} + \sum_i \left(\tilde{x}_i \cdot \delta \tilde{A}^-_{i,g'}(\tilde{x}_i = 1) + (1 - \tilde{x}_i) \cdot \delta \tilde{A}^-_{i,g'}(\tilde{x}_i = 0) \right). \tag{31}$$

In the case of substitutional disorder, the evaluation of Eqs. (30) and (12) is only required for $\tilde{x}_i = 1$ (full occupation by element A) and $\tilde{x}_i = 0$ (full occupation by element B). The results for all intermediate values $\tilde{x}_i$ can then be obtained in the final step of the calculation, again reducing the necessary computational effort.

A third frequent application of Tensor LEED is the treatment of thermal vibrations [15]. In the case of isotropic vibrations with a root mean square amplitude a_{vib}, this effect is taken into account through a Debye–Waller Factor multiplied to the atomic t-matrix [31]. Thus, we obtain

$$\delta t_i = \tilde{t}_i(a_{\mathrm{vib}}) - t_i. \tag{32}$$

Of course, Tensor LEED is not restricted to isotropic thermal vibrations. In fact, any non-diagonal t-matrix may be used, allowing for the efficient treatment of anisotropic and anharmonic vibrations [36] of individual atoms in the unit cell. The same advantage may even be exploited to approximate any asphericities of the atomic scattering potential within the muffin-tin approximation [37], avoiding a complex full dynamic treatment.

In practice, one frequently encounters multi-parameter problems where several different quantities are varied simultaneously that apply to the same lattice site i. It is then necessary to compute the desired amplitude changes for each combination of these parameters separately, storing them so that a superposition of amplitude changes from different atoms is possible in the final step.

2.4. Structural search in a multi-parameter space

Once the desired changes $\delta \tilde{A}^-_{i,g'}$ between amplitudes diffracted from the *reference surface* and the modified structure have been calculated, the contributions from each atom i are added according to Eqs. (10) or (31), and LEED intensities of the modified structure can be obtained as usual, i.e.

$$I_{g'}(E) = \frac{k^+_{g'z}}{k^+_{0z}} \cdot \left| \tilde{A}^-_{g'} \right|^2. \tag{33}$$

They can then be compared to experimental $I(E)$ data from the surface in question, e.g., by conventional R-factor analysis [1].

Generally, the surface structural model contains a number of free parameters that need to be optimized until the combination is found which yields the best, i.e. minimum R-factor between theoretical and experimental data. The simplest approach to this problem is a grid search, choosing a number of grid points for each

parameter and calculating $I(E)$ spectra and R-factors for each combination of these parameters. Tensor LEED facilitates this approach dramatically since the simple superposition of amplitude changes due to different atoms is much faster than a conventional multiple-scattering computation. Yet, even with this advantage a grid search is far too time-consuming in a reasonably large parameter space, and a more sophisticated approach should be envisioned.

As mentioned above, our preferred search strategy is that introduced by Kottcke and Heinz [18], which retains both global and local aspects of a search. After randomly choosing an initial set of trial structures within the parameter space to be investigated, each of these trial structures develops individually towards the global minimum in the following manner. A new trial structure is randomly chosen according to a Gaussian probability distribution (PD) centered about the previous structure. The new structure replaces the previous one if its R-factor is lower. Otherwise, the search keeps the previous structure, and so on. The search is stopped once a certain number of trial structures (e.g., 20%) have reached the same parameter configuration with minimum R-factor.

The proper handling of the PD employed in the choice of successive trial structures is central to this search scheme. While the average R-factor is high, the PD's width stays large, allowing the search procedure to access the entire parameter space. As the R-factor decreases during the search, so does the PD's width, so that, near its end, the search is practically a local one. Moreover, the PD's width is chosen individually for each parameter. For parameters with a strong influence on the fit, a narrower distribution is used than for those with less impact (for details see Ref. [18]).

An important feature of a search procedure is its scaling behaviour with the number of free parameters to be optimized, N. For the above scheme, the total number of $I(E)$ calculations needed to identify the *best fit* scales approximately as $N^{2.5}$ [18]. This scaling behaviour is slightly worse than that of steepest-descent-like methods (approximately N^2) [24–28] but clearly better than that reported for simulated annealing (N^6) [29] or even the exponential scaling of a grid search. A further benefit of the procedure comes from the nature of its convergence criterion. With several trial structures developing simultaneously and independently of one another, only a certain number of them need to reach the global minimum of the R-factor surface for the *best fit* to be identified. So, in unfavorable cases with several local minima at a comparable R-factor level, not only will the global minimum be identified but also will the other minima be reached by some trial structures, giving an important insight into the properties of the optimization problem under consideration.

3. The Erlangen Tensor LEED package

In line with Section 2, the computational strategy employed for a full structure analysis using the TensErLEED package consists of three steps, visualized in Fig. 1. In the reference calculation, full dynamic $I(E)$ spectra are calculated for one particular surface structure. Along with the usual $I(E)$ output, the outgoing LEED amplitudes $A^-_{g'}$ and the wave field components incident on each atomic site i in the structure, $A_{i;lm}(\boldsymbol{k}_\parallel)$ and $A_{i;lm}(-(\boldsymbol{k}_\parallel+\boldsymbol{g}'))$, are stored in separate files. The second step is the calculation of amplitude changes $\delta\tilde{A}^-_{i,g'}$ for a set of suitably chosen geometrical and thermal displacements around the reference values, for all chemical elements possibly found on that site. In the last step, $I(E)$ spectra are calculated for different parameter combinations according to Eqs. (10) or (31). In our case, the structural search algorithm described above handles the comparison to the experimental data, identifying the combination of structural parameters that yields the minimum R-factor within a given portion of the parameter space. In case that the *best fit* deviates strongly from the *reference surface*, a further reference calculation using the *best fit* as its input is advisable in order to minimize possible uncertainties introduced by the Tensor LEED approximation. Needless to say, if there are different classes of structural models (e.g., different adsorption sites on a surface), the procedure must be repeated for them all.

The following sections give details on the input data needed for the above steps. Each case includes an overview of the program that performs the actual calculation.

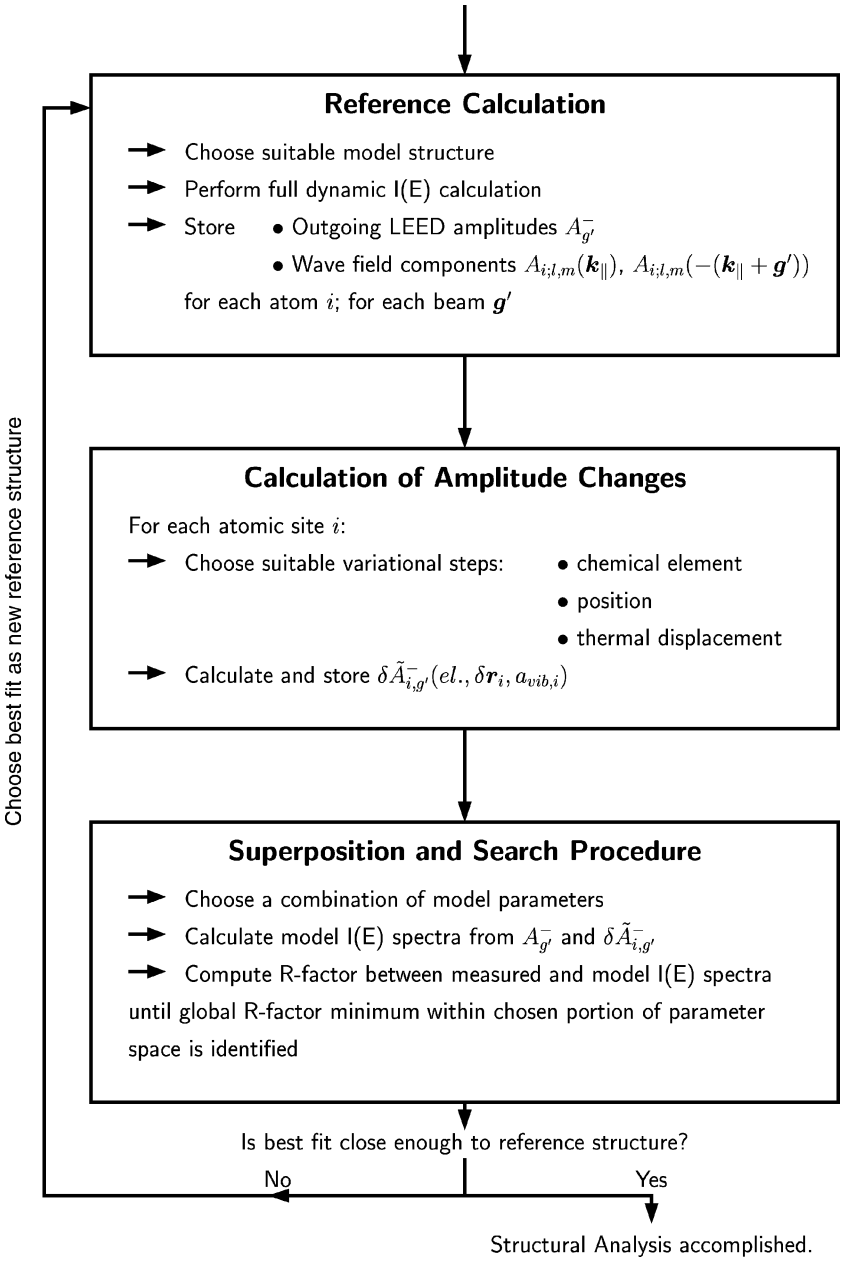

Fig. 1. Flowchart of a complete LEED $I(E)$ analysis using the TensErLEED package.

