Ultramicroscopy 158 (2015) 89–97

Contents lists available at ScienceDirect

Ultramicroscopy

journal homepage: www.elsevier.com/locate/ultramic

FDES, a GPU-based multislice algorithm with increased efficiency of the computation of the projected potential

W. Van den Broek *, X. Jiang, C.T. Koch

Institute for Experimental Physics, Ulm University, Albert-Einstein-Allee 11, 89081 Ulm, Germany

ARTICLE INFO

Article history:
Received 5 May 2015
Received in revised form
10 July 2015
Accepted 21 July 2015
Available online 23 July 2015

Keywords:
Forward dynamical electron scattering (FDES)
Multislice simulation
HRTEM
Diffraction
CBED

ABSTRACT

While the computational complexity of calculation of the projected potential in the multislice algorithm through reciprocal space scales quadratically with the number of atoms A per slice, a pure real-space calculation scales linearly with A. A hybrid strategy is introduced that has a theoretical complexity of $O(A \log A)$, but that, when measured, outperforms both the reciprocal-space and the real-space approach by approximately an order in A and a large factor, respectively.

This strategy is implemented in a new program, dubbed forward dynamical electron scattering (FDES), which simulates high resolution transmission electron microscopy images, diffraction patterns and convergent beam electron diffraction patterns. FDES attains a further increase in speed by running on a graphics processing unit and is made available to the community as open software.

1. Introduction

The multislice algorithm (MSA) [1–5] is at the heart of many of the modern simulation codes in the field of transmission electron microscopy (TEM) [5–9]. It describes the propagation of fast electrons through a solid, neglecting backscattering, and has been used in the simulation of a myriad of image modes in TEM, most notably high angle annular dark field scanning TEM (HAADF-STEM) and conventional TEM (CTEM).

The MSA can be seen as having two main parts: (i) the projected-potential step, where the specimen is built from the electrostatic projected potentials of the constituting atoms and (ii) the propagation step, where the electron wave is propagated through the specimen.

In HAADF-STEM simulations, the propagation must be carried out for each beam position, while the projected potential can be reused. The potential's contribution to the total computation time is therefore non-dominating. It is for this reason that when Dwyer presented the first graphics-processing-unit (GPU) accelerated HAADF-STEM simulation code [10], the potential was not included in the parallelization.

In recent years an increase in two-dimensional and three-dimensional reconstruction methods is seen that use as input images other than HAADF-STEM images. For instance, ptychography [11] uses a set of convergent beam electron diffraction (CBED) patterns, and in [12–14] three-dimensional reconstructions from high resolution TEM (HRTEM), ptychography and scanning confocal electron microscopy [15] images have been proposed.

Accurate theoretical assessment of these novel methods requires extensive simulation. Since often the projected potential can only be reused a limited number of times (ptychography) or requires a complete recalculation for each simulated image (HRTEM), a fast potential computation is imperative. This need is addressed in this paper by introducing a novel and more efficient approach to the calculation of the projected potential.

As pointed out in [16], since the MSA is based in real space, it is excellently suited for structures with deviations from perfect crystallinity, caused by for example strain fields, defects, interfaces or the thermal vibration of the atoms about their equilibrium positions. In order to achieve diffraction data up to high angles while at the same time resolving fine details in the diffraction pattern, a small sampling distance and a large spatial extent must be combined.

The improvement of GPUs in the last years allows one to combine these two demands, which is reflected in the fact that various GPU-programs have recently been presented for e.g. projected-potential calculations through reciprocal space [17], stacked-Bloch-wave calculations [18] and various imaging modes based on inelastic scattering [19], and a real-space calculation of the three-dimensional potential for a MSA in [20]. In this paper, forward dynamical electron scattering (FDES) [21] is presented in

* Corresponding author.
E-mail address: wouter.vandenbroek@uni-ulm.de (W. Van den Broek).

http://dx.doi.org/10.1016/j.ultramic.2015.07.005
0304-3991/

full, a multislice program which implements our novel and more efficient projected-potential calculation on a GPU. This increase in speed also allows the available memory to be used very efficiently, so that systems comprising millions of atoms can easily be handled.

An overview of the multislice algorithm and its practical implementation is given in Section 2. In Section 3 it is shown that a reciprocal-space based projected-potential calculation requires an amount of operations that scales quadratically with the number of atoms per slice of the MSA and that a real-space approach scales linearly. A hybrid alternative is proposed that outperforms both. This assertion bears out in Section 4 when tested in practice with GPU-accelerated code. In Section 5 FDES is presented, a new multislice program which implements this new strategy and which attains a further increase in speed by running on a GPU.

2. The multislice algorithm

In the MSA the specimen is divided in thin slices and each slice is replaced with a transmission function t_j, where the index j runs over all slices. The electron wave ψ_j impinging on slice j is multiplied with t_j and propagation to the next slice is accomplished by a convolution with the Fresnel propagator p [5],

$$\psi^{(j+1)} = p \otimes \left(\psi^{(j)} t^{(j)}\right). \tag{1}$$

The transmission function $t^{(j)}$, also known as the phase grating, is defined as

$$t^{(j)}(x, y) = \exp\left(\mathrm{i}\sigma V^{(j)}(x, y)\right) \quad \text{with } V^{(j)} = \int_{-\infty}^{+\infty} v^{(j)}(x, y, z)\, \mathrm{d}z, \tag{2}$$

where σ is the interaction constant and $v^{(j)}$ is the summation of the electrostatic potentials of the atoms located within slice j. $V^{(j)}$ is called the projected potential and

$$V^{(j)}(x, y) = \sum_{k=1}^{A^{(j)}} V_{Z(k)}(x - x_k, y - y_k). \tag{3}$$

The index k runs over all $A^{(j)}$ atoms within layer j, $Z(k)$ is the atomic number of atom k, V_Z is the projected atomic potential of the atom with atomic number Z and x_k and y_k are the coordinates of atom k.

2.1. Practical implementation of the projected potential computation

2.1.1. Real-space approach

The common approach of computing the projected potential in real space comes with a few difficulties: V_Z exhibits a (mild) divergence in the origin and the parametrization of V_Z in [5], one of the best available, involves the computationally intensive modified Bessel function K_0. Furthermore, to reduce computation time, V_Z is only calculated within a cut-off radius r_c around the atoms' center. Sensible as this strategy might be, it does introduce a trade-off between model accuracy and computation time, as too small a radius causes scattering off the artificial discontinuity at the cut-off.

