X-Ray Diffraction Analysis of GaN/InGaN

Leon Barrett, James Buckley, Daniel Leopold, Patrick Gibbons

April 4, 2004

1 Introduction

Epitaxial semiconductor layers grown on a variety of single crystal substrates have created many opportunities for studying basic materials science properties of structures designed on a nanometer scale, as well as allowing novel quantum electronic and photonic devices to be fabricated. Some of the more interesting and useful properties of multilayered epitaxial structures occur when dissimilar materials having different bulk crystal structures and lattice constants are combined. However, heteroepitaxial growth of lattice mismatched systems forces a competition to take place between anisotropic strain and defect formation. As the strain energy builds with each additional monolayer, defect formation becomes more energetically favorable, relieving some of the built-in strain. The most common type of defect is a dislocation. This is the main reason why strained epitaxial films have a larger dislocation density closest to the epitaxial layer/substrate interface. Typically, as thicker layers are grown, the dislocation density is reduced.

One standard nondestructive method frequently used to characterize the crystalline quality of semiconductor epitaxial layers is x-ray diffraction. X-ray diffraction gives data of x-ray reflection intensity versus the angle at which the x-ray beam impacts the crystal. This contains peaks in intensity that indicate at what angles x-ray reflection occurs in phase. When using the common θ - 2θ diffraction systems, these peaks can be artificially broadened by intrinsic instrumental limitations. Thus it becomes quite important to remove, or at least take into account, any instrumental broadening effects when analyzing diffraction peaks in order to obtain accurate information on the crystalline structure. It is also important to properly incorporate some realistic model of the physical effects from defects and strain with any approach used to analyze x-ray diffraction peaks. The standard method for analyzing a diffraction peak is to fit it with a Gaussian, Lorentzian, or Voigt (a convolution of Gaussian and Lorentzian) function and then attempt to extract information from the function's parameters. However, this neglects some of the physics present in the crystal, and we attempt to remedy this.

We have developed a first principles technique for analyzing x-ray θ - 2θ diffraction peak lineshapes that is shown to work fairly well on strained epitaxial layers. In developing this method, we have taken into account not only instrumental effects, but also include mosaic spread (tilt) as well as lattice parameter strain relaxation effects due to defect formation. Examples from GaN/InGaN layers grown on sapphire substrates by molecular beam epitaxy are used to illustrate the quantitative structural information that can be extracted using this method of analyzing x-ray diffraction lineshapes. One basic assumption we make with our approach is that all single-crystal substrate (sapphire) peak lineshapes are entirely broadened by instrumental effects, which then allows us to quantify the instrumental broadening function and use it to evaluate the epitaxial layer crystal structure.

These first results provide valuable information on the bulk quality of GaN/InGaN structures that we are fabricating for photonic devices–in particular providing evidence for good crystal growth in a graded sapphire, GaN, InGaN (P-type) photocathode structure. Preliminary results of device performance will be presented elsewhere. Here we concentrate on the diffraction analysis methodology and present novel numerical techniques for diffraction analysis.

2 Model description

To extract physical information about the GaN crystal structure from its X-ray diffraction lineshape, we take the approach of performing a first-principles calculation of the full diffraction pattern taking into account only physically meaninful free parameters and performing Monte Carlo simulations of defects. Free parameters include those describing strain in the graded composition between GaN and InGaN, strain resulting from the substrate, a small mosaic spread, crystal unit cell sizes, and so forth. We then simulated the X-ray diffraction lineshape of the model by running a set of Monte Carlo calculations. A best fit to the data was found by altering the model parameters to minimize the χ^2 value.

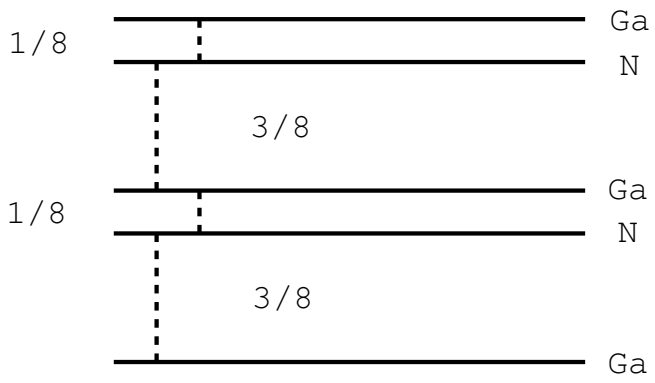

Figure 1: GaN Crystal Structure

2.1 Crystal Structure

We know that GaN has a wurtzite crystal structure. The wurtzite crystal has four layers perpendicular to the c axis per unit cell, each with equal numbers of atoms. We let the size of a single unit cell in the c direction be one of our parameters. Each layer is composed of only one element, and the elements alternate between layers. The separation between layers is uneven and alternates between 1/8 and 3/8 of the unit cell size. Even though all layers have the same number of atoms, the Ga and N layers have different numbers of protons and electrons per atom. This crystal structure is shown in Fig. 1. This combination of 1/8 and 3/8 spacing and the differing charges leads to incomplete destructive interference yielding a family of smaller secondary peaks. To reflect this effect in the model, we let each layer reflect a weighted amount of the incoming X-rays. We let the ratio of weights between layers be a free parameter, as this could change when, for example, we alloy with In or dope the material.

Also, the model crystal has a large but finite number of layers. Adding layers increases the computational complexity of the task, but having too few layers would cause unrealistic lineshapes. We estimated the number of layers from the crystal's thickness and layer spacing and determined that the crystals we have been working with are on the order of 1,000 unit cells thick. (These relatively thin layers have been chosen to balance optical absorption and diffusion losses in a thin transmission-mode photocathode.)