3.1. The reference calculation

Setting up the calculation

In the following, we describe the strategy to set up the input for a full dynamic reference calculation (program ref-calc.f) that additionally yields the output required for Tensor LEED. Table 1 contains a list of general input

Table 1
(2D) geometry input data and quantities controlling the accuracy of a *reference calculation* using ref-calc.f

Quantity	Format	Explanation
title	A80	Descriptive title
ei, ef, de	3F7.2	perform calculation for energy values between *ei* and *ef* in steps of *de* [eV]
ara1	2F7.4	2D surface unit vectors (“bulk”) [Å]
ara2	2F7.4	
ss1	2F7.4	2D registry shifts of the unit cell –
ss2	2F7.4	unused in current version
ss3	2F7.4	
ss4	2F7.4	
arb1	2F7.4	2D surface unit vectors (“overlayer”) [Å]
arb2	2F7.4	
so1	2F7.4	2D registry shifts of the unit cell –
so2	2F7.4	unused in current version
so3	2F7.4	
fr, ase	2F7.4	*fr* currently unused. *ase* is the distance between top layer and vacuum boundary [Å]
spqf, ksym	“beam list” generated by utility beamgen.f (see text)	
tst	F7.4	To describe interlayer scattering, use only those beams whose amplitude does not decay by a factor of less than *tst* between two layers.
npu	26I3	$npu(k) = j$ means that full dynamic $I(E)$ data are stored for beam no. j of the “beam list” *spqf*
theta, fi	2F7.2	polar and azimuthal angles of incidence [°]
eps	F7.4	convergence criterion for layer doubling
liter	I3	maximum number of layer doubling iterations
lmax	I3	highest angular momentum quantum number to be considered in multiple scattering
nel	I3	number of chemical elements for which phase shifts are provided
The following block is expected for each energy step value in phase shift table:		
es	F7.4	current energy step value [Hartrees]
The following input is expected *nel* times:		
phss	10F7.4	scattering phase shifts $\delta_l(es)$, $l = 0, \ldots, l_{\max}$
iform	I3	0: binary, 1: ASCII output of tensor parts
The following line is expected for each beam for which tensor output is desired:		
pqfex	2F7.4	(2D) reciprocal lattice vector as in “beam list”

quantities including some that are not used by the current program version. Since they might be useful for future enhancements, their purpose is explained in Table 1 although they are not mentioned otherwise. All run-time input is read from the standard input channel.

After selecting a brief title and the energy range for which the calculation should be performed – from a start value *ei* up to *ef* in steps of *de* – the 2D unit mesh of the bulk-truncated surface (vectors *ara1* and *ara2*) and that of a possible superlattice (vectors *arb1* and *arb2*) are defined. The distance between the topmost atomic position in the crystal and the onset of the inner potential (towards the vacuum) is given by *ase*. Next, the program expects a list of beams *spqf* (in units of the bulk reciprocal lattice) to be used as the basis set $\{\boldsymbol{g}\}$ for all calculations in plane wave space (i.e. layer doubling and layer stacking). The accompanying utility beamgen.f should be used to generate this list for the required range of energy steps and interlayer distances.

At a given energy, only a subset of the input beams is used in the actual calculation. Its selection is governed by the *tst* criterion to ensure that enough evanescent waves are taken into account in layer doubling or layer stacking. Next, the numbers of those beams *npu* are listed for which full dynamic $I(E)$ output is desired. The polar angles *theta* and *fi* account for oblique incidence of the primary electron beam. Following them, the criteria *eps* and *liter* are set that govern the convergence behaviour of layer doubling. When a checksum criterion between two successive layer doubling steps differs by less than *eps*, the procedure is considered to be converged. In the unusual case of non-convergence, the layer doubling procedure is stopped after a maximum of *liter* iterations.

For the parts of the calculation in spherical wave representation, *lmax* is the highest angular momentum quantum number used. A list of scattering phase shifts *phss* for each chemical element, energy and angular momentum value is expected in the standard format used by the Van Hove/Tong codes [11]. [1]

To ensure proper convergence, we advise the careful choice of *tst*, *eps*, *liter*, and *lmax*. Usually, *tst* values between 0.002 and 0.0001 are sufficient; likewise, values of 0.001 for *eps* and above 8 for *liter* ensure good results. *lmax* should not be chosen excessively high since computing times scale strongly with it but sufficiently large to ensure that only negligibly small phase shifts δ_l of the chemical elements are ignored in the calculation. Since this depends on the elements and energy range used, no general advice as to its magnitude is given, although the classical approximation $l_{\max} \approx \kappa \cdot r_{\mathrm{mt}}$ (where r_{mt} is the muffin-tin radius) may serve as an estimate.

Tensor components may be written as binary or ascii data according to the flag *iform*. Since, in principle, Tensor LEED allows to consider a superstructure not actually present in the reference structure, tensor components $A_{i;lm}(-(\boldsymbol{k}_{\parallel} + \boldsymbol{g}'))$ may be obtained for beams $\boldsymbol{g}'$ not present in the full dynamic beam list $\{\boldsymbol{g}\}$. Although this feature is not active in the current program version, it may easily be re-animated by some minor adjustments in the source code. Therefore, the program expects a list *pqfex* of all beams $\{\boldsymbol{g}'\}$ for which tensor output is desired separate from that listing the full dynamic output beams *npu*.

What remains to be specified is the (3D) atomic structure of the (2D) surface unit mesh in the language of full dynamic LEED. Table 2 summarizes this final part of the input data. The multiple scattering formalism of Section 2 divides the crystal into separate parts, the smallest of which is a single atomic scatterer, represented by a t-matrix in l-space. The t-matrix is determined by the chemical nature of the scatterer, or, in the case of substitutional disorder, by the average chemical occupation of the lattice site in question. It also accounts for isotropic thermal vibrations of the atoms. The first part of the geometry input is, therefore, a list of the *nsite* different atomic scatterers present within the unit cell, distinguished by the occupation probability *conc* and vibrational amplitude *vib* of each element in the phase shift list on the respective lattice site. Next, these atomic scatterers are bundled together into *nltype* different types of atomic layers, the scattering properties of which are calculated in angular momentum space and then transformed to plane wave space. The 2D periodicity of a layer type is specified by the flag *latt*. *latt*=1 means that the superlattice unit mesh (*arb1*, *arb2*) is used, in the case of *latt*=2 the substrate unit mesh (*ara1*, *ara2*) is

[1] The TensErLEED package does not contain any utilities to calculate elemental scattering phase shifts. Separate tools for this task are available in the literature, most notably the package of Barbieri and Van Hove [23] or, based on that work, the one by Rundgren [38] which additionally takes the energy dependence of the real part of the inner potential into account.

Table 2
Input data for ref-calc.f that specify the (3D) atomic structure of the unit cell, i.e. the atomic scattering properties, the definition of atomic layers, and the stacking sequence of these layers to form a semi-infinite slab

Quantity	Format	Explanation
nsite	I3	number of different atomic scatterers (t-matrices) in the (3D) unit cell
The following block is expected *nsite* times:		
The following line is expected for each chemical element, in order of appearance in the phase shift list above:		
conc, vib	2F7.4	concentration and vibrational amplitude [Å] of current element in current scatterer
nltype	I3	number of different layer types, the scattering properties of which are calculated in angular momentum space
The following input is expected *nltype* times:		
latt	I3	1: layer type has periodicity *arb1, arb2* 2: layer type has periodicity *ara1, ara2*
nsub	I3	number of atoms (sublayers) in layer type
The following line is expected *nsub* times:		
stype *subpos*	I3, 3F7.4	atomic scatterer type no. of current sublayer sublayer position (z, x, y) within layer type (note that neg. z point towards vacuum) [Å]
tslab	I3	0: layer doubling for bulk required 1: layer doubling not required
asa	3F7.4	interlayer vector for layer doubling [Å]
toplayb	I3	upper of the two alternating layer types forming the bulk
botlayb	I3	lower of the two alternating layer types forming the bulk
asbulk	3F7.4	interlayer vector between the two layers forming the bulk unit for layer doubling [Å]
nstack	I3	number of layers to be “stacked” onto bulk (using the plane-wave representation)
The following block is expected *nstack* times:		
ltype *ldist*	I3, 3F7.4	layer type number to be stacked next (3D) interlayer vector below current layer [Å]
tens	I3	0: no tensor output required for current layer 1: output of tensor files required
If tensor output is required, the following line is expected for each sublayer of the current layer:		
layfile	A20	file name of tensor output file for current sublayer within current layer

assumed. Layer types for which tensor output is required must have the periodicity of the superlattice (*latt*=1). Each layer type may have an unlimited number *nsub* of primitive sublayers. For each sublayer, the type of scatterer *stype* and its position *subpos* within the layer must be specified. Note that the atomic coordinates are entered in the order (z, x, y) (where z is the axis perpendicular to the surface), and *negative* z values point towards the vacuum.

Once all layer types in the model surface are defined, they merely need to be "stacked" on top of each other (in plane-wave space) to yield the LEED reflection properties of the entire crystal. Unless required otherwise (*tslab*=1), the basis of this stack of layers is a slab generated by the layer doubling scheme [31] in order to obtain the reflection properties of a semi-infinite solid. Note that none of the atoms within this slab are accessible to Tensor LEED. In order to allow for a flexible sequence of layers, this slab consists of a unit composed of two different layer types *toplayb* and *botlayb* which are placed atop one another and separated by an interlayer vector *asbulk*. It is this unit which is then subject to the layer doubling scheme with the vector *asa* as its interlayer vector. *nstack* further layers are next stacked on top of the underlying slab, forming the actual surface which is accessible to Tensor LEED. For each of these layers, its layer type *ltype* and the interlayer vector *ldist* connecting it with the lower-lying layers must be defined. In case tensor output is required for that layer (*tens*=1), an output file name *layfile* is expected for every primitive sublayer of that layer. The geometry input is complete once the topmost layer has been specified in this way.

Aside from this input read by the program at run time, two more modifications are routinely expected from the user. Both concern the source code of the program itself. Since Fortran 77 requires static dimensions, it is necessary to adjust the array dimensions included in the file PARAM (created by the controlling shell script) according to the run-time input specified above. Second, the inner potential V_0 (see Eq. (1)) to be used in the calculation needs to be set. As it depends on the electron energy and the material under investigation, it must be possible to enter a general expression for its shape. Hence, a dedicated subroutine `muftin` is created by the controlling shell script where any functional form for $V_0(E)$ may be specified, to be compiled together with the remaining source code later. The real part of the inner potential in the bulk, *VV*, and its imaginary part *VPI* should be set here. The option to adjust an overlayer inner potential different from the bulk (*VO* and *VPIO*) is not supported by the current program version.

The program structure

The structogram of a reference calculation as implemented in ref-calc.f is sketched in Fig. 2. After reading the input described above (subroutines `readin`, `readtl` and `readgeo`) and verifying its consistency, the program generates those quantities that are independent of the electron energy. In particular, different types of Gaunt coefficients are calculated by the subroutines `caaa`, `celmg`, and `cppp` (see Ref. [11]). The remaining part of the calculation depends on the electron energy and must be repeated for each step in the desired energy range. Again, some general quantities are obtained first – in particular, the energy-dependent inner potential is set by the subroutine `muftin`. Then, atomic t-matrices are calculated for each type of atomic scatterer specified in the input. To that end, subroutine `tscatf` is invoked for each chemical element included in the input phase shift list, also taking into account its isotropic vibrational amplitude *vib*. The total t-matrix of the scatterer is obtained by averaging the elemental t-matrices according to their site occupation probability *conc*.