In the implementation used in this paper the atom positions are distributed over the GPU's processors and each processor calculates the projected potential in all pixels within the radius r_c. The divergence of K_0 at the origin is avoided by setting radii below 0.001 nm to 0.001 nm, r_c was set to 0.5 nm and the K_0-functions were computed with the closed-form parametrization from [22].

2.1.2. Reciprocal-space approach

The problems of the real-space approach are avoided when instead one works with the scattering factors f_Z, which are the Fourier transforms of V_Z. The parametrization that is used throughout this work, from [5], only involves Lorenzians and Gaussians and thus is well-behaved everywhere.

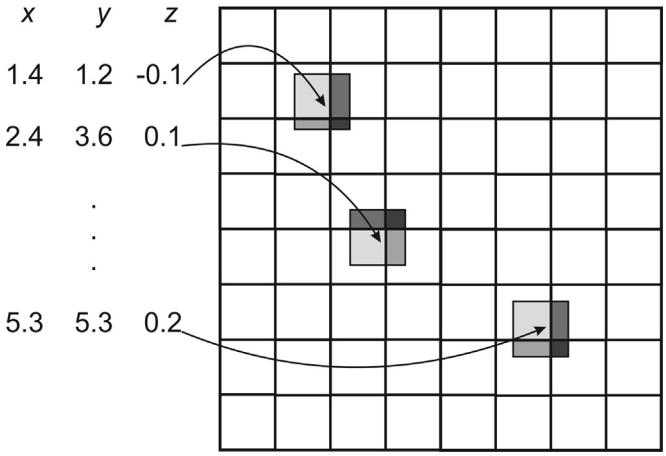

Fig. 1. Illustration of the sq-step in the hybrid phase-grating approach. The four pixels surrounding the locations of the atoms within a slice are assigned values corresponding to the intersection of these pixels with a pixel-sized square centered on the atom locations.

In Fourier space, the real-space translations of the atomic potentials V_Z in Eq. (3) are accomplished by multiplying f_Z with a phase ramp, so that

$$V^{(j)}(x, y) = \mathcal{F}^{-1}\left(\sum_{k=1}^{A^{(j)}} f_{Z(k)}(q_x, q_y) \exp\left(-2\pi\mathrm{i}(q_x x_k + q_y y_k)\right)\right), \tag{4}$$

where q_x and q_y are the spatial frequencies. This approach is followed in [17]. Furthermore, the computation speed is increased by factoring the exponential terms in two independent q_x- and q_y-components.

2.1.3. Hybrid approach

In this paper a hybrid between the real-space and the reciprocal-space approach is proposed. There are two steps to this approach: assigning squares and performing a convolution and a deconvolution, abbreviated as sq and de/c, respectively.

sq As illustrated in Fig. 1, the positions of all atoms of a given element are distributed over the GPU's processors and on each processor the four pixels surrounding the locations of the atoms within a slice are assigned values corresponding to the intersection of these pixels with a pixel-sized square centered on the atom locations.

de/c 1. The image resulting from the sq-step is Fourier transformed,
2. multiplied with f_Z,
3. divided by a sinc-function to offset the finite extent of the square in the sq-step and
4. the inverse Fourier transform is taken.

These steps are repeated for each different element in the specimen and the results are summed up.

2.2. Practical implementation of the wave propagation

In most practical implementations,[1] the real-space convolution in Eq. (1) is executed as a multiplication in Fourier space and thus requires a succession of a forward and an inverse Fourier

[1] For some notable exceptions, see [23,24].

transform. Another succession of a forward and an inverse Fourier transform is required to bandwidth-limit the transmission function and the Fresnel propagator to 2/3 of the Nyquist frequency in order to avoid aliasing artifacts [5]. The bandwidth-limited Fresnel propagator can be pre-calculated and reused for each slice of the MSA and therefore does not significantly contribute to the computation time of the wave propagation.

Furthermore, since in this work computation of the transmission function is taken as part of the wave-propagation, the wave propagation follows these steps:

1. construct the transmission function t according to Eq. (2),
2. bandwidth-limit t, this requires a succession of a forward and an inverse Fourier transform,
3. multiply the incoming electron wave with t and
4. convolve the result with the bandwidth-limited Fresnel propagator, again requiring a succession of a forward and an inverse Fourier transform.

3. Computational complexity of the steps of the multislice algorithm

The theoretical computational complexities of the projected-potential calculations in Section 2.1 and of the wave-propagation in Section 2.2 are analyzed.

To keep the derivations tidy, from now on only one slice of the MSA is considered and therefore the superscripts j are omitted. Furthermore, only specimens with a single element are used. The images are assumed square with $\sqrt{N} \times \sqrt{N} = N$ pixels. The amount of atoms within the slice is denoted A and $N = \alpha A$, where α depends on the spatial extent of the slice and thus on the width d of the pixels. Furthermore, the atom density ρ, i.e. the number of atoms per unit of area, equals $1/(d^2\alpha)$.

The fast Fourier transform (FFT) is a family of efficient algorithms whose computational complexity is $O(N \log N)$, with N the number data points or pixels. In the remainder of this work, all Fourier transforms are assumed to be carried out with an FFT.

3.1. Real-space approach

Since the projected potential is only calculated within the radius r_c around the atoms' center, the computation time for the real-space approach scales with the product of A and the number of pixels within r_c: $\pi r_c^2/d^2 = \pi r_c^2 \alpha\rho$. Thus resulting in a computational complexity of $O(\alpha A)$.

3.2. Reciprocal-space approach

Since only a single element is assumed, the factor $f_{Z(k)}$ in Eq. (4) can be brought outside the summation and the computation time of the projected potential is dominated by that of the individual phase ramps. Since the amount of operations in Eq. (4) scale with N and A, the total reciprocal-space computational complexity is $O(AN) = O(\alpha A^2)$.

3.3. Hybrid approach

The amount of operations in the sq-step scales with $4A$, so that the computational complexity T_{sq} of the sq-step is $O(4A)$. The most demanding steps in the de/c-stage are the forward and the inverse Fourier transforms, therefore, the computational complexity of the de/c-step is $O(2N \log N) = O(2\alpha A \log(\alpha A))$.