2.2 Regions

Because the real crystal has graded concentrations of In, we had to introduce a simple model to describe the graded crystal structure. At the same time, it was important to minimize the number of new free parameters. To this end, we divided the crystal into three regions, the boundaries of which are described by free parameters. Each region can have separate parameters, such as spacings, layer weights, and so on.

In order to also model nonlinear gradings, we introduced parameters describing the linearity of a transition between regions. Suppose, in the linear grading, we have x proportion of the first set of parameters and $1-x$ proportion of the second set of parameters. Then, if we use k to be the nonlinearity parameter and p_1 and p_2 to be the parameters,

$$p = p_1\left((1-x) - kx\left(1-x\right)\right) + p_2\left(x + kx\left(1-x\right)\right)$$

As k goes to 0, this becomes a linear equation. As k goes to ± 1, the equation becomes that of a 2^{nd} degree B-spline. Fig. 2 shows how much $p1$ contributes to p in the transition region as a function of the nonlinearity parameter.

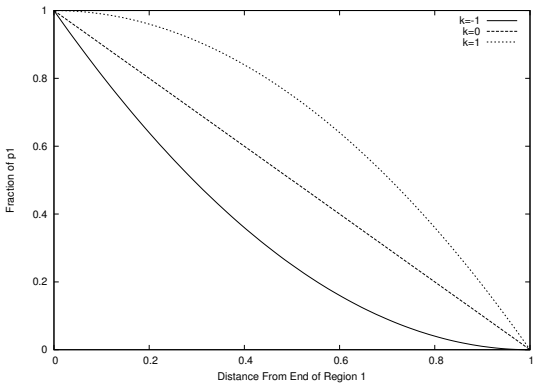

Figure 2: Nonlinearity Between Regions

2.3 Defects

As the crystal grows by epitaxy, small errors build up in the crystal structure. Eventually, these build up to a point that the stresses on the next layer are too great, and they are finally relieved by the formation of a *stacking defect - dislocation - Dan?*. We suppose that the defects are distributed in a Poisson manner. Therefore, there is an exponential distribution of lengths of unbroken coherent crystal. Each domain of unbroken crystal gives a $\mathrm{sinc}^2 x = \left(\frac{\sin x}{x}\right)^2$-shaped peak in the X-ray diffraction lineshape, and the thicker a segment is, the narrower its peak. The defects are random and relatively large, causing the different domains to add in random phase at the detector. Therefore, each peak in the lineshape is a sum of sinc^2 peaks with widths distributed in a Lorentzian manner. This feature was originally introduced to explain Lorentzian tails in the lineshape that could not be described by mosaic spread (Sec. 3.2) or instrumental broadening (Sec. 2.5). Since averaging over $\left(\frac{\sin x}{x}\right)^2$ terms gives a $\frac{1}{x^2}$ Lorentzian tail and since such stacking defects have a physical basis, we believe the introduction of an additional free parameter was warranted.

This explains why a Voigt function often approximates the lineshape of an X-ray diffraction peak very well. A Voigt function is the convolution of a Lorentzian and a Gaussian, which is similar to what our physical model predicts.

However, when we applied this technique to our model, we found that very small domains of the crystal were very common. These short domains produced very wide sinc^2 functions, leading to a peak with a base much wider than the one we saw in our data. We therefore concluded that strain builds up as more layers are added, and defects are unlikely to occur when strain has not had a chance to build up. So, we set a minimum number of layers between defects, reducing the number of very small domains in the crystal at the expense of adding another parameter.

The crystal has many domains with boundaries that contain the c direction, and each one might have a different set of defects. To model this, we simulate the crystal a number of times, each time with different random defects. Then, we average the lineshape of all these crystals.

2.4 X-Rays

X-ray diffraction data were collected using a Rigaku Geigerflex powder diffractometer in the Bragg-Brentano geometry, Cu Ka radiation, and a graphite exit monochromator. For the results from this diffractometer, instrumental broadening was significant. Before describing the method by which we account for such broadening, we describe our calculation of the intrinsic lineshape expected for an ideal diffractometer.

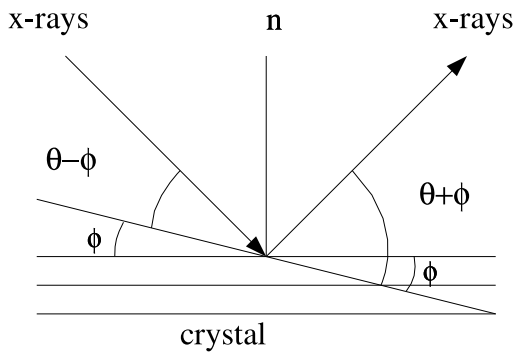

Figure 3: Mosaic Spread

For a detailed modeling of linewidths, we take into account the intrinsic line shape, spacing, and relative amplitude of Cu Kα lines. The line shape is given by a Lorentzian distribution in which the width parameter Γ is inversely prportional to the lifetime of the state. The energies are distributed around the central value E_0:

$$f(E) = \left(\frac{A\Delta E}{(E_0 - E)^2 + (\Delta E)^2} \right)^2$$

Here, A is the normalization factor. To account for this, first we calculate the lineshape due to X-rays of energy E_0. Then, we perform something akin to a convolution on this lineshape. See §3.1 for information on this convolution-like operation.

Further, in copper, there are two energy levels which produce X-rays of similar wavelengths. These are Kα_1 and Kα_2. As a result, we simulate the crystal under the two energies produced by the X-ray source. Each of the two energies has its own Lorentzian distribution of similar X-rays, with widths of 2.11 eV and 2.17 eV respectively, and both distributions are approximated as stated above.