The following block of the program calculates the scattering properties of the entire crystal up to the second layer in plane-wave space, represented by diffraction matrices with the full dynamic input beam list $\{\boldsymbol{g}\}$ as their basis set. As shown in Section 2, Tensor LEED requires a separate multiple scattering calculation not just for incidence of the primary beam $\boldsymbol{k}_\parallel$ but also for all time-reversed output beams $-(\boldsymbol{k}_\parallel + \boldsymbol{g}')$ for which $I(E)$ spectra are to be approximated later. Hence, the program performs a loop over all required incident beams $\boldsymbol{k}_\parallel$ and $-(\boldsymbol{k}_\parallel + \boldsymbol{g}')$. Since we are dealing with diffraction *matrices* that take care of several beams $(\boldsymbol{k}_\parallel + \boldsymbol{g})$ at a time, a recalculation is only necessary if the previous and the current incident beam are not connected by a reciprocal lattice vector $\boldsymbol{g}$ of the *reference surface*. (In the current program version, this occurs only at oblique incidence $(\boldsymbol{k}_\parallel \neq 0)$ since the momentum of the primary beam $\boldsymbol{k}_\parallel$ and its time-reversed counterpart $-\boldsymbol{k}_\parallel$ then depend on the electron energy and

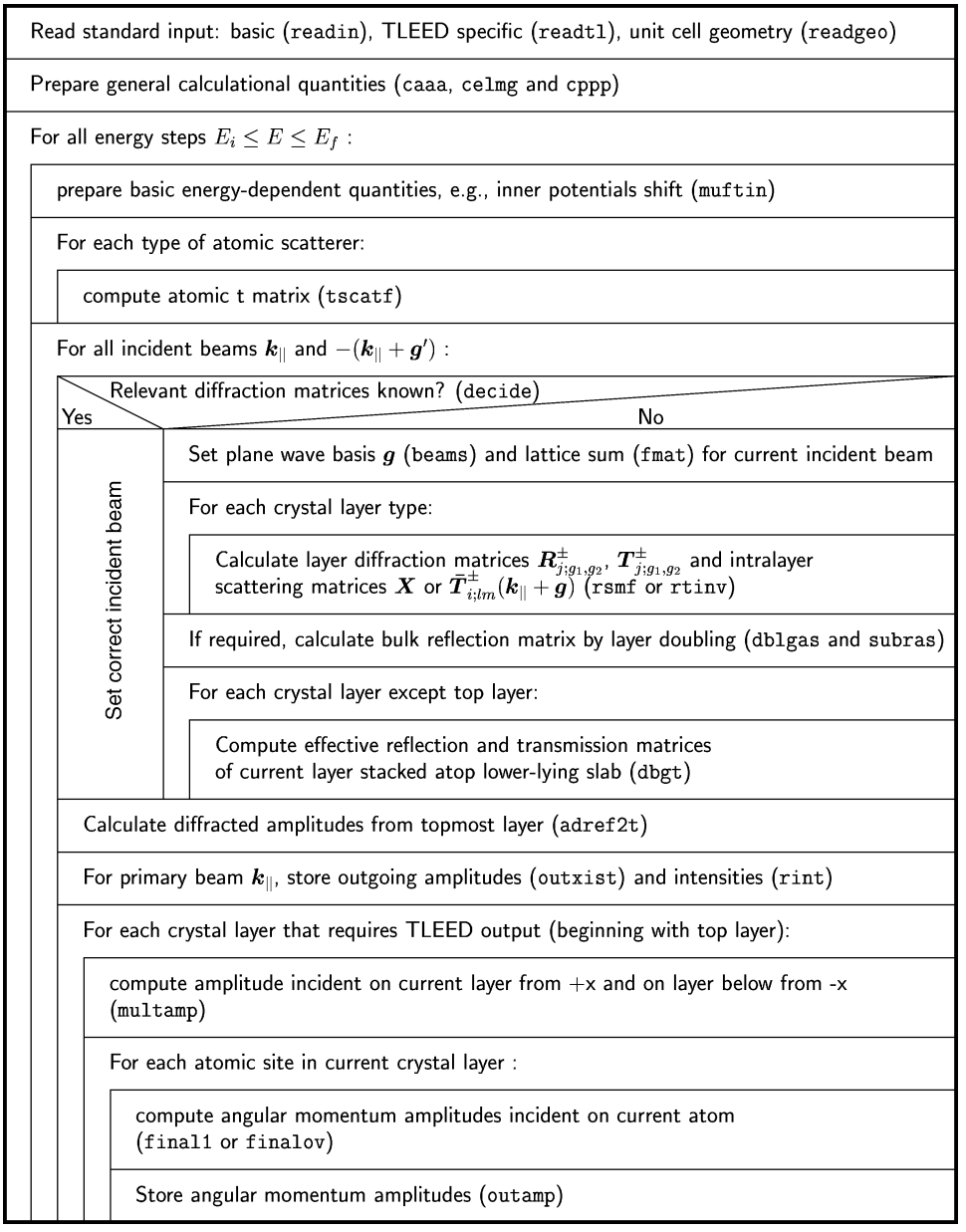

Fig. 2. Structogram of a full dynamic *reference calculation* using ref-calc.f.

are generally not related through a reciprocal lattice vector $\boldsymbol{g}$.) In short, only if the scattering matrices containing the current incident beam have not yet been calculated for a previous one (verified by the subroutine `decide`), a new and time-consuming complete multiple scattering calculation must be performed.

In that case, the program first chooses the plane-wave basis $\boldsymbol{g}$ appropriate for the current energy (subroutine `beams`) and obtains the lattice sums required in the calculation of layer diffraction matrices (subroutine `fmat`). For each layer type defined in the input, reflection and transmission matrices $\boldsymbol{R}^{\pm}_{j;g_1,g_2}$ and $\boldsymbol{T}^{\pm}_{j;g_1,g_2}$ are then computed

in angular momentum space and transformed to plane wave space. For a Bravais layer, subroutine `rsmf` is used, additionally storing its $\boldsymbol{X}$-matrix for later use. For the case of a composite layer, subroutine `rtinv` generates the layer diffraction matrices as well as the intra-layer scattering matrices $\overline{\boldsymbol{T}}^{\pm}_{i;lm}(\boldsymbol{k}_{\parallel}+\boldsymbol{g})$ for each sublayer i of the current layer type. Once the individual layer diffraction properties are known, the program calculates the reflection matrix of a semi-infinite slab (composed of up to two alternating layer types with two alternating interlayer vectors), combining first the diffraction matrices of both layers (subroutine `dblgas`), then stacking this composite object using the layer doubling technique (subroutine `subras`). The remaining layers of the crystal, in particular those which will be handled using Tensor LEED, are then stacked onto this slab self-consistently. In a full dynamic calculation, each layer stacking step would only require the effective reflection matrix of the lower-lying slab with the current layer placed atop. For Tensor LEED, subroutine `dbgt` additionally produces the effective transmission matrices of an electron wave field incident from above into the space between the current and the lower-lying layers (Section 2.2).

The stacking of the final (topmost) layer is different from the lower ones in that it needs to be repeated for each incident beam $\boldsymbol{k}_{\parallel}$ or $-(\boldsymbol{k}_{\parallel}+\boldsymbol{g}')$. Subroutine `adref2t` calculates the LEED amplitudes leaving the surface as well as those traveling into the crystal to meet with the second layer. For the primary beam $\boldsymbol{k}_{\parallel}$, the outgoing amplitudes $A^{-}_{g'}$ are written to the output files *layfile* (subroutine `outxist`) and the intensities $I(E)$ are stored in fd.out (subroutine `rint`). The remaining part of the program generates the angular momentum amplitudes $A_{i;lm}(\boldsymbol{k}_{\parallel})$ and $A_{i;lm}(-(\boldsymbol{k}_{\parallel}+\boldsymbol{g}'))$ for each atomic site i in the unit cell, moving back into the crystal layer by layer. For each layer, plane-wave amplitudes incident on it from below and onto the next lower layer from above are obtained according to Eqs. (22) and (23) (subroutine `multamp`). For each sublayer, Eq. (24) (subroutine `final1`) or its composite-layer equivalent, Eq. (26) (subroutine `finalov`) is evaluated, and the result is written to the files *layfile* specified in the input (subroutine `outamp`). The calculation then continues with the next incident beam or the next energy step. During the entire run, the standard output channel is used to monitor the progress of the calculation, and also to log possible warning or error messages.

3.2. Calculating amplitude changes

Setting up the calculation

Once the wave fields $A_{i;lm}(\boldsymbol{k}_{\parallel})$ and $A_{i;lm}(-(\boldsymbol{k}_{\parallel}+\boldsymbol{g}'))$ incident onto a lattice site i have been calculated full dynamically and stored in the appropriate files, amplitude changes $\delta\tilde{A}^{-}_{i,g'}$ due to modifications of the t-matrix t_i can be calculated with comparatively little effort.

The program delta.f allows to calculate amplitude changes for a particular chemical element on a particular lattice site i of the crystal. Its standard input is summarized in Table 3. As in the reference calculation, a title is chosen, and the energy range for the calculation is set, bounded by *ei* and *ef*. Both may differ from the bounds of the reference calculation – in that case, amplitude changes are only calculated for those energy steps of the reference calculation situated between the current *ei* and *ef*. Next, the bulk unit vectors *ara1* and *ara2* of the substrate lattice and the superlattice unit vectors *arb1* and *arb2* are read; all four vectors must be identical to those of the reference calculation. The same holds for the angles of incidence, *theta* and *fi*, the flag *formin* (which determined whether the input tensor components will be read in binary or ascii form, according to the flag *iform* of the reference calculation), and the list of beams *pqfex* to be approximated using Tensor LEED. The following input list of scattering phase shifts *phss* may in principle differ from that used in the reference calculation although the input format stays the same.