3.4. Wave propagation

The most demanding steps in the wave propagation are the two forward and inverse Fourier transforms, therefore, the computational complexity is $O(4N \log N) = O(4\alpha A \log(\alpha A))$.

4. Experimental determination of computation time

Promising as the results in Section 3 may be, they do not tell anything about the relative sizes of possible prefactors, or the influence of parallelization on the respective computation times when the algorithm is carried out on a GPU. Therefore, test-code was written in CUDA and executed on a Tesla K20c GPU (NVIDIA) to measure the run time of various functions. The runtimes are measured in clocks, with 1 clock = 1 ms.

4.1. Fourier transform

CUDA's FFT for complex arrays, cufftExecC2C and denoted cufft from here on, is tested on complex arrays with random entries and sizes varying from 32×32 to 8192×8192. The results are presented in Table 1. A polynomial fit reveals that the run time T_{cufft} of a single Fourier transform scales linearly with the number of data points,

$$T_{cufft} = 4.2 \times 10^{-7} \cdot N^{1.0} \text{ clocks}, \quad (5)$$

for arrays larger than or equal to 256×256. This means that although the number of operations in the de/c-step is $O(2N \log N)$, in practice the computation time outperforms that due to cufft's efficient parallelization.

4.2. Real-space approach

The computation time T_{rs} of the real-space approach is measured with the aid of a test specimen that consists of a random arrangement of carbon atoms with a volume density of 100 atoms per nm^3. At a slice thickness of 0.5 nm, the three typical pixel sizes d of 0.01 nm, 0.02 nm and 0.04 nm correspond to three α-values of 200, 50 and 12.5, respectively; furthermore, ρ equals 50 atoms/nm^2. These settings lead to a large dynamic range of the number of atoms A, spanning four orders of magnitude.

The measured computation times are listed in Table 2.

From Fig. 2 it can be seen that T_{rs} has two different regimes. A polynomial fit reveals

$$T_{rs} = 0.36 \cdot \alpha^{0.96} N^{0.037} \text{ clocks} \quad \text{for } A < 8.1 \times 10^3 \quad (6)$$

$$\simeq 0.56 \cdot \alpha^{0.96} \text{ clocks} \quad \text{and} \quad (7)$$

Table 1
Computation times T_{cufft} of cufft and T_{pr} of the phase ramp in Eq. (4). The functions were run E times in a row to amass approximately 100 clocks for the fastest processes.

$\sqrt{N} \times \sqrt{N}$	E	$E \cdot T_{cufft}$ (clocks)	$E \cdot T_{pr}$ (clocks)
32×32	655 360	11 421	4722
64×64	163 840	2699	1190
128×128	40 960	874	353
256×256	10 240	296	148
512×512	2560	249	118
1024×1024	640	281	102
2048×2048	160	250	95
4096×4096	40	249	93
8192×8192	10	296	92

Table 2
Computation time T_{rs} of the real-space approach. The function was run E successive times. Note how the number of atoms A ranges from 3.3×10^2 to 1.3×10^6.

$\sqrt{N} \times \sqrt{N}$	E	A	$E{\cdot}T_{rs}$ (clocks)	T_{rs} (ms)
$d=0.04$ nm, $\alpha = 12.5$				
256×256	256	5.2×10^3	1650	6.4
512×512	64	2.1×10^4	815	12.7
1024×1024	16	8.4×10^4	953	59.6
2048×2048	4	3.4×10^5	1123	280.8
4096×4096	1	1.3×10^6	1138	1138.0
$d=0.02$ nm, $\alpha = 50$				
256×256	256	1.3×10^3	6022	23.5
512×512	64	5.2×10^3	1560	24.4
1024×1024	16	2.1×10^4	1077	67.3
2048×2048	4	8.4×10^4	1139	284.8
4096×4096	1	3.4×10^5	1186	1186.0
$d=0.01$ nm, $\alpha = 200$				
256×256	256	3.3×10^2	23 244	90.8
512×512	64	1.3×10^3	5913	92.4
1024×1024	16	5.2×10^3	1576	98.5
2048×2048	4	2.1×10^4	1217	304.3
4096×4096	1	8.4×10^4	1201	1201.0

$$T_{rs} = 4.5 \times 10^{-5} \cdot N^{1.03} \alpha^{-0.032} \text{ clocks} \quad \text{for } A \geq 8.1 \times 10^3 \tag{8}$$

$$\simeq 4.0 \times 10^{-5} \cdot N^{1.03} \text{ clocks}. \tag{9}$$

Eqs. (7) and (9) are derived from Eqs. (6) and (8), respectively, by replacing the $N^{0.037}$-factor and the $\alpha^{-0.032}$ – factor by the averages taken over all values contributing to the fit.

4.2.1. Precomputed real-space approach

At first sight it might seem more efficient to precompute the projected potential. To check this, test code was written that distributed the atom positions over the GPU's processors and let each processor infer the projected potential in all pixels within the cut-off radius r_c through linear interpolation of projected-potential values in a precomputed array with 128 elements.

For a number of atoms $A \geq 8.1 \times 10^3$, the computation time T_{rs} turns out to be comparable to, and mostly even a bit larger than, that of the non-precomputed case: for $\alpha = 12.5$ and N adopting the values 256×256, 512×512, 1024×1024, 2048×2048 and 4096×4096, the computation time T_{rs} equals 3.6, 10.7, 75.1, 342.8 and 1404.0 ms, respectively.

