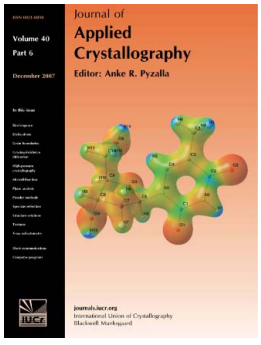
Journal of
Applied Crystallography
ISSN 0021-8898
Editor: **Anke R. Pyzalla**

Fitting dynamical X-ray diffraction data over the World Wide Web

Sergey Stepanov and Rebecca Forrest

J. Appl. Cryst. (2008). **41**, 958–962



Journal of
Applied Crystallography
ISSN 0021-8898

Received 29 April 2008
Accepted 15 July 2008

Fitting dynamical X-ray diffraction data over the World Wide Web

Sergey Stepanov[a]* and Rebecca Forrest[b]

[a]Argonne National Laboratory, 9700 S. Cass Avenue, Argonne, IL, USA, and [b]University of Houston, 4800 Calhoun Road, Houston, TX 77004, USA. Correspondence e-mail: sstepanov@anl.gov

The first implementation of fitting X-ray Bragg diffraction profiles from strained multilayer crystals at a remote web-based X-ray software server is presented. The algorithms and the software solutions involved in the process are described. The suggested technology can be applied to a wide range of scientific research and has the potential to promote remote collaborations across scientific communities.

1. Introduction

Ongoing intensification of scientific research in all fields and the complexity of physical models inevitably lead to a certain specialization in science; it often becomes difficult (or inefficient in terms of consumed time) for a single researcher or a small group to combine experimentation with the derivation and programming of equations for the numerical analysis of their data. These tendencies have led to the emergence of various software aggregation projects aimed at collecting data analysis software and making it available to experimentalists. In the X-ray and neutron diffraction fields, some well established software projects are CCP4 (1979), CXRO (1984), *XOP* (ESRF, 1996) and DANSE (2004). X-Ray Server (XRS, 1997; Stepanov, 2004*a*,*b*), another popular source of X-ray software, differs from the above projects in some of its goals and consequently in the chosen approaches. The major difference is that XRS targets researching how a theorist could directly share his/her software with the community without loosing a link to users and their feedback. Since physical models typically contain multiple approximations valid for a certain range of applications, there is a danger when an additional layer of software engineers is introduced between the original author of physical modeling software and the end user (DANSE), or when such software is given away (*XOP*). The problem is that restrictions of physical models may not be obvious from the beginning or may not be built in to the software because it originally targeted a narrower range of applications. Thus, when the developer has no way to monitor usage, end users, not knowing the restrictions, may derive incorrect conclusions as a result of misusing software beyond its range. This is a serious problem uncovered many times at XRS, where a way to monitor and correct misuse was built in from the beginning. It should be noted that a need for monitoring, being crucially important for physical modeling, is much less of an issue for scientific tools like data visualization software, scientific calculators or computer programs based on well established models, such as the crystallographic software included in the CCP4.

An ideal solution to the problem of miscommunication with end users would be a theorist and a software engineer working closely as a team. Unfortunately, much of science is produced in small groups, such as at universities, where resources are limited and funding for software engineers to help scientists is not realistically available. In addition, collaboration with remote software developers sponsored by some national or international projects (*e.g.* DANSE) may be difficult to realize because of the amount of required communication.

To achieve the goal of keeping a link to end users, software in the XRS project is not given away but is instead supplied with a small wrapper allowing online usage through a personal web server that is easy to monitor on a regular basis. Thus, when anything goes wrong, the user can be notified and the software may be corrected promptly. Additional benefits of this technology compared with providing software for download are that users always deploy the latest, most refined version and there is no need for installation on or porting of software to different operating systems.

At present, XRS can perform online modeling of dynamical X-ray diffraction from strained multilayers (Stepanov *et al.*, 1998), X-ray scattering from interface roughness (Kaganer *et al.*, 1995; Kondrashkina *et al.*, 1997), X-ray reflectivity (Stepanov *et al.*, 1998), resonant magnetic scattering (Stepanov & Sinha, 2000) and multiple Bragg diffraction (Stepanov & Ulyanenkov, 1994), as well as calculation of X-ray structure factors (Lugovskaya & Stepanov, 1991). With ten years of experience (the project started in 1997) and more than 300 000 calculations performed on requests from about 5000 users, XRS can claim to be a mature project providing well tested, reliable software. However, for a long time access to XRS programs was only available through web browsers, inevitably restricting applications such as massive modeling and data fitting. The recent addition of automation templates (Stepanov, 2007) has helped to overcome that long-standing restriction, and in this paper we report on the first successful case of fitting Bragg diffraction data from strained AlSb/AlAs superlattices *via* the World Wide Web (WWW).