The remaining part of the input is specific to lattice site i for which amplitude changes are calculated. Flag *formout* determines whether these amplitude changes should be written in binary or ascii form. The chemical element (marked by its number in the phase shift list, *iel*) assumed on site i is defined next. In order to account for geomeric displacements of the atom relative to its position in the *reference surface*, *ncstep* geometry displacement steps *cdisp* are expected (using the undisplaced position as their reference point). Note that, as in the reference calculation, atomic coordinates are expected in the order (z, x, y) with z denoting the axis perpendicular to the

Table 3
Input data needed for the calculation of amplitude changes through structural modifications (program delta.f). For a given element occupying the lattice site in question, geometrical and vibrational modifications may be specified

Quantity	Format	Explanation
title	A80	Descriptive title
ei, ef	2F7.2	Amplitude changes are calculated only between *ei* and *ef* (may differ from energy range of reference calculation) [eV]
ara1	2F7.4	bulk unit mesh as in reference calculation [Å]
ara2	2F7.4	
arb1	2F7.4	superstructure unit mesh as in reference
arb2	2F7.4	calculation [Å]
theta, fi	2F7.4	polar and azimuthal angles of incidence as in reference calculation [°]
formin	I3	0: Tensor input in binary format 1: Tensor input in ascii format
The following line is expected for each beam to be handled by Tensor LEED:		
pqfex	2F7.4	(2D) reciprocal lattice vector as in "beam list"
nel	I3	number of chemical elements for which phase shifts are provided
The following block is expected for each energy step value in phase shift table:		
es	F7.4	current energy step value [Hartrees]
The following input is expected *nel* times:		
phss	10F7.4	scattering phase shifts $\delta_l(es)$, $l = 0, \ldots, l_{\max}$
formout	I3	0: binary output of $\delta\tilde{A}^{-}_{i,g'}$ 1: ascii output of $\delta\tilde{A}^{-}_{i,g'}$
iel	I3	number of that element in phase shift list occupying the current site for the current calculation of amplitude changes
cundisp	3F7.4	undisplaced position of current site with respect to its position in the reference surface (not currently supported, set to zero) [Å]
ncstep	I3	number of geometrical displacement steps to calculate amplitude changes for
The following line is expected *ncstep* times:		
cdisp	3F7.4	geometrical displacement step (dz, dx, dy) with respect to *cundisp* [Å]
ndeb	I3	number of thermal vibration amplitude steps to calculate LEED amplitude changes for
The following line is expected *ndeb* times:		
drper_a	F7.4	thermal vibration amplitude step [Å]

surface, and *negative* z values pointing towards the vacuum. Last follows a list of *ndeb* isotropic thermal vibration amplitudes *drper_a*. Note that LEED amplitude changes are calculated for each possible combination of input thermal vibration amplitudes and input geometrical displacements.

Except for its standard input, the program also expects an input file AMP that contains the tensor components $A_{i;lm}(\boldsymbol{k}_\parallel)$ and $A_{i;lm}(-(\boldsymbol{k}_\parallel + \boldsymbol{g}'))$. This file should be a simple copy of the tensor output file for the current lattice site i as written by the reference calculation. Aside from the tensor components, it contains all necessary information not included in the standard input – in particular, the inner potential, the unperturbed atomic t-matrix of the site in question, and the full dynamic amplitudes $A^-_{g'}$ leaving the *reference surface* are obtained from this file. Last, as for the reference calculation, certain static array dimensions consistent with the input must be set by the user. To that end, a file PARAM is created by the controlling shell script, to be compiled together with the source code of the program.

The program structure

Fig. 3 summarizes the structure of the program delta.f. After reading the general input data as defined in the standard input (subroutine `readbas`) and preparing the output file DELWV (subroutine `outdelt_dwg`), the program calculates Gaunt coefficients to be used for the later calculation (subroutines `belmg` and `cppp` as described in Refs. [11,21] where `belmg` plays the role of subroutine `gaunt` of Ref. [21]). The remainder of the calculation is repeated for each energy step of the full dynamic reference calculation recorded in the input file AMP. The energy loop is initialised by reading the first part of the information contained in AMP, i.e. the energy itself, the inner potential, the unperturbed LEED amplitudes leaving the *reference surface*, and the unperturbed t-matrix t_i of the current lattice site i (subroutine `indata`). Next, subroutine `inamp` reads the tensor components $A_{i;lm}(\boldsymbol{k}_\parallel)$ and $A_{i;lm}(-(\boldsymbol{k}_\parallel + \boldsymbol{g}'))$ for all beams $\boldsymbol{g}'$.

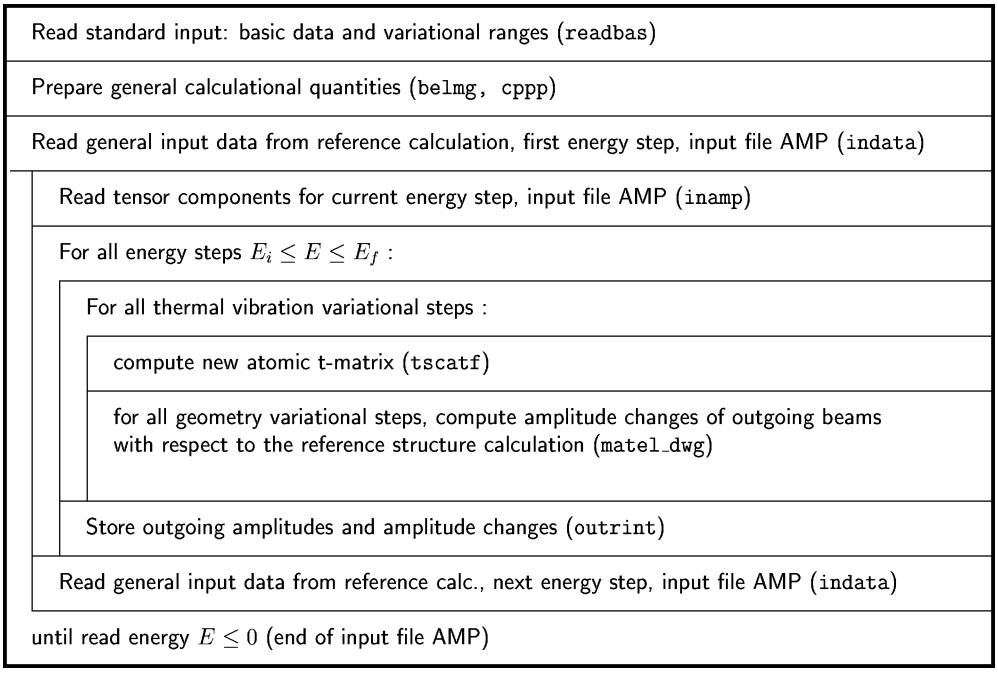

Fig. 3. Structogram of delta.f, the program that calculates amplitude changes that result from modified scattering properties of a given lattice site i using the Tensor LEED approximation.

If the current energy value lies between the energy boundaries *ei* and *ef* as defined in the input, amplitude changes are calculated for all variational steps requested in the input. To that end, a loop over the different thermal vibration amplitude grid steps is performed. Subroutine `tscatf` calculates the perturbed but not yet geometrically shifted t-matrix of the scatterer in question, defined by its chemical nature (element *iel*) and the current vibrational displacement. For each thermal vibration amplitude, subroutine `matel_dwg` then performs a loop over all geometric displacement steps *cdisp*, calculating the change in LEED amplitudes, $\delta\tilde{A}^{-}_{i,g'}$, for each beam $\boldsymbol{g}'$. Since subroutine `matel_dwg` is very close to stage 5 of the package TLEED2 as described by Rous and Pendry [22], we do not repeat any further details concerning its structure, referring the interested reader to the original work instead. The last part of the energy loop consists of storing the calculated amplitude changes in the output file DELWV (subroutine `outrint`). The program terminates after processing the last energy value recorded in input file AMP. As for the reference calculation, the progress of the entire run and possible error messages are monitored at the standard output channel.

3.3. Calculating $I(E)$ spectra

Using the output of delta.f of the last section, it is a computationally simple task to evaluate Eqs. (31) and (33) in order to obtain LEED $I(E)$ spectra for any model surface within the accessible parameter space. The program that performs this step, superpos.f, allows to calculate $I(E)$ spectra for many different combinations of structural parameters at a time, e.g., in order to perform a grid search for the *best fit* between model and experimental $I(E)$ spectra later. Since, at present, this functionality is largely obsolete through the structural search algorithm accompanying the package, we will not give an in-depth description of superpos.f here, but merely provide the information required to generate a single $I(E)$ spectrum for a given combination of structural parameters from the amplitude changes calculated by delta.f. The program's full functionality, if needed, may be understood from the comments included in the header of the source file itself.

Setting up the calculation

Table 4 contains a description of all quantities expected by superpos.f as its standard input. In order to evaluate Eqs. (31) and (33), it is necessary to provide *nfiles* different input files *infile* as written by delta.f, one for each chemical element of each lattice site to be varied. Furthermore, Eq. (31) also requires occupation probabilities $\tilde{x}_i$, i.e. one weighting factor *conc* for each input file must be given. Those files that correspond to the topmost lattice site of the *reference surface* should be marked by setting the flag *nsurf*=1. The flag *iform* specifies whether the data in the input file are provided in binary (*iform*=0) or ascii (*iform*=1) format. Each file *infile* as written by delta.f will, in general, contain sets of amplitude changes calculated for different combinations of the structural parameters associated with one lattice site, stored there in a fixed sequence. The specific parameter combination to be used in the current $I(E)$ calculation is specified by its position in *infile*, *ifnum*. For the surface structure thus defined, superpos.f calculates an $I(E)$ spectrum and stores it in output file INTENS. The accompanying output file DOC records the positions *ifnum* in each input file *infile*.

3.4. A global search algorithm

The remaining task of the calculation is the structural analysis of the particular surface under investigation. The program search.f simultaneously optimizes all variational parameters accessible through Tensor LEED and allows to restrict that parameter space as desired.

Setting up the calculation

search.f requires a variety of input files, listed in the following:

- input files containing Tensor LEED amplitude changes $\delta\tilde{A}^{-}_{i,g'}$ for each lattice site i and each chemical element occupying that site as written by delta.f,

Table 4
Input data required to calculate $I(E)$ data (program superpos.f) using amplitude changes stored previously. In principle, spectra may be generated for an arbitrary number of different combinations of structural parameters

Quantity	Format	Explanation
nfiles	2I3	number of different input files *infile* (one for each chemical element on each lattice site)
varall		0: if more than one input file is given, calculate $I(E)$ spectra using the $\delta\tilde{A}$ associated with *ifnum*(1) for each file, then those for *ifnum*(2) for each file, and so on. 1: calculate $I(E)$ spectra in loops, using the set *ifnum* for the first file as the outermost loop, *ifnum* for the second file inside that loop, and so on.
nconcs	I3	number of grid steps for chemical TLEED
The following line is expected *nconcs* times:		
conc	10F7.4	concentration factors for each input file *infile*
The following block is expected *nfiles* times:		
infile	A10, 3I3	name of current $\delta\tilde{A}$ input file
ifvar		number of variational steps to be used from current file *infile*
nsurf		1 if *infile* concerns the topmost lattice site, 0 otherwise
iform		0: *infile* contains binary data 1: *infile* contains ascii data
ifnum	27I3	identifiers of the *ifvar* sets of $\delta\tilde{A}$ in current file *infile* for which $I(E)$ spectra are calculated: *ifnum*$(j) = k$ means that, for $I(E)$ spectrum no. j, parameter combination no. k as stored in *infile* is used.

- the file search.steu containing all control parameters for the search, including a description of the accessible parameter space in terms of the amplitude changes calculated previously,
- and the standard input rf.info containing details on the R-factor calculation and the experimental $I(E)$ data to be compared with the calculated spectra.