4.3. Reciprocal-space approach

The phase ramp in Eq. (4) is tested on complex arrays with sizes varying from 32×32 to 8192×8192. The results are presented in Table 1. A polynomial fit reveals that the run time T_{pr} of one complete phase ramp scales sub-linear with the number of data points N,

$$T_{pr} = 4.1 \times 10^{-7} \cdot N^{0.93} \text{ clocks}, \tag{10}$$

for arrays larger than or equal to 256×256. However, one must keep in mind that the total time required for building the complete projected potential is AT_{pr}. As shown in Fig. 2, fitting AT_{pr} for all three α-values, yields

$$AT_{pr} = 4.0 \times 10^{-7} \cdot \left(\alpha A^2\right)^{0.96} \text{ clocks}. \tag{11}$$

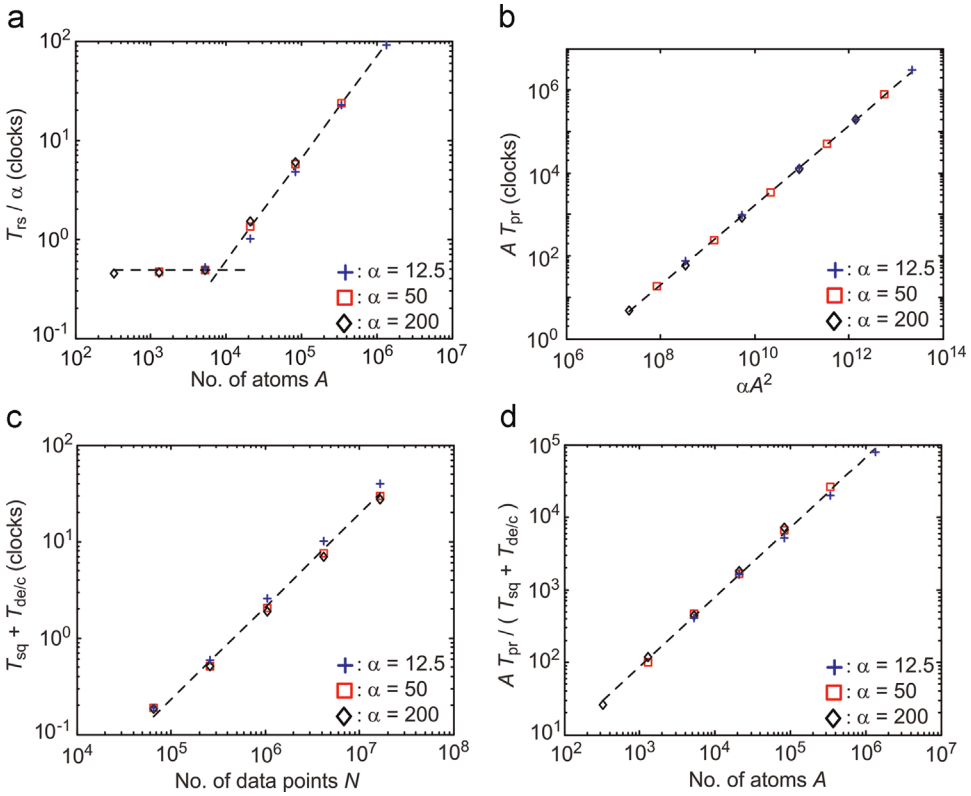

Fig. 2. (a) Normalized computation time T_{rs}/α of the real-space approach as a function of the number of atoms A for all three α-values of Table 2, see Eqs. (7) and (9) for the polynomial fit. (b) Computation time AT_{pr} of the reciprocal-space approach for all three α-values of Table 3, see Eq. (11) for the polynomial fit. (c) Computation time of the hybrid approach $T_{sq} + T_{de/c}$, see Eq. (12) for the polynomial fit. (d) The ratio of increase in computation speed between the hybrid approach and the reciprocal-space approach, defined as $AT_{pr}/(T_{sq} + T_{de/c})$, see Eq. (15) for the polynomial fit.

Table 3
Computation times T_{sq} and $T_{de/c}$ of the sq-step and the de/c-step of FDES, respectively. The functions were run E times in a row to amass a minimum of 100 clocks for the fastest processes. Note how the number of atoms A ranges from 3.3×10^2 to 1.3×10^6.

$\sqrt{N} \times \sqrt{N}$	E	A	$E \cdot T_{sq}$ (clocks)	$E \cdot T_{de/c}$ (clocks)	$T_{sq} + T_{de/c}$ (ms)
$d = 0.04$ nm, $\alpha = 12.5$					
256×256	51 200	5.2×10^3	1226	8384	0.19
512×512	12 800	2.1×10^4	1372	6223	0.59
1024×1024	3200	8.4×10^4	2341	5895	2.6
2048×2048	800	3.4×10^5	2627	5390	10.0
4096×4096	200	1.3×10^6	2701	5284	39.9
$d = 0.02$ nm, $\alpha = 50$					
256×256	51 200	1.3×10^3	1185	8424	0.19
512×512	12 800	5.2×10^3	310	6213	0.51
1024×1024	3200	2.1×10^4	608	5880	2.0
2048×2048	800	8.4×10^4	671	5372	7.6
4096×4096	200	3.4×10^5	686	5304	30.0
$d = 0.01$ nm, $\alpha = 200$					
256×256	51 200	3.3×10^2	1027	8403	0.18
512×512	12 800	1.3×10^3	295	6208	0.51
1024×1024	3200	5.2×10^3	125	5898	1.9
2048×2048	800	2.1×10^4	172	5389	7.0
4096×4096	200	8.4×10^4	171	5287	27.3

4.4. Hybrid approach

The computation times T_{sq} and $T_{de/c}$, corresponding to the sq-step and the de/c-step of the projected potential, are measured with the same test specimens as described in Section 4.2. The measured computation times are listed in Table 3.

The total computation time obeys the polynomial relationship

$$T_{sq} + T_{de/c} = 5.0 \times 10^{-6} \cdot N^{0.94} \text{ clocks}, \tag{12}$$

as shown in Fig. 2.

The computation time relative to the real-space approach varies between a minimum and a maximum of 21.5 and 493.0, respectively, and has an average and standard deviation of

$$\frac{T_{rs}}{T_{sq} + T_{de/c}} = 82.3 \pm 121.6, \tag{13}$$

when the whole dataset of the real-space approach is considered. If only data points for $A > 8.1 \times 10^3$ are considered, the ratio varies between a minimum and a maximum of 21.5 and 44.0, respectively, and has an average and standard deviation of

$$\frac{T_{rs}}{T_{sq} + T_{de/c}} = 33.3 \pm 8.5. \tag{14}$$

The increase in computation speed relative to the reciprocal-space approach is calculated as $AT_{pr}/(T_{sq} + T_{de/c})$ and is plotted in Fig. 2. Again, a polynomial dependency was observed:

$$\frac{AT_{pr}}{T_{sq} + T_{de/c}} = 0.11 \cdot A^{0.96}. \tag{15}$$

Due to the high values of A, this ratio varies between 29 and 8.1×10^4 for the test cases used in this section.