The paper is organized as follows: §2 provides a short summary of the dynamical Bragg diffraction modeling program presented through XRS with the focus on the changes introduced to it after Stepanov *et al.* (1998). §3 outlines the wrapper technology for making scientific software available on the WWW through XRS and the templates for non-web browser access. §4 describes the programming needed on the user's side to implement fitting at XRS. §5 presents fitting experience and results. In *Conclusions*, some suggestions are made concerning how the proposed WWW techniques could be deployed in other areas of science.

2. Dynamical diffraction modeling program, GID_{sl}

The dynamical diffraction modeling program presented through XRS and named GID_{sl} (grazing incidence diffraction from superlattices) is based on the work presented by Stepanov *et al.* (1998) and in several

preceding publications (Stepanov, 1994; Stepanov & Koehler, 1994*a*,*b*; Stepanov *et al.*, 1995). As the program name states, originally it was designed for GID. However, for the X-ray server operations, GID_{sl} was extended to an arbitrary geometry of Bragg-case diffraction (Fig. 1). The program can calculate Bragg diffraction profiles for a crystal with a stack of surface layers, each layer having its own structure susceptibilities χ_0 and χ_h and lattice strain $\Delta a/a$ in the direction perpendicular to the substrate.

The program works as follows: Firstly, the extended dynamical equations for a perfect crystal plate are solved in every layer and the layers' reflection and transmission coefficients are found. The term 'extended' means that a non-approximated fourth-power dispersion equation of the dynamical diffraction is applied, and the boundary equations take into account X-ray refraction and specular reflection at interfaces. Secondly, the overall reflection from the stack of layers is calculated with the help of a recursive matrix algorithm for 2×2 scattering matrices (Stepanov *et al.*, 1998). The layers in the program can be either real layers of a multilayer structure or imaginary slices approximating material density and strain profiles in a crystal. Stepanov *et al.* (1998) explain why the dynamical diffraction equations work for even very thin layers. The program has easy input for periodic groups of layers, making it simple to describe superlattices, and it accounts for interface roughness following the model of Stepanov & Koehler (1994*b*). The structure factors χ_0 and χ_h of each layer can be either specified implicitly or calculated automatically using a database of crystal structures as described by Lugovskaya & Stepanov (1991). The structures database is expandable through submissions from XRS users. It is also possible to specify χ_0 and χ_h *via* scale factors with respect to known structures or to present layers as a mix of several materials (*e.g.* 60% of AlAs and 40% of GaAs).

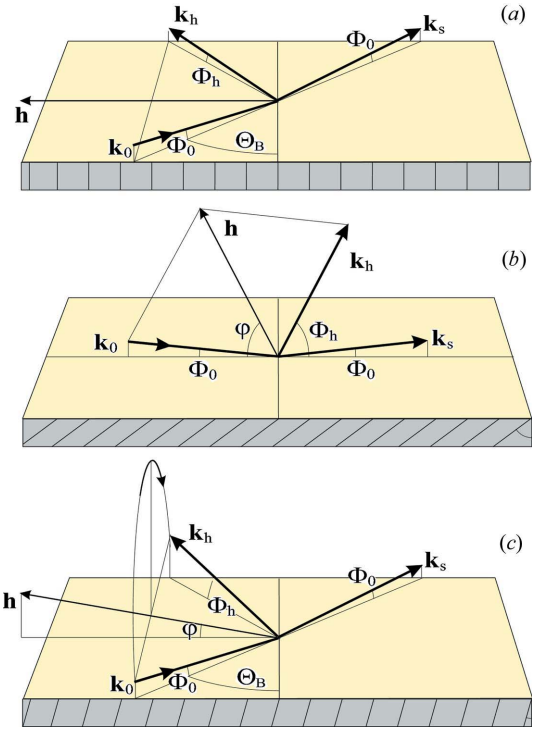


Figure 1
Diffraction geometries supported by the GID_{sl} program: (*a*) grazing incidence diffraction, (*b*) coplanar asymmetric diffraction including extremely asymmetric cases and (*c*) noncoplanar asymmetric diffraction when the scattering plane does not contain the surface normal.

Although GID_{sl} is quite general, it still has certain limitations, the most notable of which is an inability to model lateral strains that would result in misfit dislocations or crystal bending. The program also cannot calculate transitions between different Bragg reflections, as in the models of Holy & Fewster (2003) and Podorov *et al.* (2006). Still, a study by Grundmann & Krost (2000) demonstrated that the performance and accuracy of GID_{sl} for symmetric Bragg diffraction is comparable to that of commercial software available in the field. That study did not, however, include the cases of grazing incidence and/or exit where GID_{sl} had no analogues.

