$Revision: 1.18 $Date: 2010/01/13 15:16:41

Interpolation of pixelized images in Fourier domain

Gary M. Bernstein[1], Daniel Gruen[1,2]

garyb@physics.upenn.edu

ABSTRACT

In the analysis or simulation of sky images, one often takes a galaxy or point-spread function (PSF) $F(x)$ observed as a grid of samples, and wishes to know its value $\tilde{F}(u)$ in Fourier domain at arbitrary frequency u. A key example is the simulation of real galaxy images as they would appear under weak gravitational lensing shear and convolution by a known PSF. We show that a wrapped sinc function is the exact solution for interpolation of $\tilde{F}(u)$, and investigate approximations that have smaller, more practical kernel footprints than the sinc function. We show that these approximations produce a multiplicative error plus a pair of ghost images (in each dimension) in the simulated image. We recommend a scheme whereby the input data are zero-padded by a factor 4 before a discrete Fourier transform, and a custom-designed 6-point polynomial kernel used to interpolate in u space. This scheme limits the interpolation errors to < 1 part in 1000 of the original flux. In the worst circumstances, when the ghost images are folded to be nearly atop the original image, $6\times$ zero-padding may be needed in order that simulated image shear be produced to better than part-per-thousand accuracy.

Subject headings: Data Analysis and Techniques

1. Introduction

In real images, one obtains a finite, pixelized (*i.e.* sampled) rendition of objects, for example the point spread function (PSF) from stellar images, or galaxy images. Many forms of subsequent analysis require continuum representations of the objects or their Fourier transforms, for example to resample the objects onto a new grid, or to predict the appearance of the object after rotation, distortion, or convolution with a new PSF. The development of weak gravitational lensing techniques requires that we perform these operations with great accuracy.

[1]Dept. of Physics and Astronomy, University of Pennsylvania, Philadelphia, PA 19104

[2]University Observatory Munich, Scheinerstrasse 1, 81679 Munich, Germany

A first question arises because the PSF of an exposure is often estimated empirically by fitting stellar images with a model consisting of a grid of pixel values and a specified interpolation function, *e.g.* as done by the PSFEx software (Bertin 2011). To recover the intrinsic galaxy shape, the observed shape must be corrected for the PSF, an operation that is most straightforwardly done in the Fourier domain (Bernstein 2010), and requires part-per-thousand accuracy in the PSF representation (Huterer *et al.* 2006; Amara & Réfrégier 2008). One question, therefore, is how to compute the Fourier transform of a PSF defined by interpolation on a grid of samples. A simple discrete Fourier tranform (DFT) is insufficient, as it represents a periodic PSF, and does not include the effects of the interpolation function.

A second question arises when simulating the effect of weak gravitational lensing on real galaxies. This requires taking pixelized images of the real galaxies and calculating their appearance after application of a lensing shear, convolution with a new PSF, and sampling on a new pixel grid, *e.g.* Mandelbaum et al. (2011). Rotation, distortion, or an incommensurate re-pixelization require interpolation in either real or Fourier space, and the application of convolution a favors Fourier-domain solution. Hence we ask what schemes for interpolation of the Fourier representation of the galaxy (between DFT samples) are needed to produce a simulated sheared, resampled galaxy at part-per-thousand level.

2. Transform of interpolated, sampled image

We assume that an object is correctly modeled as an interpolation of a square, finite grid of values, so that its surface brightness $F(x,y)$ is defined by the finite set of values a_{ij} for $-N/2 \leq i,j < N/2$ (we assume N is even; an odd-valued N can be padded with another row and column of zeros). Given an interpolation kernel $K(x,y)$,

$$F(x,y) \equiv \sum_{i,j=-N/2}^{N/2-1} a_{ij} K(i-x, j-y). \tag{1}$$

We will assume that the interpolation kernel has $K(0,0)=1$ and $K(m,n)=0$ at integer m,n other than the origin—this is the condition that the interpolation agree with the input samples at the sampled locations. We will assume that the kernel is even, $K(x,y)=K(-x,-y)$. F is zero beyond a bounded region if K is. We wish to know the Fourier transform

$$\tilde{F}(u,v) = \int dx\, dy\, F(x,y) e^{-2\pi i(ux+vy)}. \tag{2}$$

For notational simplicity we solve the one-dimensional case, which extends easily to two or more dimensions.

$$F(x) \quad \equiv \quad \sum_j a_j K(x-j) \tag{3}$$

$$\begin{aligned}
&= \sum_j \int dx' a_j \delta(x'-j) K(x-x') &(4)\\
&= (f*K)(x), &(5)\\
f(x) &\equiv \sum_{j=-N/2}^{N/2-1} a_j \delta(x-j). &(6)
\end{aligned}$$