Table 4
Computation time T_{prop} of the propagation step of the MSA. The functions were run E times. From Table 3 it can be seen that T_{prop} is comparable to the computation time $T_{sq} + T_{de/c}$ of the potential in the hybrid approach.

$\sqrt{N} \times \sqrt{N}$	E	$E \cdot T_{prop}$ (clocks)	T_{prop} (ms)
256×256	25 600	5444	0.21
512×512	6400	4804	0.75
1024×1024	1600	4774	3.0
2048×2048	400	4196	10.5
4096×4096	100	4072	40.7

4.5. Propagation

For comparison, the computation time T_{prop} of the propagation of the electron wave through the object potential is measured as well. From Table 4 it can be seen that T_{prop} is comparable to the computation time $T_{sq} + T_{de/c}$ of the potential in the hybrid approach. A polynomial fit yields

$$T_{prop} = 5.7 \times 10^{-6} \cdot N^{0.95} \text{ clocks}. \tag{16}$$

The ratio $T_{prop}/(T_{sq} + T_{de/c})$ – derived from the measurements in Tables 3 and 4 and *not* from Eqs. (12) and (16) – shows a negligible dependency on N. Instead, it is more instructive to write down its average and standard deviation:

$$\frac{T_{prop}}{T_{sq} + T_{de/c}} = 1.34 \pm 0.21. \tag{17}$$

5. FDES – forward dynamical electron scattering

The large increase in computation speed brought about by the hybrid approach and parallelization on a GPU is made available as the open-software package FDES (forward dynamical electron scattering). It is written in the CUDA programming language [25], with the CUFFT, CUBLAS and CURAND libraries.

FDES is capable of simulating specimen-tilt series, beam-tilt series, focal series, or any combination thereof, of

- high resolution transmission electron microscopy (HRTEM) images,
- diffraction patterns (DP) and
- convergent beam electron diffraction (CBED) patterns.

Thermal diffuse scattering can be approximated with an absorptive potential [26] or with the frozen phonon approach (Einstein model) [27]. As mentioned in Section 2.1, the parametrization in [5] is used for the projected-potential calculations. The counting statistics are applied to the electron wave intensity right before the blurring by the CCD's modulation transfer function (MTF) [28] takes place, resulting in more realistic, correlated, noise.

Poisson distributed random values, needed for an exact treatment of the counting statistics, require preparation on the central processing unit (CPU) for each expectation value (i.e., in this case, for each pixel value) and introduce extra data transfer between CPU and GPU, thereby slowing down the process. Since normally distributed variables have no such difficulties, FDES applies the inverse Anscombe transform [29]

$$x_A = \max\left[\text{round}\left(x_N^2/4 - 3/8\right), 0\right] \tag{18}$$

to variables x_N drawn from a normal distribution with a mean and a variance of

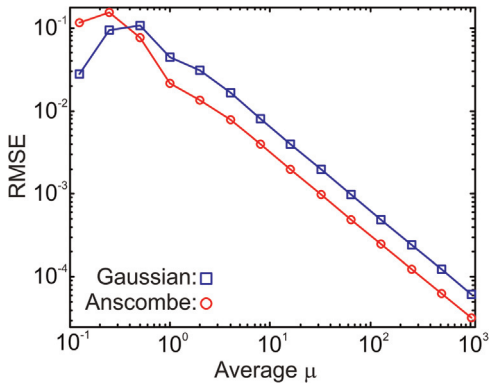

Fig. 3. RMSE of Gaussian noise and Anscombe noise as a function of expectation value μ, the ratio between both RMSEs is 2.0 on average for $\mu \geq 1$.

$$2\sqrt{\mu + 3/8} - \sqrt{\mu}/4 \quad \text{and} \tag{19}$$

$$1 - \exp(-\mu/0.777134), \tag{20}$$

respectively, to yield approximate Poisson distributed variables x_A with expectation value μ. This is dubbed Anscombe noise. The exponential term in Eq. (20) is a heuristic adaptation to extend the applicability of this approach to values of μ below 5. In Fig. 3 it is shown that for μ larger than or equal to 1 (corresponding to a signal-to-noise ratio of 1 or higher) the root mean squared error (RMSE) of Anscombe noise with respect to Poisson noise is approximately twice as small as the RMSE of Gaussian noise with mean and variance μ. For μ smaller than 0.38, Gaussian noise is preferable to Anscombe noise.

5.1. Demonstration

The capabilities of FDES are demonstrated on [001] Si. A supercell of $40 \times 40 \times 100\ \text{nm}^3$ with 8.1×10^6 atoms is constructed. Subsequently, two different amounts of bending are applied in the x-direction, amounting to a tensile strain of 1.25% and 2.5% in the upper layer of the supercell, respectively. The unbent and the two bent specimens are imaged through HRTEM and CBED.

In the x- and y-directions, the supercells are discretized into 4000×4000-matrices, corresponding to a pixel size of $0.01 \times 0.01\ \text{nm}^2$. The slice thickness for the MSA is set to 0.01 nm to avoid the systematic errors associated with too-large slice thicknesses explained in Fig. 1 of [16], resulting in 10 000 slices. Thermal diffuse scattering (TDS) is taken into account through the frozen phonon approximation with Debye–Waller factors of $5.38 \times 10^{-3}\ \text{nm}^2$ [30]. The acceleration voltage is 100 kV and no lens aberrations have been included. The MTF is set to

$$0.58\exp(-1.4s) + 0.42\exp(-3.9s^2),$$

with s in units of the sampling frequency.

This test is run on a Tesla K20c GPU (NVIDIA), which has 2496 processors and 5120 MiB of memory.

5.1.1. HRTEM

In Fig. 4 HRTEM images for the unbent and the two bent specimens are shown. For TDS, 50 frozen phonon configurations were included. The bend-contours are clearly visible. Furthermore, it is clear that TDS hardly affects the overall appearance of the images. Nonetheless, for strains of 0%, 1.25% and 2.5%, respectively, a 27%, 23% and 21% reduction in image contrast can be observed after TDS has been included. Here, image contrast is defined as the standard deviation of the image intensity taken in the central $2 \times 2\ \text{nm}^2$ region of the image.

The total runtime for the TDS calculation of the 0%-strain case is 44 571 739 clocks, which at 1 ms per clock corresponds to 12 h 23 min, or 14 min 51 s per phonon configuration. The other two cases have similar runtimes.