After GID_{sl} began serving requests *via* XRS, the following most notable improvements were made as a result of monitoring the program usage and direct requests from users:

(1) Enhanced program input for easy construction of noncoplanar Bragg geometries on the one hand (Fig. 1*c*) and simplified specification of more traditional coplanar geometries on the other hand (Fig. 1*b*).

(2) Support for wider deviations from the Bragg condition and ways to define scans around an arbitrary axis in a crystal.

(3) Extension of calculations to noncubic crystals.

(4) Establishment of limitations imposed by the diffraction model and the numerical algorithm applied in the program, and addition of validation of all input parameters against a reasonable range of values.

The first two items above deserve a detailed discussion because they are of general interest to the X-ray diffraction field.

2.1. Construction of the Bragg geometry for coplanar and noncoplanar diffraction

The first step for any GID_{sl} calculation is to construct the complex diffraction geometries presented in Fig. 1. This is implemented using algebraic operations with reciprocal lattice vectors expressed by their Miller indices in crystals of arbitrary symmetry, from cubic to triclinic (Sirotin & Shaskolskaya, 1982). The input parameters for GID_{sl} are the Miller indices of vector $\mathbf{h}$ corresponding to the Bragg planes, the X-ray wavelength or energy giving the length k of the incident wavevector $\mathbf{k}_0$, and the Miller indices of the surface normal $\mathbf{n}$. [Actually, to simplify the inputting of miscut surfaces, $\mathbf{n}$ is specified by its angular deviation, φ_n, from some reciprocal lattice vector $\mathbf{n}_0$ along another reciprocal lattice vector $\mathbf{n}_1$, *i.e.* $\mathbf{n} = x_0\mathbf{n}_0 + x_1\mathbf{n}_1$, $(\mathbf{n}\cdot\mathbf{n}_0) = \cos(\varphi_n)$, where both $\mathbf{n}_0$ and $\mathbf{n}_1$ are specified by their Miller indices; however, this does not change the general approach to constructing the geometry.] As illustrated by Fig. 1(*c*), with $\mathbf{h}$ and k known, the incident vector $\mathbf{k}_0$ lies on a cone around $\mathbf{h}$ given by the Bragg law $2k\sin\Theta_B = h$ and one more condition is needed to fully determine the direction of $\mathbf{k}_0$. For this purpose, GID_{sl} suggests the input of one of the following parameters:

(1) Incidence angle Φ_0 of $\mathbf{k}_0$ providing $\gamma_0 = \sin\Phi_0 = (\mathbf{k}_0\cdot\mathbf{n})/k$.

(2) Exit angle Φ_h of $\mathbf{k}_h = \mathbf{k}_0 + \mathbf{h}$ providing $\gamma_h = \sin\Phi_h = (\mathbf{k}_h\cdot\mathbf{n})/k$.

(3) Requirement of coplanar diffraction geometry so that vector $\mathbf{k}_0$ must reside in the same plane as $\mathbf{h}$ and $\mathbf{n}$ (see Fig. 1*b*). This may give two solutions for $\mathbf{k}_0$, so an additional condition of either grazing incidence, $\Phi_0 < \Phi_h$ (Fig. 1*b*), or grazing exit, $\Phi_0 > \Phi_h$ (inverted Fig. 1*b*), is requested.

(4) Condition of the symmetric Bragg case, $\gamma_0 = |\gamma_h|$.

For all of these cases GID_{sl} presents $\mathbf{k}_0$ as

$$\mathbf{k}_0 = c_1 k\mathbf{n} + c_2\mathbf{h} + c_3(\mathbf{h}\times\mathbf{n}), \qquad (1)$$

where c_1, c_2 and c_3 are given by the three equations

$$(\mathbf{k}_0 \cdot \mathbf{n}) = k\gamma_0 = c_1 k + c_2(\mathbf{h} \cdot \mathbf{n}),$$
$$(\mathbf{k}_0 \cdot \mathbf{h}) = -kh \sin\Theta_B = c_1 k(\mathbf{h} \cdot \mathbf{n}) + c_2 h^2, \quad (2)$$
$$(\mathbf{k}_0 \cdot \mathbf{k}_0) = -k^2 = c_1^2 k^2 + c_2^2 h^2 + c_3^2(\mathbf{h} \times \mathbf{n})^2 + 2c_1 c_2(\mathbf{h} \cdot \mathbf{n}).$$