2.5 Instrumental Broadening

The X-ray diffraction instrument also broadens the diffraction peaks we see. To model this, we examined the peaks that came from the substrate, which in our case was sapphire. The substrate was a very thick, high-quality crystal. For this reason, the peaks should be very sharp. Therefore, most of the broadening of the substrate peaks should be from the instrumental broadening convolved with the intrinsic linewidth. We took the widths of the substrate peaks to give us the instrumental effect, which was a Voigt function, the convolution of an assumed Gaussian instrumental function and a Lorentzian describing the intrinsic linewidth.

2.6 Mosaic Spread

In addition to being split into several domains throughout its thickness along the c direction, the crystal is also split into several domains through its length and breadth perpendicular to c. Each of these domains may be slightly tilted with respect to the others. This changes the distance that each incoming X-ray must travel to layers of that domain and hence the phase at the detector. Fig. 3 shows the sizes of angles in mosaic spread.

Instead of each ray traveling $2d\sin\theta$, a ray travels

$$d\left(\sin\left(\theta + \phi\right) + \sin\left(\theta - \phi\right)\right)$$

where we assume that ϕ follows a normal distribution with a standard deviation of σ_ϕ. We found that we were able to use a convolution-like method for calculating this spread, in much the same way as we calculate the spread in X-ray energies. See section 3.2 for an explanation of this convolution-like operation.

2.7 Non-Physical Parameters

In each simulation, we use a Monte Carlo method to produce the lineshape. In each iteration, we first generate a model crystal with random, Poisson-distributed stacking defects. Then, for each angle of the detector, we calculate the phase and amplitude of X-rays returning from each layer of the crystal.

Each Monte Carlo run results in one possible crystal structure with random defects. This results in fluctuations in the average behavior that grow lower as we simulate more crystals, but time restrictions limit us to using a finite number of iterations. Therefore, we have placed upper and lower bounds on the number of Monte Carlo iterations. As we get nearer the parameters which match the real data best, we use more and more iterations, taking more time while increasing accuracy. In this way, we do not waste computation in the early, rough stages of the minimization. The additional fluctuations in the early stages play a similar role to fluctuations introduced for the method of simulated annealing and help somewhat in avoiding local minima in the multidimensional parameter space (but still do not guarantee finding the global minimum).

3 Modified Convolution

Naively, one might expect the resulting diffraction pattern to result from a simple convolution of the intrinsic line shape, instrumental broadening, and the diffraction pattern obtained from a monochromatic X-ray source. More precisely, if we have monochromatic light, we have:

$$Diff(\theta) = \left(\sum_d e^{ik\,2d\sin\theta}\right)^2 \tag{1}$$

$$I(\theta) = (Inst \otimes Diff)(\theta) \tag{2}$$

However, with mosaic spread ϕ, the distance should no longer be $2d\sin\theta$ but rather $d\sin(\theta+\phi)+d\sin(\theta-\phi)$, giving us

$$Diff(\theta) = \left(\sum_d e^{ik\,(d\sin(\theta+\phi)+d\sin(\theta-\phi))}\right)^2$$

Here, k and ϕ vary continuously, and it can be seen that one cannot simply convolve $Diff$ with the distributions of k and ϕ. Fortunately, we can deal with this by noting the following.

When we measure an X-ray diffraction line, we record the intensity detected at equally-spaced values of the angle θ (figure 4). A change in the angles θ of the crystal and 2θ of the detector correspond exactly to a scaling of the crystal; according to Bragg's Law, X-rays must travel $2d\sin\theta$ farther to reach an inner layer a distance d below the surface of the crystal than to reach the outermost layer. Clearly, we could reproduce the effects of changing θ simply by changing d. By reversing this, we can simulate a change in wavelength, or another parameter, by a change in angle. Using this simple transformation, we can greatly speed up certain aspects of the computation by simply calculating the lineshape once and then resampling it at values that give the needed scalings of the crystal. Note that this is not rigorously a convolution, but it is conceptually similar.

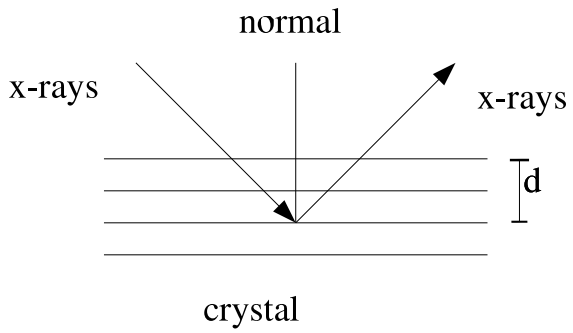

Figure 4: Crystal Distances

3.1 X-Ray Energy Distribution

The X-ray source does not produce a single wavelength, but rather X-rays with a distribution of energies around the transition energy. The distribution of intensities is a Lorentzian:

$$I(E) = \frac{AR}{(E - E_0)^2 + R^2}$$

A is a normalization factor and $2R$ is the full width half max of the distribution. The intrinsic line-width of the distribution is inversely proportional to the lifetime of the excited state. We can approximate how much radiation of energy E the detector will receive at angle 2θ by asking if there is an angle at which X-rays of E_0 would have experienced the same phase shift. Fig. 4 shows a diagram of the crystal with some of these quantities labeled.