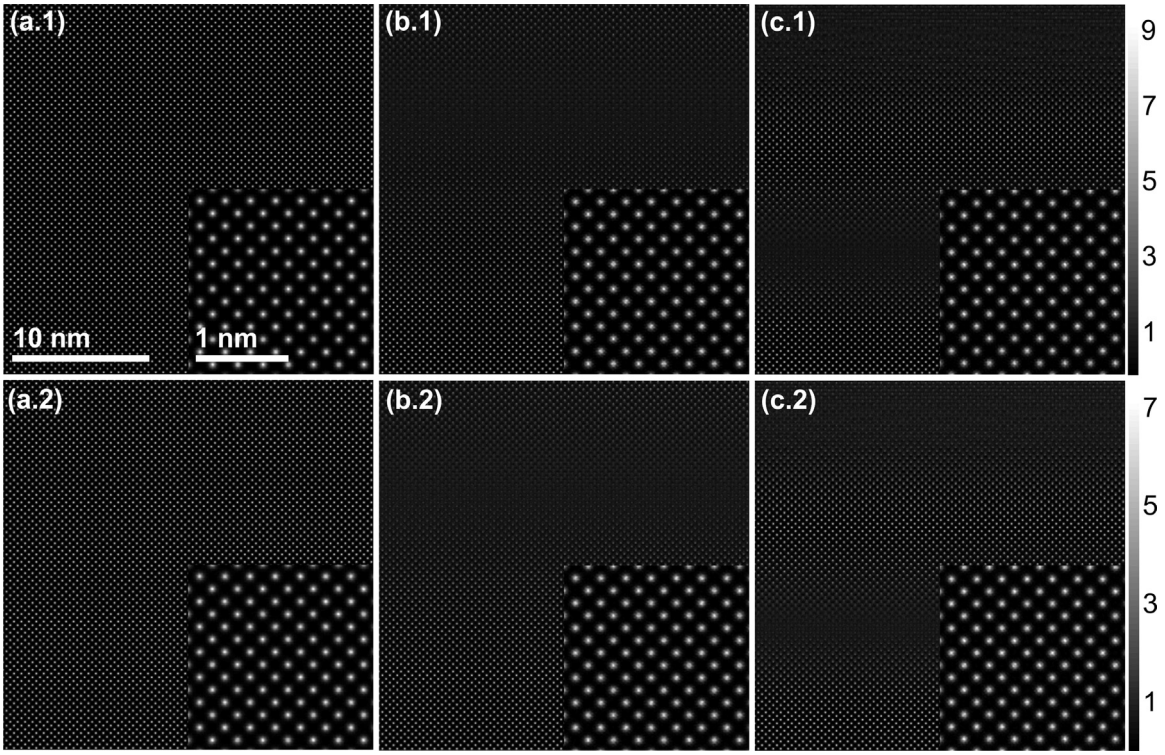

Fig. 4. Upper left quadrant of the HRTEM images of [001] Si with 0%, 1.25% and 2.5% of tensile strain in the supercell's upper layer (images a, b and c, respectively); calculated without and with TDS (images 1 and 2, respectively). The insets are the images' central $2 \times 2\ \text{nm}^2$ squares. No imaging aberrations have been applied.