Cases 1 and 2 are solved by equations (2) directly; for case 3 the last two equations of (2) are combined with the condition $c_3 = 0$; and for case 4 it is easy to find that $\gamma_0 = -(\mathbf{h} \cdot \mathbf{n})/2k$ and then apply equations (2). In the special case when $\mathbf{h}||\mathbf{n}$, vector $\mathbf{k}_0$ is presented as $\mathbf{k}_0 = c_1 k\mathbf{n}_1 + c_2\mathbf{h}$, where $\mathbf{n}_1$ is a vector specifying the surface miscut direction (though the miscut angle may be zero). In this case the diffraction is always coplanar and symmetric, so any extra inputs from 1 to 4 are ignored.

Although *GID*$_{sl}$ offers such flexibility to specify complex diffraction geometries, monitoring of XRS usage revealed that many calculations are still restricted to coplanar schemes as in Fig. 1(*b*): obviously because coplanar experiments are simpler. Understanding this fact led us to design a simplified *GID*$_{sl}$ input for coplanar geometries, where the surface normal is not requested but is instead built internally. In the simplified case, only one input parameter is sufficient to define the geometry, in addition to the diffraction indices and the X-ray wavelength/energy:

(1) angle φ between the Bragg planes and the surface or
(2) incidence angle Φ_0 of $\mathbf{k}_0$ or
(3) exit angle Φ_h of $\mathbf{k}_h$ or
(4) asymmetry factor $\beta = \gamma_0/|\gamma_h|$.

At first, the above data are used to determine angle φ. Cases 2 and 3 give $\varphi = \Theta_B - \Phi_0$ and $\varphi = \Theta_B + \Phi_h$, respectively, and in case 4 one can find $\tan\varphi = \tan\Theta_B(\beta - 1)/(\beta + 1)$. Then, $\mathbf{n}$ is presented as $\mathbf{n} = x_1\mathbf{h} + x_2\mathbf{h}_t$, where $\mathbf{h}_t$ is an arbitrary vector nonparallel to $\mathbf{h}$. It is derived from $\mathbf{h}$ by inverting any non-zero index and adding 1 to another index, *e.g.* $(hkl) \rightarrow (\bar{h}, k + 1, l)$. Parameters x_1 and x_2 are calculated from the conditions $(\mathbf{n} \cdot \mathbf{h}) = h\cos\varphi$ and $|\mathbf{n}| = 1$. The rest of the calculations are performed according to equations (2).

2.2. Wider range of deviations from the Bragg condition

For constructing scans around arbitrary axes *GID*$_{sl}$ uses the same vector technology as for building diffraction geometries. It also helps to accommodate a larger range of deviations from the Bragg condition. Normally, the parameter α describing the deviation of incident X-ray wavevector $\mathbf{k}_0$ from the Bragg condition is approximated as follows:

$$\alpha = [(\mathbf{k}_0 + \mathbf{h})^2 - k^2]/k^2 = 2(\Delta\mathbf{k} \cdot \mathbf{h})/k^2 \simeq 2\sin(2\Theta_B)\Delta\Theta. \quad (3)$$

Here $\Delta\mathbf{k} = \mathbf{k}_0 - \mathbf{k}_B$ and $\mathbf{k}_B$ is the wavevector satisfying the exact Bragg condition, $\alpha = 0$, and $\Delta\Theta$ is the angle between $\mathbf{k}_0$ and $\mathbf{k}_B$. Instead of the approximation on the right-hand side of equation (3), *GID*$_{sl}$ makes use of the following exact equations. Let $\mathbf{a}$ be a unit vector along the scan axis. *GID*$_{sl}$ allows for specifying $\mathbf{a}$ along the surface normal $\mathbf{n}$, or vectors $[\mathbf{k}_0 \times \mathbf{h}]$ or $[\mathbf{k}_0 \times \mathbf{n}]$, or an arbitrary reciprocal lattice vector given by its Miller indices; however, in all of these cases $\mathbf{a}$ is internally represented as a set of the reciprocal lattice coordinates. Since $\Delta\mathbf{k}$ is perpendicular to $\mathbf{a}$, it can be written as

$$\Delta\mathbf{k} = x\mathbf{b} + y\mathbf{c},$$

where $\mathbf{c} = (\mathbf{k} \times \mathbf{a})$ and $\mathbf{b} = (\mathbf{a} \times \mathbf{c}) \equiv [\mathbf{a} \times (\mathbf{k}_0 \times \mathbf{a})] \equiv \mathbf{k}_0 - (\mathbf{k}_0 \cdot \mathbf{a})\,\mathbf{a}$ are the two vectors perpendicular to $\mathbf{a}$ and to each other. It is easy to find that

$$x = (b/c)\sin(\Delta\Theta) \quad \text{and} \quad y = -2\sin^2(\Delta\Theta/2).$$

Thus, the calculations are not restricted by the linear approximation over $\Delta\Theta$ as in equation (3).