The input files generated by delta.f, of course, must be provided as required for the parameter space in question. Details on the properties of these files are read from file search.steu, the general part of which is detailed in Table 5. The first two quantities affect the width assigned to the Gaussian probability distribution (PD) used during the search. *npar* is the number of free parameters in the search, not double-counting any coupled quantities. *rmut* is a global scaling factor by which the PD width is multiplied in order to allow to scale the locality of the search as a whole. The next point concerns the choice of random numbers for the search procedure. Standard random number generators need an initial seed, after which a predetermined sequence of pseudo-random numbers is generated. If the quantity *init* is set to zero, the seed used is the current system time, giving a different sequence of numbers in each run. Otherwise, the random seed is simply *init* itself so that the search procedure may be reproduced (e.g., for debugging purposes). For the optimization, two different R-factors may be used for experiment-theory comparison, either Pendry's R-factor R_P [39] or the R_2 R-factor as described in Ref. [1]. Furthermore, the optimization may be performed for one of two subsets of the beams (defined by *mittel* in rf.info, see below), e.g., integer or fractional-

Table 5
General settings for the optimization algorithm (program search.f) as specified in the head part of the input file search.steu

Quantity	Format	Explanation
npar	I5	number of parameters to be optimized independently
rmut	F7.4	global scaling factor for the width of the gaussian probability distribution used in the search
init	I5	seed value for the random number generator 0: use system time 1–99999: use this value
whichr	I5	1: use Pendry R-factor R_P 2: use R_2
whichg	I5	0: use R-factor average of all beams 1: use R-factor average of beam group 1 2: use R-factor average of beam group 2
outint	I5	number of search generations after which output is forced
maxgen	I5	maximum number of search generations (unlimited search if *maxgen*=0)
dmisch	I5	step width for incoherent average [%]
seaname	A10	name of search output file (use SD.TL)

order beams only, instead of the overall average R-factor, selected by flag *whichg*. Output is written to the file called *seaname* whenever an improvement occurs from one generation to the next, or, routinely, every *outint* generations. After a certain number of generations *maxgen*, the search is terminated.

The remaining quantities in search.steu, summarised in Table 6, concern the structure of the parameter space made available by the previously calculated Tensor LEED amplitude changes. The surface may consist of more than one long-range ordered domain in which case the $I(E)$ spectra from different surface areas must be averaged over using an area fraction step width *dmisch*. For each of the *ndom* different domains, detailed information is provided in a separate block.

For the *nplaces* lattice sites in the unit cell of each domain, *nfil* different input files containing amplitude changes (as written by delta.f) are expected, one for each chemical element to be considered. A name *infile* is expected for each file, the contents of which may be written in ascii or binary format (flag *iform*), and which may contain amplitude changes calculated for *partyp* different variational parameters (varied in loops; e.g., thermal vibrations and geometric displacements). The number of different steps, *varst*, considered for each parameter is expected in the following lines, starting with the parameter that belongs to the "outermost" variational loop. Positional and possible symmetry properties of each site are handled by *nsurf* and *filrel*. *nconcs* different variational steps (at least one) for the occupation probabilities *conc* of the *nfil* different elements on the site in question must be provided to evaluate Eq. (31). Once this input has been given for each domain in the model surface, the only remaining question concerns the parameter values for the first search generation. These are generated randomly when *stafla* is set to zero. Otherwise, the settings of the input parameters are taken from search.steu, listed in the format written to the output file control.chem in each search generation, allowing to restart a previous search run.

All details concerning the experiment-theory R-factor calculation are read from the standard input, rf.info (Table 7). In particular, the energy range for the optimization may be restricted to energies between *emin* and

Table 6
Input data for search.f (main part of input file search.steu) that describe the model surface and its free parameters in terms of the output files produced by the preceding calculation of amplitude changes

Quantity	Format	Explanation
ndom	I5	number of incoherent domains on the surface
The following block is expected *ndom* times:		
nplaces	I5	number of lattice sites addressed by Tensor LEED in current domain
The following block is expected *nplaces* times:		
nsurf	I3	set 1 for the lattice site at the very surface of the crystal, 0 otherwise
filrel	I3	when equal to *filrel* of another lattice site, force identical parameter values for both sites in the search
nfil	I3	number of elements on current site for which δA input files are provided
The following block is expected *nfil* times:		
infile	A15	file name for current element on current site
iform	I3	0: file contains binary data 1: file contains ascii data
partyp	I3	number of different parameters for which δA are stored in current file in a loop
The following line is expected *partyp* times:		
varst	I3	number of variation steps for current parameter, beginning with the outermost loop
nconc	I3	number of chemical concentration grid steps for current site
The following line is expected *nconc* times:		
conc	5F7.4	concentration values for each element on current site for current concentration grid step (their sum must equal 1 !)
stafla	I3	0: begin search with random trial structures 1: take initial trial structures from input
If *stafla*=1, the following line is expected for each independent trial structure:		
parind(nprmk)	500I3	initial parameter step number for each variational parameter of current trial structure

emax on a grid with step width *eincr* onto which intensities are interpolated. For debugging purposes, the amount of data written to the standard output channel may be regulated (*ipr*). As to the inner potential, its imaginary part *vi* (to calculate R_P) and real part *v0rr* should be given, as well as a range *v01* to *v02* between which the theoretical $I(E)$ data will be shifted with respect to the experimental energy scale in steps of *vincr*, thus effectively treating the energy-independent part of the inner potential as a variational parameter. To reduce possible residual noise in

Table 7
Input data controlling the R-factor calculation between calculated and measured $I(E)$ curves by the structural optimization algorithm search.f (input file rf.info)

Quantity	Format	Explanation
emin *emax* *eincr*	F7.2 F7.2 F7.2	Energy range and grid for which R-factor is calculated. Between *emin* and *emax*, $I(E)$ data are interpolated onto a grid with step width *eincr* [eV]
ipr	I3	0, 1, 2: Determines amount of output to stdout
vi	F7.4	Imaginary part of inner potential for R_P. Use average over entire energy range [eV]
v0rr	F7.4	Real part of inner potential (offset only) [eV]
v01 *v02* *vincr*	F7.4 F7.4 F7.4	To vary inner potential, theoretical spectra are shifted with respect to the experimental energy axis from *v01* to *v02* in steps of *vincr* [eV]
ismoth	I3	If necessary due to noise, smooth experimental data *ismoth* times.
eot	I3	0: Use true experimental data 1: Use calculated "pseudo-experimental" $I(E)$ data in Van Hove/Tong style format
nth, nex	2I3	Number of calculated and experimental beams, respectively
kav	25I3	Averaging scheme for theoretical intensities. $kav(k) = j$ means that calculated beam no. k belongs to beam group no. j.
mittel	25I3	grouping of experimental beams into two different sets for R-factor averaging
ibp	25I3	Compare theoretical beam group number $ibp(i)$ with experimental beam number i
wb	25F3.1	Weight of experimental beam for R-factor averaging
If *eot*=1, simply append a calculated $I(E)$ output file here. If *eot*=0, append experimental $I(E)$ data as outlined below:		
text	A80	description of experimental data
nbea	25I3	averaging scheme for experimental $I(E)$ data
fmt	A19	Fortran style input format for $I(E)$ data
The following block is expected *nex* times:		
bename	A76	name of current beam
nee, fac	I4, E13.4	number of measured $I(E)$ data points and normalisation factor (if any), respectively
ee, ae	*fmt*	$I(E)$ data points *ae(ee)*

the experimental data, the program offers the option to smooth the data *ismoth* times using a three-point formula. Instead of using true experimental $I(E)$ data (*eot*=0) to optimize the theoretical model structure against, it is also possible to compare to pseudo-experimental (i.e., calculated) $I(E)$ data (*eot*=1).

The next part of rf.info concerns the handling of the individual beams in the R-factor averaging procedure. *nth* and *nex* denote the number of available experimental and calculated beams, respectively. In cases where a surface has lower symmetry than the underlying bulk, the measured $I(E)$ curves are the average of several beams from symmetrically equivalent domains. To mimic this behaviour, calculated $I(E)$ data from different beams may be averaged prior to determining the R-factor (*kav*). Next, for reconstructed surfaces, apart from the overall average R-factor that of different beam sets, e.g., integer and fractional beams, should also be considered. The program currently allows to group the given beams into two different sets (*mittel*). The order of comparison between measured and calculated beams in general may be modified using *ibp*, and the weight of individual beams in the average R-factor may be adjusted using *wb*.

The remaining part of rf.info contains the experimental data against which the model surface is optimized. If pseudo-experimental data are used (*eot*=1), it is sufficient to append an $I(E)$ data file formatted as those produced by ref-calc.f or superpos.f. For experimental data (described by *title*), an additional scheme *nbea* for averaging the measured $I(E)$ spectra prior to the R-factor calculation is expected. The $I(E)$ data for each of the *nex* beams are included using an arbitrary Fortran style format *fmt*. For each beam, an identifier *bename*, the number of data points *nee* and a normalisation factor *fac* applied to that particular beam after measurement are expected prior to the actual $I(E)$ data.

Apart from the input files read at run-time, the search procedure requires two more modifications that directly affect the source code. First, certain static array dimensions must be specifically adapted by the user (include file PARAM) in agreement with the run-time input. The second modification concerns possible restrictions of the accessible parameter space, e.g., to force different chemical elements on the same lattice site onto the same geometric position. In order to allow for the greatest possible generality, such restrictions are performed in a dedicated subroutine restrict.f. The set of variational parameters is stored in an array *PARIND*(*IPARAM*, *IPOP*) for each independent trial structure *IPOP* considered in the search. Thus, any mathematical expression coupling two (or more) parameters *IPARAM* will serve to reduce the degrees of freedom in the search as desired.

The program structure

Fig. 4 summarizes the structure of search.f. After reading all input data from rf.info (`readrf` and `reade` or `readt`), search.steu (`readsc`) and the stored amplitude changes (`readfile`), the program first calculates Pendry's Y-function [39] for the experimental $I(E)$ data (`preexp`) and verifies the consistency of the input data (checkval). Next, the initial parameter values for each independent trial structure in the search are randomly selected by subroutine `sea_rcd` unless otherwise indicated by *stafla*.

In the following, these trial structures are simultaneously optimized step by step until the preset maximum number of search generations has been reached. After meeting the parameter space restrictions of subroutine `restrict`, the amplitude changes $\delta\tilde{A}^{-}_{i,g'}$ corresponding to the current search parameters are selected (`getgrid`), $I(E)$ spectra are calculated for each domain in the surface (`getint`) and averaged over all domains (`mischung`) weighted by their respective surface area fractions. Subroutine `rfaktor` then calculates either the Pendry R-factor R_P or the R_2 R-factor, providing the R-factor average over all beams or that over one beam set only for optimization. The procedure is repeated for each trial structure. Subsequently, all changes between the last and the current search generation are evaluated (`getwid`) to estimate the sensitivity of each variational parameter and set the width of the Gaussian PD for the next step accordingly. Following this, the variational parameters are reset to their previous values for those trial structures that did not improve their R-factor. If an improvement occurred, all trial structures are sorted according to their current R-factor (`order`) and output is written to the file *seaname* (`outrf`) as specified in search.steu. The last action in a search generation is the random selection of variational parameter grid points for the next generation of each trial structure (`sea_rcd`).