In accordance with Bragg's law, the X-rays travel $2d\sin\theta$ farther to a layer d inside the crystal than to the first layer of the crystal. For an X-ray of energy E, the wavelength is $\frac{hc}{E}$. Therefore, this X-ray travels a number of wavelengths equal to

$$\frac{2d\sin\theta}{\frac{hc}{E}}$$

If we equate the phase of two X-rays of energy E and E_0, we get

$$\theta_0 = \arcsin\left(\frac{E\sin\theta}{E_0}\right) \tag{3}$$

Note that the equation holds for layers of any depth in the crystal, rather than just layers at some specific depth d. So, to estimate the radiation the detector receives at angle θ from X-rays of energy E, we use the amount of radiation the detector receives at angle θ_0 from X-rays of energy E_0. Here, if the X-ray energies are distributed according to the function $D(E)$, I_{final} is the final lineshape, and I_{E_0} is the lineshape of an infinitely sharp X-ray line at energy E_0, we see

$$I_{final}(\theta) = \int_{all\ energies\ E} dE D(E) I_{E_0}\left(\arcsin\left(\frac{E\sin\theta}{E_0}\right)\right)$$

So, to simulate the effects of the energy distribution, we divide the energy distribution into a set of discrete samples. Then, we calculate a lineshape for each shifted as above, and add them all together.

$$I_{final}(\theta) = \sum_{all\ energies\ E} \Delta E D(E) I_{E_0}\left(\arcsin\left(\frac{E\sin\theta}{E_0}\right)\right)$$

3.2 Mosaic Spread

In mosaic spread (Fig. 3), instead of the X-rays traveling a distance of $2d\sin\theta$, they travel

$$d\left(\sin\left(\theta+\phi\right)+\sin\left(\theta-\phi\right)\right)$$

Equivalently, they travel

$$d\left(2\sin\theta\cos\phi\right)$$

Equating this to an X-ray with $\phi = 0$ at angle θ_0, we get

$$\sin\theta_0 = \sin\theta\cos\phi \tag{4}$$

Note that, from this equation, we know that for all ϕ, we must have

$$\theta_0 \leq \theta$$

Also, we have

$$\frac{\partial\phi}{\partial\theta_0} = \frac{-\cos\theta_0}{\sin\theta\sqrt{1-\frac{\sin^2\theta_0}{\sin^2\theta}}}$$

We define some symbols.

- I' is the lineshape including mosaic spread
- I is the lineshape not including mosaic spread
- $D(\phi)$ is the distribution of ϕ in the mosaic spread; it is a normal distribution, with standard deviation σ_ϕ

We can approximate the lineshape with mosaic spread by an integral over all possible ϕ.

$$I'\left(\theta\right) = \int_{all\ \phi} d\phi D\left(\phi\right) I\left(\theta_0\left(\theta,\phi\right)\right)$$

Considering the nature of our data, it makes sense to change from integrating over ϕ to integrating over θ_0, which reverses the endpoints of the integral, changing the sign of the integral if we conventionally integrate over increasing θ_0.

$$I'\left(\theta\right) = -\int_{all\ \theta_0} d\theta_0 D\left(\phi\left(\theta_0,\theta\right)\right)\frac{\partial\phi\left(\theta_0,\theta\right)}{\partial\theta_0} I\left(\theta_0\right)$$

If we transform the integral to a sum, we then get

$$I'\left(\theta\right) \approx \sum_{all\ \theta_0} D\left(\phi\left(\theta_0,\theta\right)\right)\frac{\cos\theta_0}{\sin\theta\sqrt{1-\frac{\sin^2\theta_0}{\sin^2\theta}}} I\left(\theta_0\right)\Delta\theta_0$$

Note that this equation is only meaningful for θ_0 less than θ. However, we know from equation 4 that this is the case for all ϕ except $\phi = 0$. In the case when $\phi = 0$, $\theta_0 = \theta$. However, since we are dealing with discrete θ data, we are approximating the lineshape value at θ to be the average from $\theta - \Delta\theta$ to $\theta + \Delta\theta$. Therefore, we can take

$$I'(\theta=\theta_0) = \int_{\theta-\Delta\theta/2}^{\theta+\Delta\theta/2} d\theta_0 D\left(\phi\left(\theta_0,\theta\right)\right)\frac{\cos\theta_0}{\sin\theta\sqrt{1-\frac{\sin^2\theta_0}{\sin^2\theta}}} I\left(\theta_0\right)$$

Restricting the range to $\theta_0 < \theta$ and approximating everything with its first-order Taylor series expansion, we get

$$I'(\theta=\theta_0) \approx I(\theta_0)\left(\sqrt{\cot\theta_0}\left(\Delta\theta\right)^{\frac{1}{2}} + \frac{1}{6}\left(\frac{3}{4}\sqrt{\tan\theta_0} + \left(\frac{1}{4}-\frac{1}{\sigma_\phi^2}\right)\cot^{\frac{3}{2}}\theta_0\right)\left(\Delta\theta_0\right)^{\frac{3}{2}}\right)$$

4 Fitting Method

Although we have attempted to be conservative in introducing parameters, the resulting parameter space of the model is still very large. Using physical measurements and constraints, we can fix some of the parameters, but the number of remaining free parameters is still quite large. Therefore, it is difficult to examine the parameter space to find the best set of parameters corresponding to the true global minimum of χ^2. To this end, we use a modified simplex minimization method with some of the randomness of simulated annealing. The algorithm is essentially the "amoeba" algorithm from Numerical Recipes, with a few alterations [PTVF92].

4.1 Goodness of Fit

In order to fit the model to the target lineshape, we had to choose a metric to measure how well the model matches the target. We use the standard χ^2 metric. First, we use the model to generate our own X-ray diffraction lineshape. Then, we take χ^2 to be a weighted sum of the squares of the differences of pairs of points between our simulated lineshape and the target lineshape. The weight is the reciprocal of the expected variance of the X-ray intensity values.

$$\chi^2 = \sum_i \frac{(t_i - s_i)^2}{\sigma_{t_i}^2} \tag{5}$$

Here, t_i is the target value and s_i is the simulated value.