Having calculated $\mathbf{k}_0$, *GID*$_{sl}$ also calculates the exact values of the γ_0 and γ_h parameters for each scan point:

$$\gamma_0 = (\mathbf{k}_B \cdot \mathbf{n}) + (\Delta\mathbf{k} \cdot \mathbf{n}), \quad \gamma_h^2 = \left[\gamma_0 + (\mathbf{h} \cdot \mathbf{n})\right]^2 - \alpha. \quad (4)$$

Use of the exact equations (4) is especially important for grazing incidence and exit geometries where the variations of γ_0 and γ_h at scans around the Bragg peak may be comparable to their initial values at the exact Bragg condition. A discussion of the dependency of γ_h on α presented by the second equation of (4) is given by Stepanov *et al.* (1998).

2.3. Validation of input parameters

While validation of input parameters seems to be a clear case, establishing restrictions for the input parameters has been an ongoing process and the greatest effort in the maintenance of XRS. Validation issues can be roughly classified into three groups. Firstly, scientific software often contains presumptions of 'reasonable' input. For example, no one would expect the requested scan range to be $\pm 2000^\circ$. However, the XRS experience shows that such requests do happen, so respective filters have been set up.

The second group is constituted by more difficult model limitations. For example, the interface roughness model (Stepanov & Koehler, 1994*b*) fails within the Bragg peak at asymmetric high-angle Bragg diffraction. Therefore, *GID*$_{sl}$ has been restricted to use that model either when $\max(\gamma_0, |\gamma_h|) \leq |\chi_0|^{1/2}$ (*i.e.* under the GID conditions for which the model was originally suggested) or at $\beta = 1$ (the symmetric Bragg diffraction condition). In all other cases users are advised to model interface roughness with transition layers. The same group includes the requirement to check that the requested scan range does not overlap with regions of other Bragg reflections. However, since *GID*$_{sl}$ has no way to check that, the decision on the applicability of the requested scan range is left to the user, while *GID*$_{sl}$ only contains a meaningful filter that the range must be within $\pm 45^\circ$.

The third group is related to numerical problems during computations. When some elements of scattering matrices are much larger than the others, such matrices become quasi-singular and the calculations may fail because of possible loss of precision. Typically this is related to large deviations from the Bragg conditions, $|\alpha| >> |\chi_h|$, or large strains in some layers, $|\Delta a/a| >> |\chi_h|$, but the problem may be intensified with multiple layers, very thick layers or weak reflections, $|\chi_h| << |\chi_0|$. Fortunately these cases are easily detectable, since they result in unreasonable nonphysical numbers for the reflectivity, $R >> 1$, and such a post-calculation validation filter is incorporated into *GID*$_{sl}$. Another protection mechanism is the incorporation of a threshold deviation α_{max} and the treatment of layers as amorphous if $|\alpha| > \alpha_{max}$ instead of trying to account for vanishingly small reflections from them. Typical recommended α_{max} values are $\alpha_{max} \cong 10^8|\chi_h|$. Another planned measure expected to lower the probability of precision loss is to stop accounting for specular reflection effects in cases of non-grazing Bragg diffraction. That should also speed up calculations for non-grazing geometries by a factor of four as a result of reducing 2×2 matrices to scalars (Stepanov *et al.*, 1998). Finally, it must be noted that the occurrence of precision loss in the history of XRS has been around 0.1%.

Some issues are a combination of model restrictions and numerical limitations. For example, *GID*$_{sl}$ cannot calculate forbidden reflections, $\chi_h = 0$. This is because the scattering matrix becomes singular and cannot be inverted. However, since the calculations are numeric, at some small $|\chi_h| << |\chi_0|$ the scattering matrix may also become singular because of the limited precision in the computations.

Therefore, GID_{sl} imposes the following empirically found filter for forbidden reflections:

$$|\chi_h| < \max\left[10^{-8}\sin(\Theta_B), 10^{-2}|\chi_0|\right]. \quad (5)$$

This filter additionally accounts for possible loss of precision in cases of hard X-rays when both χ_0 and χ_h may become more than eight orders of magnitude smaller than $\sin(\Theta_B)$. Condition (5) is checked against each requested X-ray polarization because the χ_h values for σ- and π-polarized X-rays are different, and when $\Theta_B \cong 45^\circ$ the calculations may be possible for σ-polarized X-rays only.