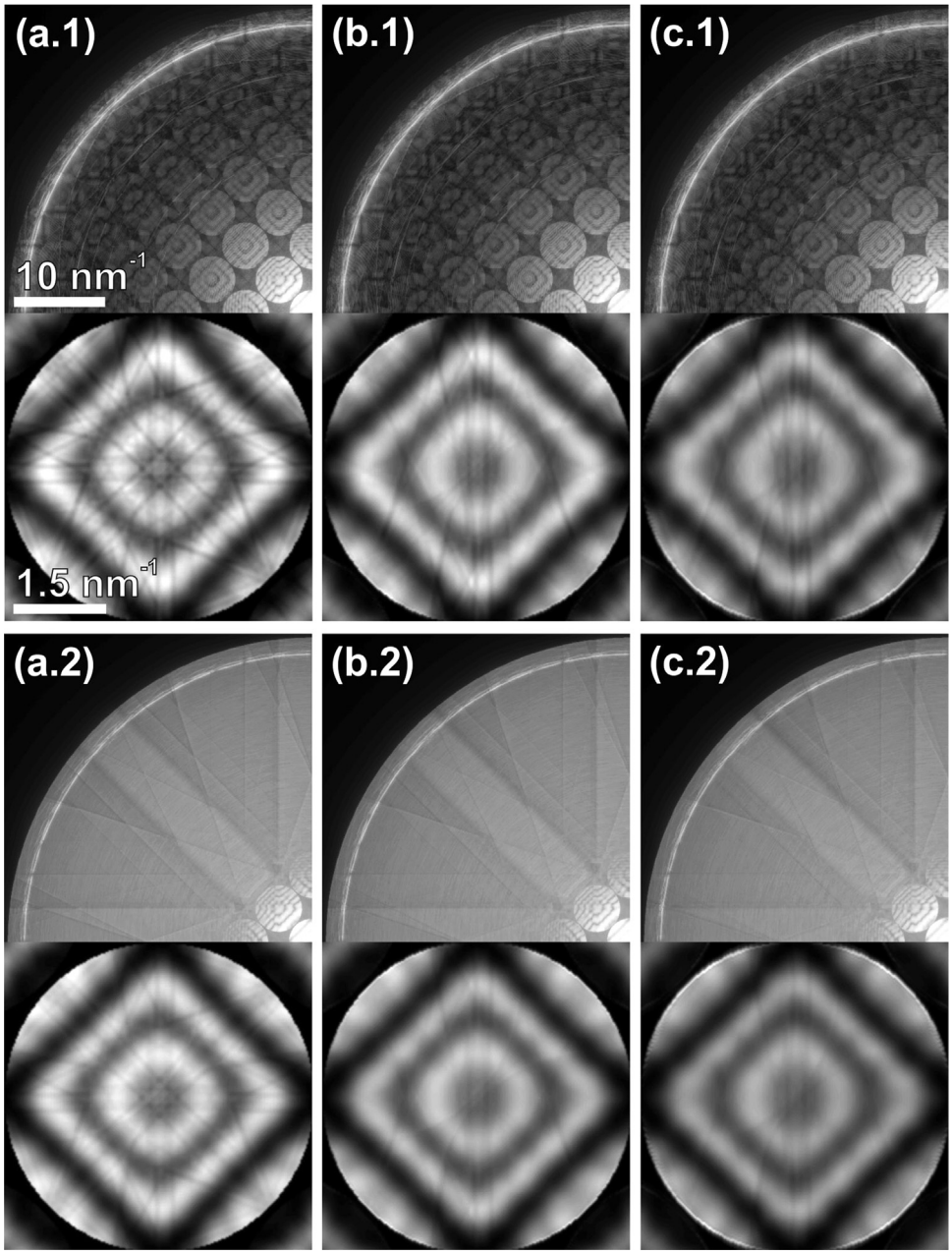

Fig. 5. The upper half of each image (logarithmic grayscale) is the left quadrant of the CBED patterns of [001] Si with 0%, 1.25% and 2.5% of tensile strain in the supercell's upper layer (images a, b and c, respectively); calculated without and with TDS (images 1 and 2, respectively). The lower halves (linear grayscale) are the patterns' central discs.

5.1.2. CBED

The convergence angle for CBED is set to 9.4 mrad. In Fig. 5 the diffraction patterns for the unbent and the two bent specimens are shown. TDS is included by means of 50 frozen phonon configurations. Especially the central CBED discs show the influence of the specimen bending. Furthermore, TDS has a profound influence on the quality of the diffraction patterns as the resulting Kikuchi bands drown out many of the features at higher frequencies.

The total runtime for the calculations is similar to that of HRTEM.

5.2. Technical specifications and availability

FDES is cross-platform supported and open source software under the GNU-license [31]. The source code is available in the github repository https://github.com/woutervandenbroek/FDES. A comprehensible building manual for Linux and Windows systems is provided.

Three different file formats are accepted as input: simple text files, QSTEM configuration files (qsc-files) [6] and electron microscopy dataset (EMD) files [32]. The text files allow input of structures obtained from virtually any simulation software, as long as a list of the atoms' Cartesian coordinates is available. qsc-files allow the user to use the QSTEM model-builder to build complicated specimen models based on experimental TEM images. The open file-format EMD is based on HDF5 [33], which is a flexible, open, free and widely used data format. A working example of each is included.

The results can be output as binary data files, ensuring compatibility with virtually any image processor further down the line. EMD files are used as output too, allowing for easy data visualization with the EMD viewer [32] or any HDF5 file viewer.

Furthermore, the EMD viewer allows easy editing of the simulation parameters, thereby facilitating the usual approach in image simulations where the parameters are manually tuned over a few iterations.

6. Discussion

Calculation of the projected potential in real space is linear in N, but suffers from large pre-factor as r_c must be chosen of considerable size (typically 0.5 nm) to prevent scattering from the artificial discontinuity at the cut-off. Furthermore, contrary to the hybrid approach presented in this paper, one faces a trade-off between accuracy and computation speed.

The two different regimes observed in the real-space approach's computation time probably stem from an incomplete parallelization of the algorithm, i.e. at lower numbers of atoms A a portion of the processors is idle while the other processors compute all values within r_c serially. An improved implementation could therefore be expected to have a better performance at low A, comparable to that of higher A. It is for that reason that the relative speed increase for the whole dataset and for $A > 8.1 \times 10^3$ are given separately in Eqs. (13) and (14).

Contrary to what one might expect, by using a precalculated potential the real-space method did not improve for large A and N. The most likely reason is that many processes are accessing the same addresses in the array with the precalculated projected potential. Such addressing-conflicts are a common issue in GPU-based calculations.

Since the computation time of the reciprocal-space approach is dominated by the phase-ramp calculations, the presence of multiple elements in the specimen does not affect it much. The sq-step of the hybrid approach is not affected either, but the de/c-step needs to be repeated for each different chemical element. Since the de/c- step is dominant (see Table 3) the total computation time scales approximately proportional to the number of elements.

The fact that the projected-potential calculation is about as fast as the wave propagation (see Eq. (17)) opens up the possibility of on-the-fly potential calculation, thereby offering an alternative for memory intensive storage of the potential or time consuming read/write operations from disk; a feature that is especially interesting for STEM calculations of large models [6].

In order to assess the inherent performance of the various approaches in Section 4, operations that are common to all approaches, like array allocations, reading and writing, data transfer between CPU and GPU, etc., were excluded from the measurements. Furthermore, in some of the computationally most expensive iterative algorithms (e.g. fitting an atomic structure to experimental data) these latter operations only have to be performed once. The calculation times reported for the HRTEM and CBED demonstration in Section 5.1 do include all these operations and are therefore relatively large.

7. Conclusions

In this paper three different methods for calculating the projected electrostatic potential, a prerequisite for multislice calculations, have been compared: a real-space approach, a reciprocal-space approach and a hybrid approach. It was shown that the reciprocal-space approach has a computational complexity that scales approximately quadratically with the number of atoms in the specimen, while the real-space approach and the hybrid approach scale approximately linearly. These theoretical assertions bore out in practice when tested on an amorphous carbon specimen. It was shown that the hybrid approach was faster than the real-space approach by a factor ranging from 22 to 44.

Furthermore, FDES – forward dynamical electron scattering – is introduced, a multislice algorithm that runs on a graphics processing unit and that implements the hybrid approach to calculate the electrostatic potential, thus resulting in a large increase in computation speed and thus enabling on-the-fly potential calculations. FDES is capable of simulating specimen-tilt series, beam-tilt series, focal series, or any combination thereof, of

- high resolution transmission electron microscopy images,
- diffraction patterns and
- convergent beam electron diffraction patterns.

FDES is open source software, available under the GNU-license.

A demonstration on a [001] Si supercell with 8.1×10^6 atoms showed that images of this large system could be computed in under 15 min on a desktop computer equipped with a graphics processing unit.

Acknowledgments

W. Van den Broek and X. Jiang acknowledge financial support by the Carl Zeiss Foundation. C.T. Koch acknowledges the German Research Foundation (DFG, Grant no. KO 2911/7-1) and the Carl Zeiss Foundation. The authors acknowledge C. Ophus for his help with the EMD file format.