The $*$ indicates convolution, and $f(x)$ is the original sampled function. From the convolution theorem, we have $\tilde{F}(u) = \tilde{f}(u)\tilde{K}(u)$. Rewriting the delta function at j, with $|j| \leq N/2$, as a product of a boxcar and a comb function (our conventions for Fourier transforms and functions are in the Appendix), we can transform $f(x)$ as

$$\begin{aligned}
\delta(x-j) &= \Pi(x/N)\frac{1}{N}\text{Ш}\left(\frac{x-j}{N}\right) &(7)\\
\Rightarrow \tilde{f}(u) &= N\text{sinc}(uN) * \left[\sum_{j=-N/2}^{N/2-1} a_j e^{-2\pi i j u}\text{Ш}(uN)\right] &(8)\\
&= \sum_{k=-\infty}^{\infty} \text{sinc}(k-Nu)\tilde{a}_k &(9)
\end{aligned}$$

where we recognize the discrete Fourier transform (DFT) of the a_j as

$$\tilde{a}_k = \sum_{j=-N/2}^{N/2-1} a_j e^{-2\pi i jk/N}. \tag{10}$$

Since we know the DFT is periodic, with $\tilde{a}_{k+N} = \tilde{a}_k$, we can write the transform of $F(x)$ as

$$\tilde{F}(u) = \tilde{K}(u) \sum_{j=-N/2}^{N/2-1} \left\{ \tilde{a}_j \sum_{k=-\infty}^{\infty} \text{sinc}[N(k+j/N-u)] \right\}. \tag{11}$$

As a further simplification, we note that the sum over sinc values can be done analytically:

$$\sum_{k=-\infty}^{\infty} \text{sinc}[N(k-v)] = \frac{\sin(N\pi v)}{N\sin(\pi v)} \times \begin{cases} \cos(\pi v) & N \text{ even} \\ 1 & N \text{ odd} \end{cases} \tag{12}$$

where we have assigned $v = j/N - u$ as the difference between the desired frequency u and the $u_j = j/N$ represented by the jth DFT coefficient. The exact expression (11) for $\tilde{F}(u)$ can thus be expressed by this recipe:

1. Obtain the $\tilde{a}_j$ from the input a_j via an N-point DFT.
2. Interpolate from the grid values $\tilde{a}_j$ at $u_j = j/N$ to the desired frequency u using sinc interpolation, wrapping the sinc function around the $u \in [-1/2, 1/2)$ region of the DFT.

3. Multiply the interpolated value by the transform $\tilde{K}(u)$ of the original x-space interpolant.

The second step is slow, since each output value $\tilde{F}(u)$ requires a sum over all N of the DFT coefficients (N^2 in two dimensions). We will examine below the consequences of approximating the sinc interpolation with a more compact interpolation kernel.

Equation (13) can be extended to two (or more) dimensional data by using the 2d DFT of the input data a_{ij} and the transform of the 2d interpolation kernel $K(x, y)$—which is often simply the product of 1d kernels in each variable—and multiplying by sinc functions in each dimension.

3. Interpolator accuracy

Consider an interpolation kernel $K(t)$, with Fourier transform $\tilde{K}(\nu)$, that is used reconstruct the value of a sine wave $\exp(2\pi i \nu_0 t)$ from samples at integral t values to some non-integral t_0. The sinc function has the unique property that it interpolates without error for any frequency $-1/2 < \nu_0 < 1/2$. Consider more generally an interpolation kernel that is symmetric, $K(-t) = K(t)$, such that $\tilde{K}(\nu)$ is real and symmetric, and is known to be exact at the interpolation nodes, *i.e.* for integral arguments j,

$$K(j) = \begin{cases} 1 & j = 0 \\ 0 & j \neq 0 \end{cases} \tag{13}$$

The interpolated value of the sine wave is

$$\begin{aligned}
R(\nu_0, t_0) &\equiv \sum_{j=-\infty}^{\infty} K(j - t_0) e^{2\pi i \nu_0 j} && (14)\\
&= \left[e^{2\pi i \nu_0 t} \text{Ш}(t) \right] * K(t) \Big|_{t_0} && (15)\\
\Rightarrow \tilde{R}(\nu_0, \nu) &= \left[\delta(\nu - \nu_0) * \text{Ш}(\nu) \right] \tilde{K}(\nu) && (16)\\
&= \sum_{j=-\infty}^{\infty} \tilde{K}(\nu) \delta(\nu - \nu_0 - j) && (17)\\
\Rightarrow R(\nu_0, t_0) &= \sum_{j=-\infty}^{\infty} \tilde{K}(\nu_0 + j) e^{2\pi i (\nu_0 + j) t_0} && (18)\\
&= e^{2\pi i \nu_0 t_0} \sum_{j=-\infty}^{\infty} \tilde{K}(\nu_0 + j) e^{2\pi i j t_0}. && (19)
\end{aligned}$$