As mentioned above, establishing validation rules has been an ongoing process, and the stability and the success of the GID_{sl} program have been strongly dependent on the ability to monitor the program usage and make timely corrections to the code. The way GID_{sl} errors are delivered to the remote user is discussed at the end of the next section.

3. Layered wrappers for WWW access to scientific software

A technology for remote access to scientific modeling software proposed by Stepanov (2004*a*,*b*, 2007) is illustrated in Fig. 2. The initial mode is that software runs with input parameters from a local data file, as shown in Fig. 2(*a*). If the program is of interest to multiple parties, simply sharing the code may spawn multiple problems, including the need to port software to different operating systems or different compilers, thus branching off from the initial development. Even more important is the risk of potential misuse when the author loses the ability to monitor usage and refine input filters, as discussed in §2.3. Therefore, an HTTP wrapper technology has been suggested and implemented, as shown in Fig. 2(*b*). With this technology all users have access to only one instance of the program running at an HTTP server. They prepare input parameters by filling out an HTML form in their HTTP browsers, and the CGI wrapper on the HTTP server side converts submitted data into a local input file, starts the modeling program and ultimately makes the data available to the HTTP browser of the remote user. Finally, when one needs to automate remote access to the software, the HTTP browser can be replaced by a user-written HTTP agent program, which is another wrapper simulating the HTTP browser with the advantage that multiple modeling requests can be processed without user intervention. For example, Fig. 2(*c*) illustrates the agent called from experimental data fitting software, which may vary some parameters and submit simulation requests *via* the agent.

To facilitate usage of HTTP agents, XRS provides some self-explanatory example templates written in Perl and Unix shell scripting languages. Although those templates are ready to use, it should be noted that the agents could be easily implemented in most modern computing languages.

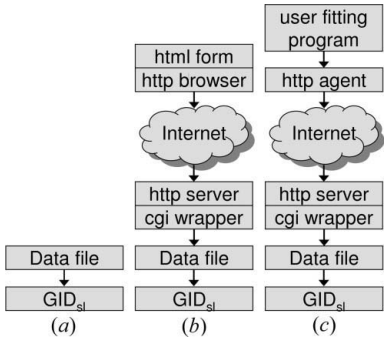


Figure 2
Three ways to access the GID_{sl} program: (*a*) directly on a local computer, (*b*) from a remote HTTP browser *via* the Internet, and (*c*) from a remote user fitting program *via* an HTTP agent and the Internet.

The example templates can also deliver errors from XRS to the user's software. In the case when the user submits an invalid input or another problem occurs, XRS returns an HTML page that contains a stop sign thumbnail (stop.gif) followed by an error message. The wrapper script inspects the page for the presence of this thumbnail. If stop.gif is found, the error message is distilled from the HTML page and saved in a log, and the script exit status is set to 'failure'. Then it is the responsibility of the user-side software to analyze the error log and take appropriate action.

We report the first successful case of fitting Bragg diffraction data with XRS *via* the WWW using this process.

4. User programming

We wished to fit the X-ray diffraction data from an AlSb/AlAs superlattice to determine its structure. Since the superlattice was about 1.0 µm thick, dynamical modeling was necessary within the fitting routine. We chose to model the data with XRS, called from within a local χ^2 simplex fitting program.

The structure to be fitted was an AlSb/AlAs superlattice acting as a buffer layer within a quantum well heterostructure, grown at the Naval Research Laboratory. The heterostructure consisted of 122 AlSb(25)/AlAs(2) 'digital alloy' superlattice buffer layer periods (∼10 000 Å thick), followed by five InAs(6)/$Ga_{0.7}In_{0.3}Sb$(11)/InAs(6)/$AlAs_{0.106}Sb_{0.894}$(77) 'W' quantum well periods (∼1524 Å thick), seven periods (∼234 Å) of the same AlSb/AlAs digital alloy barrier and a 150 Å GaSb cap layer. The numbers in parentheses are the numbers of monolayers. For further growth details see Canedy *et al.* (2003) and Boishin *et al.* (2006). The data to be fitted were radial scans through the 004 peaks performed on a four-circle diffractometer using Cu $K\alpha_1$ radiation.

Only the structure of the buffer layer was of scientific interest; therefore only parameters describing the buffer layer were varied within the fit. An approximate fit of the 'W' quantum well structure was performed separately and treated as background, being added to the simulated dynamical diffraction pattern of the buffer superlattice within the local fitting program. The simulated 'W' structure was InAs(6)/$Ga_{0.72}In_{0.28}Sb$(12)/InAs(6)/$AlAs_{0.11}Sb_{0.89}$(76).