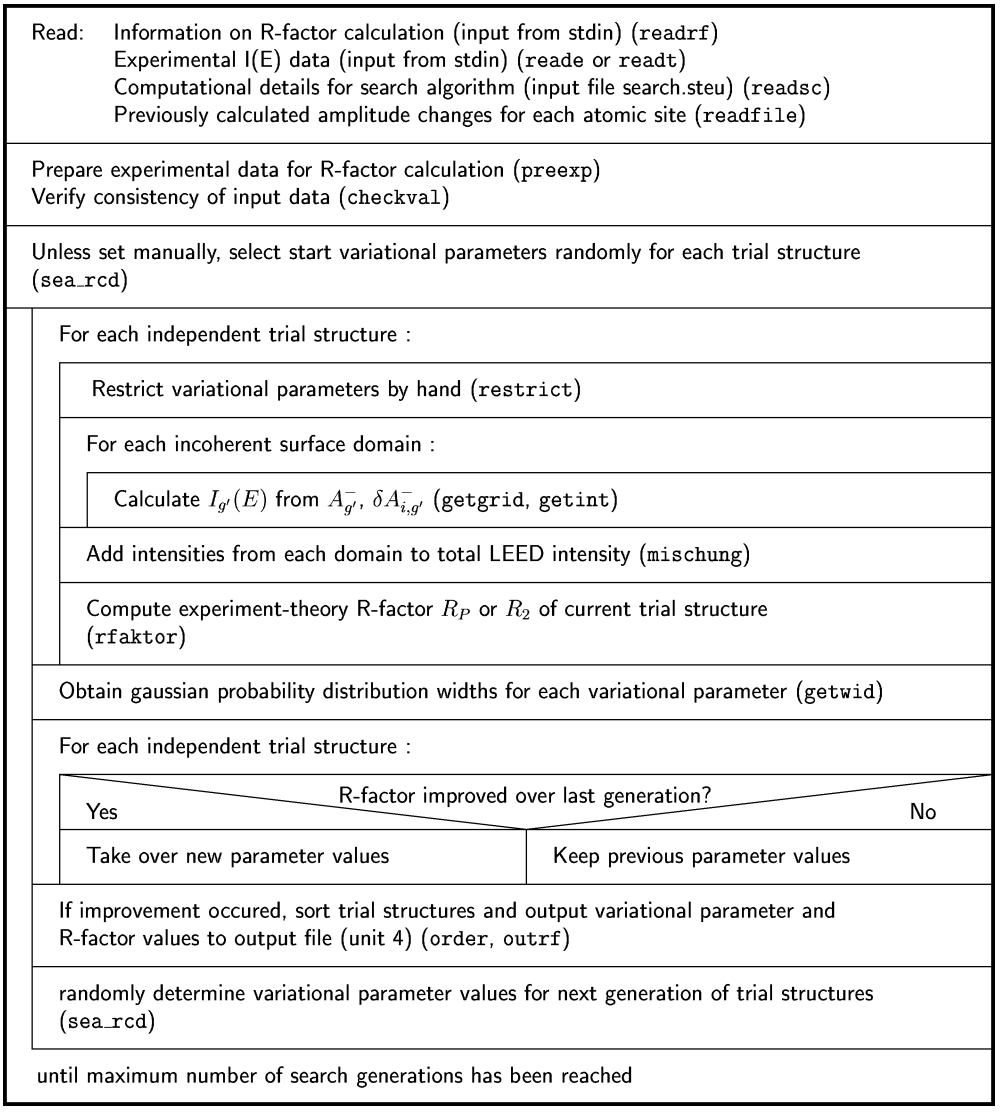

Fig. 4. Structogram of the optimization algorithm search.f that identifies the best fit model surface parameters compared to experimental $I(E)$ data.

As mentioned, in the current implementation these values are set using a Gaussian PD centered about the current parameter value. However, it is important to point out that any other iterative search scheme may be implemented in the current program version by simply using that algorithm in subroutine `sea_rcd` instead of the current scheme.

4. Organisation of the package

The TensErLEED package is distributed as a tar archive. Upon extraction, it creates a top level directory TLEED/ containing the entire package.

The subdirectory TLEED/v1.2/ holds the current versions of the programs described above. It is itself divided into a subdirectory TLEED/v1.2/src/ and a subdirectory TLEED/v1.2/lib/. The source files ref-calc.f, delta.f, search.f and superpos.f are located in src/, along with an include file GLOBAL used by all programs. GLOBAL houses fundamental constants such as atomic units needed in the calculation to ensure consistency between different steps of the calculation. lib/ contains all subroutines called by the main programs.

Each step of the calculation is performed by a separate UNIX shell script that needs to be adapted by the user. The tasks performed by these shell scripts are

- the creation of all input data files,
- the adjustment of array dimensions for the source code,
- the creation of specific subroutines which depend on the particular problem to be solved,
- the compilation of the entire source code,
- to run the actual calculation,
- storage of all relevant output, and
- to possibly initiate a successive calculation.

Other than through these shell scripts, no interaction of the user with the source code is needed except when implementing new features. Before using each script, it must be adapted to the user's platform. In particular, this concerns the FORTRAN77 compiler name along with its options, e.g., for code optimisation. Also, all pathnames and filenames defined in the header of each shell script should be set correctly, most importantly the *$TLEED* variable specifying the absolute location of the TLEED/ top directory in the user's UNIX directory tree.

All UNIX shell scripts for the test run outlined in Section 5 are located in subdirectory TLEED/testrun/, along with a series of subdirectories for the purpose of storing intermediate output between the separate steps of the calculation. All calculations are performed using the subdirectory TLEED/work/ which is automatically cleaned after a run is complete. Prior to conducting the test run, three kinds of input data must be supplied separately. Experimental $I(E)$ data to compare calculated ones against are placed in subdirectory TLEED/exp-data/ in the format required for input file rf.info (Table 7). It contains a sample file of experimental data for use with the test run. Files containing scattering phase shifts in the appropriate input format should be placed in the subdirectory TLEED/phaseshifts/. In the original archive, only those phase shifts are included that concern the test run, i.e. phase shifts for iron and aluminium (generated for an ordered FeAl compound of B2 structure). Phase shifts for other materials may be obtained from the authors by request, or may be generated by standard packages available in the community (e.g., Refs. [23,38]). Furthermore, ref-calc.f requires a list of reciprocal lattice vectors appropriate for the particular surface unit mesh under investigation. Such lists should be housed in the subdirectory TLEED/beamlists/, and may be generated using the accompanying utility beamgen.f. Subdirectory TLEED/aux/ contains this utility (in TLEED/aux/beamgen/) along with a program that calculates R-factors between experimental data and calculated $I(E)$ output in the format supplied by ref-calc.f or superpos.f (in TLEED/aux/r-factor/), a program that may be used to reformat and normalise $I(E)$ output files for plotting purposes (in TLEED/aux/plot), and some helpful UNIX commands (in TLEED/aux/bin/). For a detailed description of these utilities, see the readme file provided together with the package.

5. Test run

The test run provided with TensErLEED optimizes the structure of a bcc $Fe_{0.97}Al_{0.03}$(100)-c(2×2) alloy surface using experimental $I(E)$ data. A detailed description of this surface including a full LEED intensity analysis is given in Ref. [40]. In contrast, the surface model used here is a simplified one, depicted in Fig. 5. The structural parameters of the *reference surface* and *best fit* structures are given in Table 8. For the former, an ordered c(2×2) array of Fe and Al atoms in the topmost layer and bulklike geometry and composition of the underlying layers is assumed. Starting from there, the positions of all symmetry-inequivalent lattice sites in the topmost three layers are varied, along with the chemical composition of the second and third layer and independent thermal vibrational

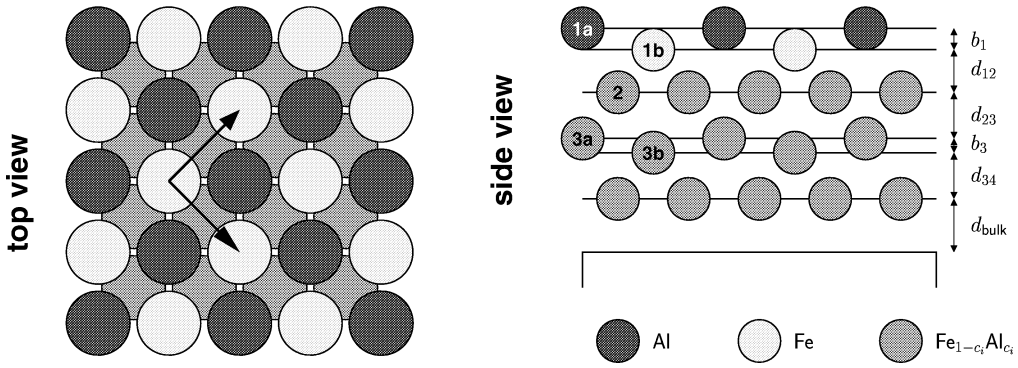

Fig. 5. Schematic illustration of the structural model for the $Fe_{0.97}Al_{0.03}(100)$-c(2×2) surface investigated in the test run. The topmost layer consists of an ordered array of Fe and Al atoms while all lower layers contain only a randomly distributed fraction of Al. Parameters for the reference and best fit structures can be found in Table 8.

Table 8
Structural parameters of the *reference structure* and the *best fit* model of the test run provided with the TensErLEED package

Parameter	Reference structure	Best fit value
Geometry		
b_1	0.00 Å	0.06 Å
d_{12}	1.4325 Å	1.3725 Å
d_{23}	1.4325 Å	1.4525 Å
b_3	0.00 Å	0.02 Å
d_{34}	1.4325 Å	1.4325 Å
$a_{\text{bcc}} = 2 \cdot d_{\text{bulk}}$	2.865 Å (not varied)	
Stoichiometry		
c_{1a}	Al (not varied)	
c_{1b}	Fe (not varied)	
c_2	3% Al	0% Al
c_{3a}	3% Al	5% Al
c_{3b}	3% Al	15% Al
Thermal vibrations		
v_1(Al)	0.06 Å	0.10 Å
v_1(Fe)	0.06 Å	0.12 Å
v_2	0.06 Å	0.10 Å
v_{bulk}	0.06 Å (not varied)	
Inner potential		
V_{00}	0 eV	−1.5 eV
$V_{0r}(E)$	$V_{00} + \max\left(-10.3, 0.39 - \frac{76.63}{\sqrt{(E/\text{eV})+9.68}}\right)$ eV (not varied)	
V_{0i}	5 eV (not varied)	

amplitudes in the top and second layer. The present structural analysis uses experimental $I(E)$ data for incident energies up to 300 eV, so that a maximum angular momentum value of $l_{\max} = 7$ is sufficient for those parts of the calculation using a spherical wave basis. On a 500 MHz Compaq XP1000 workstation, the CPU time required to perform the entire test run lies below 1 hour. Important parts of the output are attached to this paper. Note that, since our search scheme chooses subsequent trial structures randomly, the output for the final search generation will vary between different runs, but the correct *best fit* parameters should be identified in all cases.