4.2 Speeding the Fit

4.2.1 Reducing the Number of Iterations

To make the fit faster, we realized that we did not need to use the same number of Monte Carlo iterations at all stages of the fit. As each iteration has random elements, noise is introduced into the vertices of the simplex. That is, each point in parameter space has an associated value, but the calculation of this value will introduce an error. However, averaging more iterations will decrease the error; recall, $\sigma_{\overline{x}} = \sigma_x/\sqrt{N}$. At first, as the simplex is very large and very far from the minimum, the differences between corners of the simplex will be very large, making it easy to tell which way the simplex should move. Also, some randomness is desired in the simplex so that it might escape local minima. In this case, it is possible to use fewer iterations, thus requiring less time. However, as the simplex nears the minimum and begins to shrink, the differences between the corners of the simplex become less pronounced. As this happens, we perform more and more iterations. The important thing is to keep the random noise level small enough that the highest and lowest of the vertices of the simplex can easily be distinguished. As a result, we use enough iterations to keep the standard deviation smaller than some fraction of the difference between the highest and lowest vertices.

However, as the simplex approaches the termination conditions (i.e. as it gets very small), the number of iterations that must be taken becomes prohibitive. To prevent this, we use an upper bound on the number of iterations. In some trials, we found this to be useful, as a properly chosen upper bound allows convergence without overmuch computation. If one discovers that convergence is not occurring with the current upper bound, it is easily possible to increase it and run the minimization again.

4.2.2 Speeding Each Iteration

To speed each iteration, we took several measures. First, we tried a Fourier Transform method of summing the contributions of various layers of the crystal. In principle, this should be an $O(n \log n)$ method, rather

than an $O(n^2)$ method. However, there were some inevitable discretization errors. Lowering these errors to a truly acceptable level removed any time advantages of this method.

We also examined diminishing the amount of time by decimating our lineshape data. By removing three samples in four, we achieved significant speed increases. Of course, this meant a loss of information carried in the lineshape.

In the end, we decided to combine these speed-enhancing tools in a first stage. First, we fit the model with decimated data and a Fourier Transform simulation method. Once this converges, we should be at least somewhere near the correct parameters. Finally, we attempt to fit the model with a full set of data and the slower, more accurate simulation method. In this way, we were able to sidestep a large portion of the minimization time cost.

5 X-Ray Diffraction Error Analysis

5.1 Count Data Error

The X-ray diffractometer counts incoming X-rays reflected off of the sample. In count data, the variance of the error is equal to the count. Therefore,

$$\sigma_{\hat{t}_i}^2 = \hat{t}_i$$

where $\hat{t}$ is the "real" data, untainted by any systematic detector errors.

5.2 Constant and Fractional Error

X-ray diffraction data typically include primary peaks with large signal-to-noise ratio (SNR) as well as smaller secondary peaks with much lower SNR. For the incredibly high count-rates in the dominant peaks, statistical errors are negligible compared with systematic errors. However, a standard weighted fit rewards these points to the extent that information contained in the secondary peaks is rendered almost irrelevant. While a quantitative treatment of systematic errors is difficult by definition, we consider a method to take into account the effect of systematics on our fit.

We suppose that the detector introduces some error on every sample. The detector gives us t from the ideal $\hat{t}$.

$$t = f\hat{t} + c$$

If we look at where the error in t comes from, we see

$$\sigma_t^2 = \sigma_f^2 \left(\frac{\partial t}{\partial f}\right)^2 + \sigma_{\hat{t}}^2 \left(\frac{\partial t}{\partial \hat{t}}\right)^2 + \sigma_c^2 \left(\frac{\partial t}{\partial \hat{c}}\right)^2$$

If we assume that the detector gives the same mean data as the "real" data, then we get f centered at 1 and c centered at 0.

$$\sigma_t^2 = \sigma_f^2 \hat{t}^2 + \sigma_{\hat{t}}^2 + \sigma_c^2$$

5.3 Estimation of Error

To estimate the constant systematic error, we examined the error in the baseline. To calculate the percentage error, we used a fit of one peak of the lineshape. The fit was visually very good, but its calculated error was still surprisingly high. We assume that this lineshape should give a perfect fit if there were no systematic error, so its χ^2 should give the expected value of a χ^2 distribution. The expected