References

[1] J.M. Cowley, A.F. Moodie, The scattering of electrons by atoms and crystals. I. A new theoretical approach, Acta Cryst. 10 (10) (1957) 609–619.
[2] J.G. Allpress, E.A. Hewat, A.F. Moodie, J.V. Sanders, *n*-Beam lattice images. I. Experimental and computed images from $W_4Nb_{26}O_{77}$, Acta Cryst. A28 (1972) 528–536.
[3] D.F. Lynch, M.A. O'Keefe, *n*-Beam lattice images. II. Methods of calculation, Acta Cryst. A28 (1972) 536–548.
[4] P. Goodman, A.F. Moodie, Numerical evaluation of N-beam wave functions in electron scattering by the multislice method, Acta Cryst. A30 (1974) 280–290.
[5] E.J. Kirkland, Advanced Computing in Electron Microscopy, 2nd ed., Springer, New York, Dordrecht, Heidelberg, London, 2010.
[6] C.T. Koch, Determination of core structure periodicity and point defect density along dislocations (Ph.D. thesis), Arizona State University, Phoenix, Arizona, 2002.
[7] A. Rosenauer, M. Schowalter, STEMSIM—a new software tool for simulation of STEM HAADF Z-contrast imaging, in: A.G. Cullis, P.A. Midgley (Eds.), Microscopy of Semiconduction Materials, 2007, pp. 170–172.
[8] P. Stadelmann, jems website, URL: ⟨http://cimewww.epfl.ch/people/stadelmann/jemsWebSite/jems.html⟩.
[9] Total Resolution LLC, Total Resolution Home Page, URL: ⟨http://www.totalresolution.com/⟩.
[10] C. Dwyer, Simulation of scanning transmission electron microscope images on desktop computers, Ultramicroscopy 110 (3) (2010) 195–198.
[11] M.J. Humphry, B. Kraus, A.C. Hurst, A.M. Maiden, J.M. Rodenburg, Ptychographic electron microscopy using high-angle dark-field scattering for sub-nanometre resolution imaging, Nat. Commun. 3 (2012) 730.
[12] W. Van den Broek, C.T. Koch, Method for retrieval of the three-dimensional object potential by inversion of dynamical electron scattering, Phys. Rev. Lett. 109 (2012) 245502.
[13] W. Van den Broek, C.T. Koch, General framework for quantitative three-dimensional reconstruction from arbitrary detection geometries in TEM, Phys. Rev. B 87 (2013) 184108.
[14] C.T. Koch, W. Van den Broek, Measuring three-dimensional positions of atoms to the highest accuracy with electrons, C. R. Phys. 15 (2014) 119–125.
[15] S.P. Frigo, Z.H. Levine, N.J. Zaluzec, Submicron imaging of buried integrated circuit structures using scanning confocal electron microscopy, Appl. Phys. Lett. 81 (2002) 2112–2114.
[16] A. Chuvilin, U. Kaiser, On the peculiarities of CBED pattern formation revealed by multislice simulation, Ultramicroscopy 104 (2005) 73–82.
[17] A.S. Eggeman, A. London, P.A. Midgley, Ultrafast electron diffraction pattern simulations using GPU technology. Applications to lattice vibrations, Ultramicroscopy 134 (2013) 44–47.
[18] R.S. Pennington, F. Wang, C.T. Koch, Stacked-Bloch-wave electron diffraction simulations using GPU acceleration, Ultramicroscopy 141 (2014) 32–37.
[19] L.J. Allen, A.J. D'Alfonso, S.D. Findlay, Modelling the inelastic scattering of fast

electrons, Ultramicroscopy 151 (2015) 11–22.
[20] I. Lobato, D. Van Dyck, MULTEM: a new multislice program to perform accurate and fast electron diffraction and imaging simulations using Graphics Processing Units with CUDA, Ultramicroscopy (2015) 9–17.
[21] W. Van den Broek, C.T. Koch, The forward dynamical electron scattering (FDES) software; a graphics-processing-unit accelerated multislice algorithm, in: Proceedings of the 18th International Microscopy Congress, CSMS & IFSM, 2014, pp. IT–16–P–2748.
[22] F.W.J. Olver, Bessel functions of integer order, in: M. Abramowitz, I.A. Stegun (Eds.), Handbook of Mathematical Functions, With Formulas, Graphs, and Mathematical Tables, 10th ed., National Bureau of Standards, USA, 1972.
[23] D. Van Dyck, W. Coene, The real space method for dynamical electron diffraction calculations in high resolution electron microscopy, Ultramicroscopy 15 (1984) 29–40.
[24] C. Wacker, R.R. Schröder, Multislice algorithms revisited: solving the Schrödinger equation numerically for imaging with electrons, Ultramicroscopy 151 (2015) 211–223.
[25] NVIDIA, CUDA Toolkit Documentation, v6.5 ed., URL: ⟨http://docs.nvidia.com/cuda/#axzz3EKwkzsvA⟩, 2014.
[26] A. Weickenmeier, H. Kohl, Computation of absorptive form factors for high-energy electron diffraction, Acta Cryst. A 47 (1991) 590–597.
[27] Z.L. Wang, The 'frozen-lattice' approach for incoherent phonon excitation in electron scattering. How accurate is it? Acta Cryst. Sect. A 54 (4) (1998) 460–467.
[28] W. Van den Broek, S. Van Aert, D. Van Dyck, Fully automated measurement of the modulation transfer function of charge coupled devices above the Nyquist frequency, Microsc. Microanal. 18 (2012) 336–342.
[29] F.J. Anscombe, The transformation of Poisson, binomial and negative-binomial data, Biometrika (1948) 246–254.
[30] M. Schowalter, A. Rosenauer, J.T. Titantah, D. Lamoen, Computation and parametrization of the temperature dependence of Debye–Waller factors for group IV, III–V and II–VI semiconductors, Acta Cryst. A65 (2009) 5–17.
[31] Free software foundation, Licenses – gnu project – free software foundation, URL: ⟨http://www.gnu.org/licenses/licenses.en.html⟩.
[32] C. Ophus, P. Ophus, P. Ercius, Electron microscopy datasets, URL: ⟨http://emdatasets.lbl.gov/⟩.
[33] The HDF Group, Hierarchical Data Format, version 5 ⟨http://www.hdfgroup.org/HDF5/ ⟩, 1997–2015.