Acknowledgements

The full dynamic parts of these codes are derived from the program package published by M.A. Van Hove (Berkeley) and S.Y. Tong (Milwaukee), and the implementation of geometrical Tensor LEED is based on programs originally developed by P.J. Rous (now Baltimore) and J.B. Pendry (London). We wish to thank these authors for making their work available to us. Additional thanks are due to Robert Koller (Vienna) for assisting us in testing the current code version.

References

[1] M.A. Van Hove, W.H. Weinberg, C.-M. Chan, Low Energy Electron Diffraction, Springer, Berlin, 1986.
[2] K. Heinz, Rep. Prog. Phys. 58 (1995) 637.
[3] M.A. Van Hove, Surf. Rev. Lett. 4 (1997) 479.
[4] K. Heinz, L. Hammer, Z. Kristallogr. 213 (1998) 615.
[5] M. Arnold, A. Fahmi, W. Frie, L. Hammer, K. Heinz, J. Phys. C 11 (1999) 1873.
[6] M. Kottcke, H. Graupner, D.M. Zehner, L. Hammer, K. Heinz, Phys. Rev. B 54 (1996) R5275.
[7] S. Müller, P. Bayer, C. Reischl, K. Heinz, B. Feldmann, H. Zillgen, M. Wuttig, Phys. Rev. Lett. 74 (1995) 765.
[8] K. Heinz, P. Bayer, S. Müller, Surf. Rev. Lett. 2 (1995) 89.
[9] A. Seubert, J. Schardt, W. Weiß, U. Starke, K. Heinz, Appl. Phys. Lett. 76 (2000) 727.
[10] U. Starke, J. Schardt, J. Bernhardt, M. Franke, K. Reuter, H. Wedler, K. Heinz, J. Furthmüller, P. Käckell, F. Bechstedt, Phys. Rev. Lett. 80 (1998) 758.
[11] M.A. Van Hove, S.Y. Tong, Surface Crystallography by LEED, Springer, Berlin, 1979.
[12] P.J. Rous, J.B. Pendry, D.K. Saldin, K. Heinz, K. Müller, N. Bickel, Phys. Rev. Lett. 57 (1986) 2951.
[13] P.J. Rous, J.B. Pendry, Surf. Sci. 219 (1989) 355.
[14] P.J. Rous, Prog. Surf. Sci. 39 (1992) 3.
[15] U. Löffler, R. Döll, K. Heinz, J.B. Pendry, Surf. Sci. 301 (1994) 346.
[16] R. Döll, M. Kottcke, K. Heinz, Phys. Rev. B 48 (1993) 1973.
[17] K. Heinz, R. Döll, M. Kottcke, Surf. Rev. Lett. 3 (1996) 1651.
[18] M. Kottcke, K. Heinz, Surf. Sci. 376 (1997) 352.
[19] C.J. Davisson, L.H. Germer, Nature 119 (1927) 558.
[20] P.R. Watson, M.A. Van Hove, K. Hermann, NIST Surface Structure Database – Ver. 3.0, National Institute of Standards and Technology, Gaithersburg, MD, 1999.
[21] P.J. Rous, J.B. Pendry, Comput. Phys. Commun. 54 (1989) 137.
[22] P.J. Rous, J.B. Pendry, Comput. Phys. Commun. 54 (1989) 157.
[23] A. Barbieri, M.A. Van Hove, private communication (http://electron.lbl.gov/leedpack/).
[24] P.G. Cowell, V.E. de Carvalho, Surf. Sci. 187 (1987) 175.
[25] P.J. Rous, M.A. Van Hove, G.A. Somorjai, Surf. Sci. 226 (1990) 15.
[26] G. Kleinle, W. Moritz, D.L. Adams, G. Ertl, Surf. Sci. 219 (1989) L637.
[27] G. Kleinle, W. Moritz, G. Ertl, Surf. Sci. 238 (1990) 119.
[28] H. Over, U. Ketterl, W. Moritz, G. Ertl, Phys. Rev. B 46 (1992) 15438.
[29] P.J. Rous, Surf. Sci. 296 (1993) 358.
[30] R. Döll, M.A. Van Hove, Surf. Sci. 355 (1996) L393.
[31] J.B. Pendry, Low Energy Electron Diffraction, Academic Press, London, 1974.
[32] D.E. Bilhorn, L.L. Foldy, R.M. Thaler, W. Tobocman, V.A. Madsen, J. Math. Phys. 5 (1964) 435.
[33] J.L. Beeby, J. Phys. C 1 (1968) 82.

[34] S.Y. Tong, M.A. Van Hove, Phys. Rev. B 16 (1977) 1459.
[35] R. Baudoing, Y. Gauthier, M. Lundberg, J. Rundgren, J. Phys. C 19 (1986) 2825.
[36] U. Löffler, U. Muschiol, P. Bayer, K. Heinz, V. Fritzsche, J.B. Pendry, Surf. Sci. 331–333 (1995) 1435.
[37] O. Rubner, M. Kottcke, K. Heinz, Surf. Sci. 340 (1995) 172.
[38] J. Rundgren, private communication (1999).
[39] J.B. Pendry, J. Phys. C 13 (1980) 937.
[40] V. Blum, L. Hammer, W. Meier, K. Heinz, M. Schmid, E. Lundgren, P. Varga, in preparation (2000).

TEST RUN OUTPUT

Output file spec.ref-calc.Fe0.97Al0.03.100-c2x2.basic
(Full dynamic $I(E)$ output file fd.out of ref-calc.f)

```
Fe0.97Al0.03(100)-c(2x2), basic reference calculation
  8
  2   1.00000   0.00000  1
  6   1.00000   1.00000  1
 10   2.00000   0.00000  1
 14   2.00000   1.00000  1
 22   2.00000   2.00000  1
 82   0.50000   0.50000  1
 86   1.50000   0.50000  1
 94   1.50000   1.50000  1
  20.00 0.0001   0.60252E-03   0.00000E+00   0.00000E+00   0.00000E+00
   0.00000E+00   0.10761E-02   0.00000E+00   0.00000E+00
  23.00 0.0001   0.11853E-02   0.00000E+00   0.00000E+00   0.00000E+00
   0.00000E+00   0.18388E-02   0.00000E+00   0.00000E+00
  26.00 0.0001   0.20287E-02   0.00000E+00   0.00000E+00   0.00000E+00
   0.00000E+00   0.24374E-02   0.00000E+00   0.00000E+00
  29.00 0.0001   0.22349E-02   0.00000E+00   0.00000E+00   0.00000E+00
   0.00000E+00   0.27452E-02   0.00000E+00   0.00000E+00

 .....

 281.00 0.0001   0.10978E-02   0.47314E-03   0.21251E-03   0.45205E-03
   0.35821E-04   0.59011E-03   0.77880E-03   0.56914E-03
 284.00 0.0001   0.88586E-03   0.13296E-02   0.23948E-03   0.32433E-03
   0.72054E-04   0.55573E-03   0.84076E-03   0.61062E-03
 287.00 0.0001   0.73728E-03   0.29910E-02   0.22459E-03   0.37369E-03
   0.12289E-03   0.61685E-03   0.87539E-03   0.58207E-03
 290.00 0.0001   0.70683E-03   0.55589E-02   0.13507E-03   0.49014E-03
   0.17932E-03   0.68009E-03   0.91669E-03   0.55565E-03
 293.00 0.0001   0.77274E-03   0.85930E-02   0.33764E-04   0.53266E-03
   0.23166E-03   0.69069E-03   0.91260E-03   0.59751E-03
 296.00 0.0001   0.82757E-03   0.11197E-01   0.49129E-04   0.44644E-03
   0.25584E-03   0.66996E-03   0.81772E-03   0.65100E-03
 299.00 0.0001   0.82297E-03   0.12265E-01   0.28702E-03   0.28508E-03
   0.23316E-03   0.63549E-03   0.67518E-03   0.63989E-03
 302.00 0.0001   0.78320E-03   0.11396E-01   0.76361E-03   0.12418E-03
   0.21344E-03   0.63177E-03   0.56603E-03   0.60680E-03
```