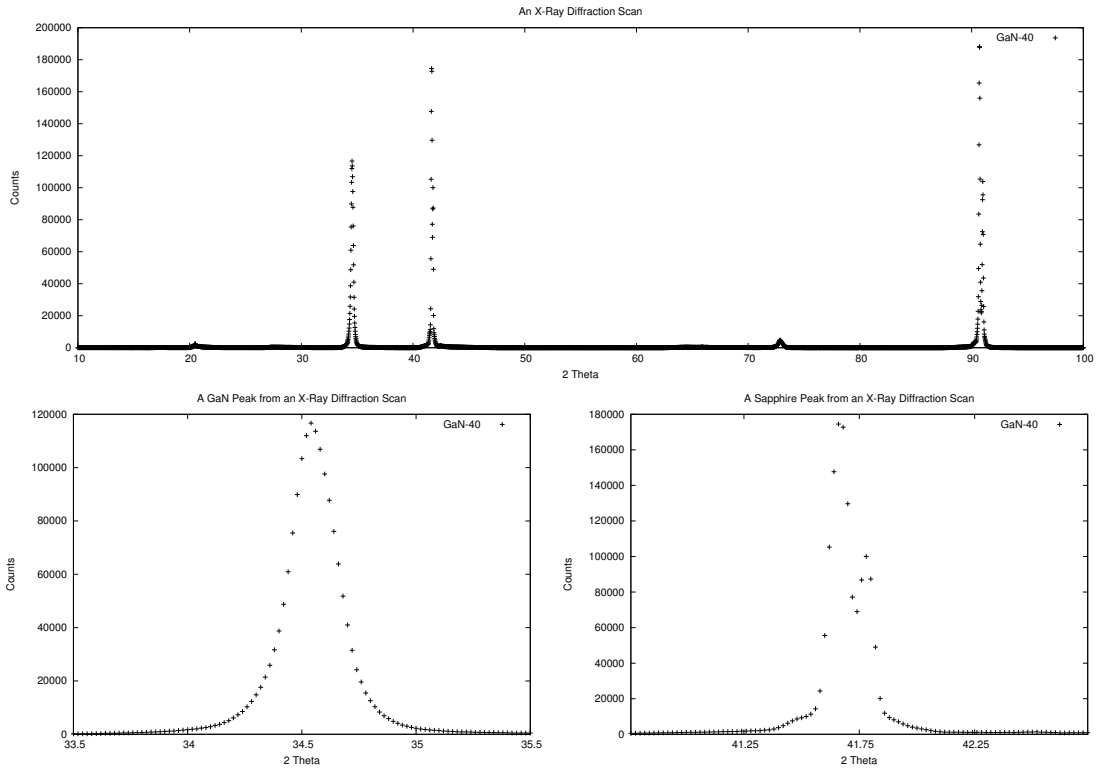

Figure 5: Raw X-Ray Diffraction Scan

value of a χ^2 distribution is the number of its degrees of freedom, $ndf = data\ points - free\ parameters$. We then estimate the percent systematic error required to have this χ^2 value:

$$ndf = \chi^2 = \sum_i \frac{\left(t_i - \hat{t}_i\right)^2}{t_i + c^2 + \left(t_i f\right)^2}$$

We then solved for the fractional error, f, by simple iteration using

$$f^2 = \frac{1}{ndf} \sum_i \frac{\left(t_i - \hat{t}_i\right)^2}{\frac{t_i}{f^2} + \frac{c^2}{f^2} + t_i^2}$$

6 Results

7 Program Usage

7.1 Setup

7.1.1 Obtaining the Program

The program `xraymodel` is available via CVS from jelley.wustl.edu under the name `xraymodel`. To obtain a copy, you must have an account on jelley.wustl.edu. Then, type the command:

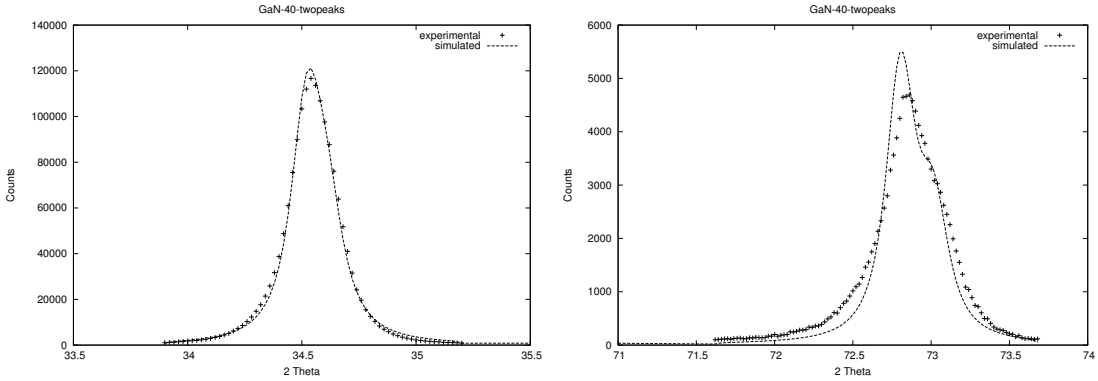

Figure 6: Results: GaN-40

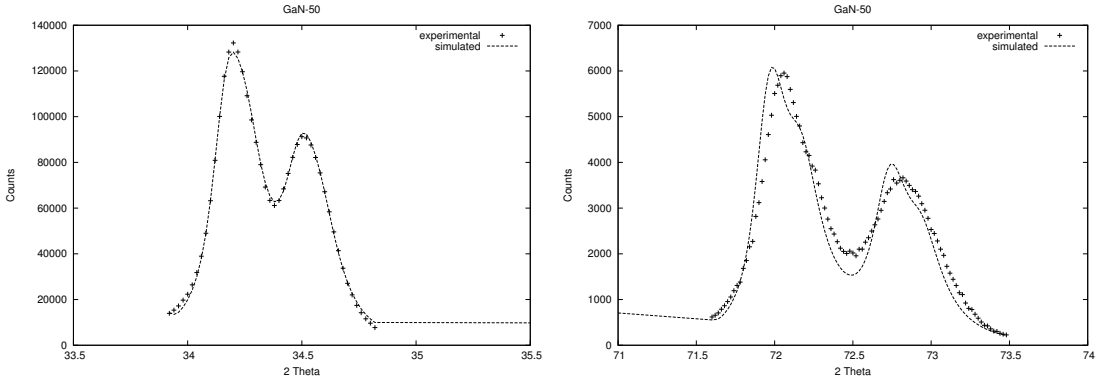

Figure 7: Results: gan50

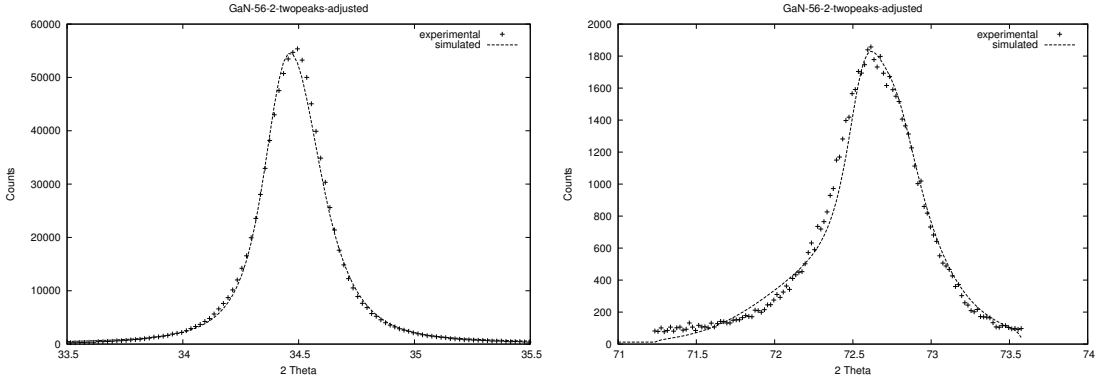

Figure 8: Results: GaN-56-2

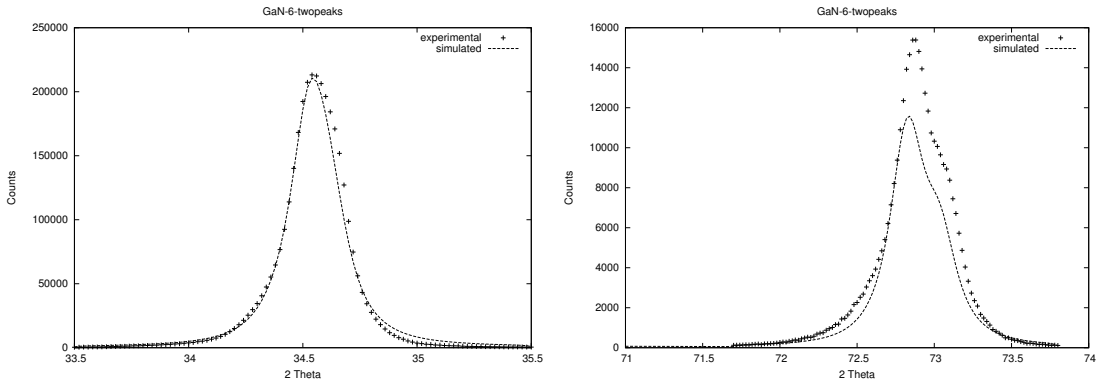

Figure 9: Results: GaN-6

```
cvs -d :ext:your_username@jelley.wustl.edu:/home/cvsroot checkout xraymodel
```

This will create a directory called `xraymodel` which contains all the source files for the program.

7.1.2 Building the Program

To build `xraymodel`, first go to the directory you checked out. Then, type

```
make
```

to compile.

The makefile attempts to automatically detect whether MPI (used in computer clusters) is installed on the system and configure the program correctly. However, if the makefile detects it incorrectly (e.g. attempting to use it when it is unavailable), you can override it by setting the variable `USING_MPI` to `true` or `false` as follows:

```
USING_MPI=true make
```

7.2 Executing the Program

If you execute the program `xraymodel` with no command-line arguments, you will get some helpful output containing the usage of various options. However, I will here summarize the most common command lines I use.

Note on PATH: the diretory containing the executable `xraymodel` must be in your PATH or else you will need to run it with the full path (e.g. `./xraymodel`). See UNIX documentation for more information on the `PATH` variable.

To use `xraymodel` to generate a crude guess for a set of beginning parameters, use:

```
xraymodel -g data_file new_parameter_file
```

To use it to fit a set of parameters to a data file, use

```
xraymodel -f data_file initial_parameters plot_out_file parameter_out_file
```

In this case, the plot_out_file will get three columns of data:

1. The 2θ data

2. The real intensity data

3. The simulated intensity data

The parameter_out_file will receive a summary of the chi-squared error and the best-matching parameters. Please note that sometimes running the program once is not enough to find the best set of parameters due to Monte Carlo error and local minima. To run the program again, starting from the previous best parameters, run:

```
xraymodel -f data_file previous_parameter_out_file plot_out_file new_parameter_out_file
```

To simply simulate the x-ray diffraction peaks resulting from a set of parameters, rather than attempt to find the best parameters, use the command:

```
xraymodel -s data_file parameter_file plot_out_file
```

(This requires a data file to determine on what 2θ range the simulation should occur.)

It is no secret that `xraymodel` takes a long time to run. As a result, it is often run on a Beowulf cluster with many different processes talking to each other via MPI. In this case, `xraymodel` needs to be run by `mpirun` and told that it should run using MPI with the `-M` option like this:

mpirun -np num_mpi_nodes `xraymodel -f`*M* `data_file initial_parameters plot_out_file parameter_out_file`

8 Parameter Interpretation

There are 28 parameters, numbered from 0 to 27, describing the crystal. They are as follows.

0. Number of layers – This is the number of unit cells in the thickness of the crystal. For the crystals I've been working on, it's about 1000. This parameter should not be fit (among other things, because it's discrete and thus not differentiable), but should instead be estimated from the thickness of the crystal and the size of a single unit cell (roughly 5.185 angstroms). There are actually 4 layers per unit cell, so you would multiply this number by 4 to get the number of layers. In each unit cell, the layers alternate between Ga and N, and they are spaced at 0, 3/8, 1/2, and 7/8 of the unit cell. This number shouldn't be terribly large, because it increases the computational complexity. However, if it is terribly small, then we might not be able to see proper effects from the defects.

1. Spacing 1 – This is the size in meters of the unit cell in region 1 of the crystal. It affects the location of the peaks of the first region. Use Bragg's equation to calculate where the peaks go: $2dsin(\theta) = n\lambda$. So calculate the wavelength of the x-rays, and you can calculate the location of the peaks (at n=1 and n=2, respectively).

2. Defect density 1 – This number gives the fraction of layers in this region which have defects. This has always been fairly small for me, around .02. This number describes the width of the peaks. Note that I haven't really localized the defect density to any particular region, so having 3 different defect density parameters is kind of bogus.