In the buffer fitting process, we allowed for four layers: $AlAs_{x1}$-$Sb_{1-x1}(n1)$/$AlAs_{x2}Sb_{1-x2}(n2)$/$AlAs_{x3}Sb_{1-x3}(n3)$/$AlAs_{x4}Sb_{1-x4}(n4)$. Group V mixing was allowed in all layers, and the extra layers were included to simulate varying interdiffusion at the interfaces. The thicknesses of each layer, *ni*, and the compositions, *xi*, were variables in the fitting process.

The local fitting program was written in Fortran and run at the University of Houston. It is a standard fitting program that updates the fitting parameters and attempts to minimize χ^2 using a simplex algorithm. In addition to traditional actions, such as data input and parameter initialization, our program involves the following steps:

(1) The program creates a text file containing the crystal profile description to be simulated by XRS, in the appropriate syntax, based on the fitting parameters.

(2) Using the system call it executes a Perl wrapper script, with the profile text file as an argument.

(3) The wrapper script requests data from XRS with all of the appropriate settings for our sample and saves the calculated Bragg curve into a local file, also checking for any error codes.

(4) After the wrapper finishes its job and control returns to the fitting program, it reads the simulated data and log files, checks for

errors, convolutes the data with the appropriate resolution function, adds background, compares it with the measured data to find χ^2, and calculates the parameters for the next iteration.

This procedure is continued until stopped by the user when an acceptably small χ^2 is achieved. Our Perl wrapper script was based on the sample script 'getGID_useragent.pl', available at http://x-server.gmca.aps.anl.gov/automation.html. The sample script has sufficient comments that it is easy to adapt. Without any prior knowledge of Perl, the user was able to modify the script to read the profile text file and set the necessary XRS parameters.

5. Fitting results

The best fit to the sample is shown in Fig. 3. This fit corresponds to a three-layer superlattice buffer: $AlAs_{0.01}Sb_{0.99}(23.3)/AlAs_{0.34}Sb_{0.66}(1.4)/AlAs_{0.89}Sb_{0.11}(1.9)$. The χ^2 of this fit is 0.295. The average superlattice period is 80.9 Å, slightly less than the intended 82.4 Å. The middle layer represents interdiffusion between the intended AlSb and AlAs layers.

The overhead of remote fitting could be estimated as follows. For the given multilayer diffraction simulation GID_{sl} was consistently reporting 26 s per calculation (400 layers in the structure and 2401 data points per Bragg curve). This time does not include the program startup and only benchmarks the duration of the loop over the data points. In its turn, the XRS log shows that, when the fitting was running, 848 curves were calculated within 6 h and 49 min, which is 28.9 s per calculation. The later includes the XRS overhead together with the contributions from the client-side wrapper and the fitting routine. Therefore, we have about 10% overhead due to remote operation. Obviously the overhead time does not depend on the length of calculations, and the percentage of the overhead would be higher for shorter calculations and lower for lengthy calculations, but in any case we consider it a reasonable price for eliminating the need for the dynamical diffraction equations to be programed by the experimentalist.

6. Conclusions

We have described the first implementation of fitting X-ray Bragg diffraction profiles from strained multilayer crystals at a remote WWW-based X-ray software server. A practical task of fitting X-ray diffraction data from a complex semiconductor superlattice structure was solved with minimum programming on the side of the user, who otherwise would have needed to develop a complex dynamical diffraction program modeling a multilayer structure. The two authors had minimal interaction: they never met in person and never even communicated by telephone. The only interaction required was some minimal consultation on the details of interfacing with the XRS program and resolving certain network firewall issues, all done by email. Basically these were one-time investments that would be documented for future implementations.

This approach is a simple, inexpensive and efficient way to make well established software available to small research groups while maintaining feedback between software providers and their end users. This approach will not be appropriate for all applications, for example, those involving high demand or massive computations. We expect, however, that the experience reported in this paper will be valuable for a rapidly growing number of WWW collaborations in all areas of science.