The prefactor is the correctly interpolated sine wave, so let us define an error function $E(\nu_0, t_0) \equiv e^{-2\pi i (\nu_0 + j) t_0} R(\nu_0, t_0) - 1$. Then this error function is

$$\begin{aligned}
E(\nu_0, t_0) &= \sum_{j=-\infty}^{\infty} \tilde{K}(\nu_0 + j) e^{2\pi i j t_0} && (20)\\
&= \sum_{j \neq 0} K(\nu_0 + j) \left(e^{2\pi i j t_0} - 1 \right) && (21)
\end{aligned}$$

In the second line we have made use of the fact that Equation (13) requires $E(\nu_0, 0) = 0$. The sinc filter has $\tilde{K}(\nu) = \Pi(\nu)$ which vanishes for $|\nu| > \frac{1}{2}$. The E function hence vanishes for $|\nu_0| < \frac{1}{2}$, as expected.

3.1. Background conservation

Many astronomical images have signals atop a large constant ($\nu = 0$) background. It is hence important that $E(0, t_0) = 0$, otherwise a resampled image will have background fluctuations as t_0 varies. Note also that the same criterion dictates whether object flux will be conserved when the interpolant is used to shift the samples by a constant fraction t_0 of a pixel. Putting $\nu_0 = 0$ in Equation (21), the fractional background fluctuations will be

$$\begin{aligned} E(0, t_0) &= 2\sum_{j=1}^{\infty} \tilde{K}(j)\,(\cos 2\pi j t - 1) & (22) \\ &\approx 2\tilde{K}(1)\,(\cos 2\pi t_0 - 1)\,. & (23) \end{aligned}$$

From the first line we can see that any interpolant having $\tilde{K}(j) = 0$ for $j \neq 0$ will conserve background level. The nearest-neighbor and linear filters (see Appendix for definitions) satisfy this exactly since $\tilde{K}(\nu) = \mathrm{sinc}(\nu)$ and $\mathrm{sinc}^2(\nu)$, respectively, and the polynomial interpolants are designed to meet this criterion. The Lanczos interpolants do not, however, meet this criterion. In the second line we have assumed that we are using interpolants, like Lanczos, that attempt to approximate the band-limiting properties of the sinc filter, and will hence have $|\tilde{K}(1)| \ll 1$ and $|\tilde{K}(j)| \ll |\tilde{K}(1)|$ for $|j| \geq 2$. In this case we can see that

- The interpolated background error will oscillate as $\cos 2\pi t_0 - 1$, *i.e.* will be worst for interpolation to the $t_0 = 0.5$ midpoint between pixels, and
- the maximum fractional background error will be $-4\tilde{K}(1)$.

In practice, the Lanczos interpolant or any other can be normalized to conserve DC background via

$$\begin{aligned} K(t) &\rightarrow K(t)/(1 + E(0, t)) & (24) \\ &\approx K(t)\left[1 - 2\tilde{K}(1)\,(\cos 2j\pi t_0 - 1)\right] & (25) \\ \Rightarrow \tilde{K}(\nu) &\rightarrow \left[1 + 2\tilde{K}(1)\right]\tilde{K}(\nu) - \tilde{K}(1)\left[\tilde{K}(\nu+1) + \tilde{K}(\nu-1)\right]. & (26) \end{aligned}$$

The effect of enforcing background conservation on the interpolant is hence to degrade slightly the band-limiting characteristic of the filter, adding $O(\tilde{K}(1))$ “wings” to the frequency responses that extend to $|\nu| = \pm 1.5$.

3.2. Interpolation in Fourier domain

In §2 we inferred that a wrapped sinc interpolation on the grid of DFT Fourier coeffients at $u_j = j/N$ is the correct way to calculate the transform $\tilde{F}(u)$ at values of $u \neq u_j$. Here we examine the errors from approximating the sinc with a smaller interpolation kernel. Let us keep in mind that we now have two potentially distinct interpolants: the one used in real space to define the function $F(x)$ from the samples a_j, and another one used in Fourier domain to approximate the sinc interpolation $\tilde{f}(u)$ between the sample $\tilde{a}_j$ at $u_j = j/N$. We will denote these as K_x and K_u, respectively, when they may be confused.