3. Relative layer weight 1,2 (w/r to weight of first layer in first level) – See Sec. 8.1 about layer weights.

4. Non-linearity between 1 and 2 – The nonlinearity parameters describe whether the grading between one region and another are linear or not. This is based on a B-spline equation. See the paper for details. Usually, the grading is linear, so the value should be 0 or near 0.

5. Relative layers of distance between levels 1 and 2 (w/r to 1) – See Sec 8.2 about relative size.
6. Relative layers of level 2 (with respect to 1) – See Sec 8.2 about relative size.
7. Spacing 2 – Like spacing 1, but for region 2.
8. Defect density 2 – Like defect density, but for region 2. Note that I really haven't managed to make the defects appear in their proper regions.
9. Relative layer weight 2,1 – See Sec. 8.1 about layer weights.
10. Relative layer weight 2,2 – See Sec. 8.1 about layer weights.
11. Non-linearity between 2 and 3 – Like the other nonlinearity parameter.
12. Relative layers of distance between levels 2 and 3 (w/r to 1) – See Sec 8.2 about relative size.
13. Relative layers of level 3 (with respect to 1) – See Sec 8.2 about relative size.
14. Spacing 3 – Like spacing 1, but for region 3.
15. Defect density 3 – Like defect density 1, but for region 3. Note that I really haven't managed to make the defects appear in their proper regions.
16. Relative layer weight 3,1 – See Sec. 8.1 about layer weights.
17. Relative layer weight 3,2 – See Sec. 8.1 about layer weights.
18. Cu K-α-2 fraction (.5 by default) – This describes the amount of copper's second wavelength with respect to its first wavelength, K-α-1. We added this parameter because we weren't sure that it was exactly 0.5. Feel free to change it if you get more data on this. Otherwise, I'd tend to assume that .5 is right.
19. Minimum size in layers of a domain in the crystal – We found that the crystal defects tended to appear to close, making a lot of very small crystal domains. This added up to a lot of very wide gaussians being added to the peak, making its shape unlike that of the real peak. We took that to mean that very small domains don't appear in the crystal. So this parameter gives the minimum spacing between defects.
20. Standard deviation in radians of the angle dislocation (phi) in the crystal – This number describes the mosaic spread of the crystal. See the paper for an explanation.
21. Overall amplitude weight – When I sum up the signal the simulated detector receives, I use some arbitrary units, making the whole lineshape scaling wrong. This parameter scales the entire lineshape. You definitely should let this parameter be fit.
22. Baseline level 1 (for 2-theta below TWO_THETA_CUTOFF) – This says what the baseline is like around the peak near 2theta = 34 degrees.
23. Baseline level 2 (for 2-theta above TWO_THETA_CUTOFF) – This says what the baseline is like around the peak near 2theta = 72 degrees.
24. ce – the constant error in each intensity value – This says what the error is in the baseline.
25. fe – the fractional error in each intensity value – This says what fractional error the detector gives.

26. Minimum number of iterations of the Monte Carlo peak-generating function – This parameter puts a bottom limit on the number of iterations to generate a lineshape. I tend to leave it around 100. This keeps the calculations from getting too imprecise.

27. Maximum number of iterations of the Monte Carlo peak-generating function – This parameter puts an upper cap on the number of iterations to generate a lineshape. This means that the random effects won't ever get smaller than some value. It also keeps the computer from working too hard. Otherwise, simulation could really take forever.

8.1 Layer Weights

Not all layers have the same weight. Recall, there are Ga and N layers, and they have different atomic numbers, so they respond to the x-rays differently. Each layer, N and Ga, gets a weight, but this can also change between regions, as some Ga is replaced with In, for example. So I need 6 layer weights. However, the relative weight is what is important, so it turns out that having 6 parameters means that there are some dependencies between parameters (the parameters overdetermine the model). Therefore, I fixed the first layer weight at 1. So, we have weights like this:

```
REGION 1:    layer 1:    1
             layer 2:    Relative layer weight 1,2
REGION 2:    layer 1:    Relative layer weight 2,1
             layer 2:    Relative layer weight 2,2
REGION 3:    layer 1:    Relative layer weight 3,1
             layer 2:    Relative layer weight 3,2
```

So what do these layer weights affect? They determine the relative amplitudes of the peaks at n=1 and n=2. It turns out that the wurtzite structure (layers at 0,3/8,1/2,7/8) means that there is some destructive interference reducing the height of the peak at n=2. In fact, I think that if the weights are the same, then the peak at n=2 (i.e. at 2θ=72 degrees) vanishes completely.

8.2 Relative Size or Layers of Distance

I wanted the 3 crystal regions to be able to change size, but I wanted the parameters to be change in a pretty much unbounded way. I could have had percentages (region 1 takes 50%, region 2 takes 20%, etc.), but then it would be too easy to have the percentages add to more than 1. What I did instead was to have weights on the size of each region. Then, the percentage size is weight[i]/sum(weights). I also added similar weights to the graded areas between regions. In this way, I could make the grading faster or slower. However, I had the same overdetermination problem as in the layer weights, so I fixed one of the weights to be 1. For example:

Region	Weight	Fraction	Number of layers
1	1	1/TOTAL	1/TOTAL*TOT_LAYERS
grade btw 1-2	param 5	param 5/TOTAL	param 5/TOTAL*TOT_LAYERS
region 2	param 6	param 6/TOTAL	param 6/TOTAL*TOT_LAYERS
grade btw 2-3	param 12	param 12/TOTAL	param 12/TOTAL*TOT_LAYERS
region 3	param 13	param 13/TOTAL	param 13/TOTAL*TOT_LAYERS

References

[PTVF92] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: the Art of Scientific Computing*. Cambridge University Press, New York, second edition, 1992.