Output file search-doc.Fe0.97Al0.03.100-c2x2.960K.1198.geo1-3.chem2-3.vib1-2
(Documentation file *seaname* of search.f. After completion of the test run, the input file search.steu and its control output file control.chem are appended to this file. "\" denotes linebreaks included for legibility that are not part of the actual output file.)

```
 R-factor minimum search using RPe for all beams for optimization.

 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
 CCCCCCCCCCC    GENERATION            1    CCCCCCCCCCC
 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC


IND ||  R-Av  |  R-In  | R-Ha   |   V0r   || Proz | P11,P11,C11,P21,P21,\
C22,P31,P31,P32,P32,C33,P41,P41,P42,P42,C44,P51,P52,C55,P61,P62,C66,
-------------------------------------------------------------------------\
----------------------------
  1 || 0.5286 | 0.5098 | 0.5577 |  -8.300 || 100% |   4   8   1   5   5 \
  1   3   4   3   4   6   3   4   3   4   6   7   7   2   7   7  10
  2 || 0.5648 | 0.4852 | 0.6869 | -14.800 || 100% |   2  18   1   3  13 \
  1   4   5   4   5   5   4   5   4   5   5  10  10   1   6   6   6
  3 || 0.5912 | 0.5517 | 0.6741 | -15.300 || 100% |   2  18   1   4  19 \
  1   3  11   3  11  10   3  11   3  11  10   7   7   6   8   8   4
  4 || 0.6030 | 0.4359 | 0.8491 | -15.300 || 100% |   3  19   1   2  16 \
  1   3   6   3   6   2   3   6   3   6   2   9   9   4   3   3   4
  5 || 0.6497 | 0.7039 | 0.5609 | -15.300 || 100% |   3  17   1   2  10 \
  1   3   8   3   8  10   3   8   3   8  10   8   8   9   4   4   8
  6 || 0.6800 | 0.6296 | 0.7647 | -14.300 || 100% |   1  18   1   2  14 \
  1   4   9   4   9   3   4   9   4   9   3   2   2  10   9   9   7
  7 || 0.7337 | 0.6597 | 0.8568 | -14.300 || 100% |   1  20   1   2  12 \
  1   1   8   1   8  10   1   8   1   8  10   7   7   3  10  10   5
  8 || 0.7444 | 0.6314 | 0.9144 | -11.300 || 100% |   5  18   1   1  15 \
  1   4   3   4   3   2   4   3   4   3   2   2   2   2  11  11   7
  9 || 0.7698 | 0.6720 | 0.9614 |  -9.800 || 100% |   4   9   1   5   8 \
  1   4   6   4   6   5   4   6   4   6   5   6   6  11  10  10   6
 10 || 0.7939 | 0.6543 | 1.0100 | -13.800 || 100% |   1  20   1   5  10 \
  1   5   2   5   2   9   5   2   5   2   9   7   7   2   9   9   4
 11 || 0.8130 | 0.9323 | 0.6103 |  -6.300 || 100% |   3   7   1   5  10 \
  1   3   9   3   9  11   3   9   3   9  11  10  10   5   3   3  11
 12 || 0.8175 | 0.7894 | 0.8622 | -11.800 || 100% |   5  13   1   1  21 \
  1   1   8   1   8   1   1   8   1   8   1   3   3   1   8   8  11
 13 || 0.8202 | 0.8653 | 0.7273 | -11.800 || 100% |   3  11   1   4  16 \
  1   3   7   3   7   8   3   7   3   7   8   4   4  10   7   7   6
 14 || 0.8227 | 0.5856 | 1.1723 | -10.300 || 100% |   4  16   1   2   3 \
  1   2   5   2   5   5   2   5   2   5   5   2   2   6   1   1  10
 15 || 0.8264 | 0.7649 | 0.9188 | -15.300 || 100% |   3  19   1   5   4 \
  1   3   8   3   8   7   3   8   3   8   7  11  11   6  10  10   2
 16 || 0.8422 | 0.9508 | 0.6719 |  -5.300 || 100% |   1   3   1   5   3 \
  1   3  10   3  10   9   3  10   3  10   9   8   8   1   1   1  11
 17 || 0.8447 | 0.8532 | 0.8273 |  -5.300 || 100% |   1   5   1   5   6 \
  1   2   4   2   4  11   2   4   2   4  11   7   7  10   6   6   6
 18 || 0.8642 | 0.7648 | 1.0208 | -15.300 || 100% |   1  20   1   3  15 \
```

```
  1   3   8   3   8   4   3   8   3   8   4   1   1   6   2   2   3
 19 || 0.8872 | 0.8124 | 1.0103 | -11.300 || 100% |   1  12   1   5  21 \
  1   1   6   1   6  10   1   6   1   6  10   6   6   1   8   8   6
 20 || 0.8974 | 0.9737 | 0.7817 |  -5.300 || 100% |   1   8   1   1  18 \
  1   5   4   5   4   9   5   4   5   4   9   9   9   1   9   9   1
 21 || 0.9023 | 0.8091 | 1.0529 | -15.300 || 100% |   1   4   1   5  13 \
  1   5   7   5   7   5   5   7   5   7   5  10  10   2  11  11   2
 22 || 0.9311 | 1.0059 | 0.8109 |  -9.800 || 100% |   3   2   1   4  11 \
  1   5   1   5   1   1   5   1   5   1   1   3   3   9   5   5  10
 23 || 0.9461 | 1.0131 | 0.8487 |  -5.300 || 100% |   2   5   1   3  17 \
  1   1   7   1   7   9   1   7   1   7   9   8   8   1   5   5   1
 24 || 0.9777 | 0.9329 | 1.0521 | -12.300 || 100% |   5   7   1   4  16 \
  1   2   6   2   6   7   2   6   2   6   7   5   5   1  11  11   3
 25 || 0.9886 | 1.1272 | 0.7942 | -12.300 || 100% |   2   3   1   1  19 \
  1   5   4   5   4   3   5   4   5   4   3   4   4   6   9   9   5

Average RPe of GENERATION      1 :  0.79361266

 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
 CCCCCCCCCCC    GENERATION            2    CCCCCCCCCCC
 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

 .....

 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
 CCCCCCCCCCC    GENERATION        10000    CCCCCCCCCCC
 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC


IND ||  R-Av  |  R-In  | R-Ha   |   V0r   || Proz | P11,P11,C11,P21,P21,\
C22,P31,P31,P32,P32,C33,P41,P41,P42,P42,C44,P51,P52,C55,P61,P62,C66,
-------------------------------------------------------------------------\
----------------------------
  1 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
  2 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
  3 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
  4 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
  5 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
  6 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
  7 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
  8 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
  9 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
 10 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
```

```
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
 11 || 0.0999 | 0.1021 | 0.0960 | -11.800 || 100% |   4  12   1   3   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
 12 || 0.1016 | 0.0967 | 0.1104 | -11.300 || 100% |   2  11   1   1   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
 13 || 0.1016 | 0.0967 | 0.1104 | -11.300 || 100% |   2  11   1   1   9 \
  1   3   4   3   4   1   3   4   3   4   1   6   6   4   5   5   2
 14 || 0.1119 | 0.1154 | 0.1056 | -12.800 || 100% |   3  13   1   3  11 \
  1   3   5   3   5   1   3   5   3   5   1   6   6   4   6   6   1
 15 || 0.1119 | 0.1154 | 0.1056 | -12.800 || 100% |   3  13   1   3  11 \
  1   3   5   3   5   1   3   5   3   5   1   6   6   4   6   6   1
 16 || 0.1119 | 0.1154 | 0.1056 | -12.800 || 100% |   3  13   1   3  11 \
  1   3   5   3   5   1   3   5   3   5   1   6   6   4   6   6   1
 17 || 0.1119 | 0.1154 | 0.1056 | -12.800 || 100% |   3  13   1   3  11 \
  1   3   5   3   5   1   3   5   3   5   1   6   6   4   6   6   1
 18 || 0.1140 | 0.1166 | 0.1094 | -12.300 || 100% |   3  13   1   3  10 \
  1   3   5   3   5   1   3   5   3   5   1   6   6   3   6   6   1
 19 || 0.1263 | 0.1202 | 0.1370 | -10.800 || 100% |   3  10   1   1   8 \
  1   2   3   2   3   1   2   3   2   3   1   5   5   3   4   4   2
 20 || 0.1263 | 0.1202 | 0.1370 | -10.800 || 100% |   3  10   1   1   8 \
  1   2   3   2   3   1   2   3   2   3   1   5   5   3   4   4   2
 21 || 0.1263 | 0.1202 | 0.1370 | -10.800 || 100% |   3  10   1   1   8 \
  1   2   3   2   3   1   2   3   2   3   1   5   5   3   4   4   2
 22 || 0.1263 | 0.1202 | 0.1370 | -10.800 || 100% |   3  10   1   1   8 \
  1   2   3   2   3   1   2   3   2   3   1   5   5   3   4   4   2
 23 || 0.1298 | 0.1236 | 0.1404 | -10.300 || 100% |   3  10   1   2   7 \
  1   2   3   2   3   1   2   3   2   3   1   5   5   3   4   4   1
 24 || 0.1509 | 0.1589 | 0.1365 | -12.800 || 100% |   3  14   1   3  12 \
  1   4   6   4   6   1   4   6   4   6   1   7   7   4   7   7   1
 25 || 0.1509 | 0.1589 | 0.1365 | -12.800 || 100% |   3  14   1   3  12 \
  1   4   6   4   6   1   4   6   4   6   1   7   7   4   7   7   1

Average RPe of GENERATION  10000 :  0.11199822

 .....
```

Output file spec.tleed.best-fit.geo1-3.chem2-3.vib1-2
($I(E)$ output file INTENS of superpos.f for the best fit structural parameters identified by the preceding search run.)

```
DEL.11.Fe DEL.12.Al DEL.21.Al DEL.21.Fe DEL.22.Al DEL.22.Fe DEL.31.Al \
DEL.31.Fe DEL.32.Al DEL.32.Fe
  8
  1   1.00000   0.00000  1
  2   1.00000   1.00000  1
  3   2.00000   0.00000  1
  4   2.00000   1.00000  1
  5   2.00000   2.00000  1
  6   0.50000   0.50000  1
  7   1.50000   0.50000  1
  8   1.50000   1.50000  1
  20.00 0.0001   0.67359E-03   0.00000E+00   0.00000E+00   0.00000E+00
   0.00000E+00   0.12544E-02   0.00000E+00   0.00000E+00
  23.00 0.0001   0.10903E-02   0.00000E+00   0.00000E+00   0.00000E+00
   0.00000E+00   0.23228E-02   0.00000E+00   0.00000E+00
  26.00 0.0001   0.16920E-02   0.00000E+00   0.00000E+00   0.00000E+00
   0.00000E+00   0.31474E-02   0.00000E+00   0.00000E+00
  29.00 0.0001   0.17998E-02   0.00000E+00   0.00000E+00   0.00000E+00
   0.00000E+00   0.35077E-02   0.00000E+00   0.00000E+00

 .....

 281.00 0.0001   0.15634E-02   0.65928E-03   0.24327E-03   0.11017E-03
   0.22544E-03   0.17365E-03   0.32051E-03   0.23353E-03
 284.00 0.0001   0.13495E-02   0.13519E-02   0.17302E-03   0.13690E-03
   0.31677E-03   0.77112E-04   0.34347E-03   0.26993E-03
 287.00 0.0001   0.12825E-02   0.24958E-02   0.11966E-03   0.27493E-03
   0.42347E-03   0.21555E-04   0.37194E-03   0.26641E-03
 290.00 0.0001   0.13880E-02   0.40686E-02   0.15027E-03   0.41328E-03
   0.51678E-03   0.15085E-05   0.39632E-03   0.27480E-03
 293.00 0.0001   0.15909E-02   0.57874E-02   0.37450E-03   0.43033E-03
   0.55960E-03   0.32817E-05   0.38405E-03   0.33844E-03
 296.00 0.0001   0.16960E-02   0.72240E-02   0.78973E-03   0.32420E-03
   0.52579E-03   0.22185E-04   0.34150E-03   0.43099E-03
 299.00 0.0001   0.16337E-02   0.78063E-02   0.12282E-02   0.18028E-03
   0.43283E-03   0.53333E-04   0.28433E-03   0.46272E-03
 302.00 0.0001   0.14829E-02   0.72280E-02   0.16642E-02   0.67527E-04
   0.32989E-03   0.10636E-03   0.21546E-03   0.42457E-03
```