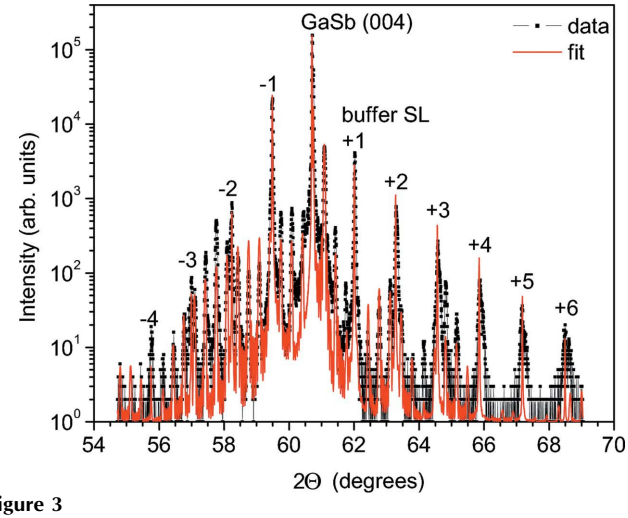


Figure 3
Radial X-ray diffraction scan through the 004 Bragg peak of the sample. Only the AlSb/AlAs superlattice (SL) buffer fit was optimized; its peaks are numbered. Unlabeled peaks are from the W quantum well and were only approximately simulated. Experimental data: black line with points; simulation: red line.

X-Ray Server uses computing resources of the GM/CA Collaborative Access Team at Argonne National Laboratory. GM/CA CAT has been financed with federal funds from the National Cancer Institute (Y1-CO-1020) and the National Institute of General Medical Science (Y1-GM-1104). The work at the University of Houston was supported in part by the M. Hildred Blewett Scholarship of the American Physical Society, http://www.aps.org. The authors are grateful to Chadwick Canedy and Jerry Meyer at the Naval Research Laboratory for supplying the sample.

References

Boishin, G. I., Canedy, C. L., Vurgaftman, I., Meyer, J. R. & Whitman, L. J. (2006). *J. Crystal Growth*, **286**, 32–36.
Canedy, C. L., Bewley, W. W., Kim, C. S., Kim, M., Vurgaftman, I. & Meyer, J. R. (2003). *J. Appl. Phys.* **94**, 1347–1355.
CCP4 (1979). *Collaborative Computational Project Number 4*, http://www.ccp4.ac.uk.
CXRO (1984). *Center for X-Ray Optics*, http://www-cxro.lbl.gov.
DANSE (2004). *Distributed Data Analysis for Neutron Scattering Experiments*, http://wiki.cacr.caltech.edu/danse.
ESRF (1996). *XOP*, *X-Ray Oriented Programs Package*, http://ftp.esrf.eu/pub/scisoft/.
Grundmann, M. & Krost, A. (2000). *Phys. Status Solidi. (b)*, **218**, 417–423.
Holy, V. & Fewster, P. F. (2003). *J. Phys. D Appl. Phys.* **36**, A5–A8.
Kaganer, V. M., Stepanov, S. A. & Koehler, R. (1995). *Phys. Rev. B*, **52**, 16369–16372.
Kondrashkina, E. A., Stepanov, S. A., Opitz, R., Schmidbauer, M., Koehler, R., Hey, R., Wassermeier, M. & Novikov, D. V. (1997). *Phys. Rev. B*, **56**, 10469–10482.
Lugovskaya, O. M. & Stepanov, S. A. (1991). *Sov. Phys. Crystallogr.* **36**, 478–481.
Podorov, S. G., Faleev, N. N., Pavlov, K. M., Paganin, D. M., Stepanov, S. A. & Förster, E. (2006). *J. Appl. Cryst.* **39**, 652–655.
Sirotin, Yu. I. & Shaskolskaya, M. P. (1982). *Fundamentals of Crystal Physics*, 2nd ed. Moscow: Mir Publishers.
Stepanov, S. A. (1994). *Crystallogr. Rep.* **39**, 182–187.
Stepanov, S. (2004*a*). *Proc. SPIE*, **5536**, 16–26.
Stepanov, S. (2004*b*). *Proc. SPIE*, **5536**, 165–170.
Stepanov, S. (2007). *Thin Solid Films*, **515**, 5700–5703.
Stepanov, S. A. & Koehler, R. (1994*a*). *J. Phys. D Appl. Phys.* **27**, 1922–1928.
Stepanov, S. A. & Koehler, R. (1994*b*). *J. Appl. Phys.* **76**, 7809–7815.
Stepanov, S. A., Kondrashkina, E. A., Koehler, R., Novikov, D. V., Materlik, G. & Durbin, S. M. (1998). *Phys. Rev. B*, **57**, 4829–4841.
Stepanov, S. A., Pietsch, U. & Baumbach, G. T. (1995). *Z. Phys. B*, **96**, 341–347.
Stepanov, S. A. & Sinha, S. K. (2000). *Phys. Rev. B*, **61**, 15302–15311.
Stepanov, S. A. & Ulyanenkov, A. P. (1994). *Acta Cryst.* A**50**, 579–585.
XRS (1997). *X-Ray Server*, http://x-server.gmca.aps.anl.gov.