Consider the case when the input a_j are zero except for a single element $a_n = 1$. The DFT produces $\tilde{a}_j = \exp(-2\pi i j n/N)$, a sine wave advancing by an amount $\nu_0 = n/N$ cycles per sample, which will be perfectly interpolated to the expected $\tilde{f}(u) = \exp(-2\pi i u n)$ at any $|u| < 0.5$ if we use the sinc kernel.

With an imperfect approximation to the sinc, however, we can use Equation (20) to infer the error in the interpolated sine wave, putting $\nu_0 = n/N$ and $t_0 = uN$:

$$\tilde{f}(u) \quad \rightarrow \quad [1 - E_0(n/N)]\, \tilde{f}(u) + \sum_{j\neq 0} \tilde{K}(j + n/N) e^{2\pi i j N u} \tilde{f}(u) \tag{27}$$

$$E_0(\nu) \quad \equiv \quad \sum_{j\neq 0} \tilde{K}(j+\nu) = \sum_{j>0} \left[\tilde{K}(j+\nu) + \tilde{K}(j-\nu)\right]. \tag{28}$$

Upon transforming back to real space, the first term will yield a scaled version of the input function, while the sum becomes a series of ghost images at distance jN from the original point:

$$f(x) \rightarrow [1 - E_0(n/N)]\, \delta(x-n) + \sum_{j\neq 0} \tilde{K}(j + n/N) \delta(x - n - jN). \tag{29}$$

Now we may generalize to an arbitrary configuration of input samples a_j. The $\hat{f}(x)$ that is obtained by executing an inverse transform after interpolation of the $\tilde{a}_j$ will be

$$\hat{f}(x) = [1 - E_0(x/N)]\, f(x) + \sum_{j\neq 0} \tilde{K}(j + x/N) f(x - jN). \tag{30}$$

These alterations to $f(x)$ will be essentially preserved when convolved with K_x to produce the function $F(x)$ and its approximation $\hat{F}(x)$, namely:

- The approximated version will be multiplied by the function $1 - E_0(x/N)$.
- A series of ghosts images appear, displaced by jN from $F(x)$, and each multiplied by the function $\tilde{K}_u(j + x/N)$.

We note further that $|x/N| < 0.5$, and we are choosing interpolants intended to approximate the band-limited behavior $\tilde{K}(\nu) = 0$ for $|\nu| > 0.5$, so it is generally true that $|\tilde{K}_u(j \pm x/N)| \ll |\tilde{K}_u(1 \pm x/N)|$ for $j > 1$. In conditions considered here, the first ghost image dominates.

Figure 1 plots the function $E_0(u)$ that describes the multiplicative errors in the central image of the reconstructed $\hat{F}(x)$, for the interpolants cataloged in the Appendix. Table 1 lists the interpolants, their kernel sizes, and their error levels at chosen maximum values of x/N. Clearly any of these interpolants will accrue errors $\gg 1\%$ for $|x/N| > 0.3$. We can hold $|x/N| < 0.25$ by zero-padding the initial array by a factor 2 before the intial DFT. The Lanczos interpolants are designed to minimize $\tilde{K}(\nu)$ for $|\nu| > 0.5$ and hence perform well in reducing the interpolation errors determined by $\tilde{K}(\nu)$ for $0.75 < |x/N| < 1.25$. The Lanczos filters of order ≥ 4 attain $|E_0(x/N)| < 0.005$ with 2-fold padding.

Equivalent precision is available from the smaller cubic interpolant kernel if we precompute the DFT with 4-fold zero-padding to keep $|x/N| < \frac{1}{8}$. In this range we note that the cubic interpolant performs just as well as the Lanczos interpolants with larger (hence slower) kernels. This is attributable to the cubic interpolant being designed to be exact for polynomials of order ≤ 2, which is equivalent to the requirement that $(d/d\nu)^m \tilde{K}(\nu) = 0$ for integral ν and $m \leq 2$. Thus the cubic interpolant has $\tilde{K}(\nu)$ smaller than a Lanczos interpolant of equal size if we stay sufficiently close to $\nu = j$. The cubic interpolant is not as closely band-limited as the Lanczos filter, but the defects are kept away from the integral values ν.

Inspired by the success of the standard cubic interpolant, we seek an interpolant that is exact for polynomial functions beyond quadratic order. This will require a 6-point interpolant, which must in turn be a piecewise-quintic polynomial function. This quintic filter, described in the Appendix, also has continuous second-order derivatives. The quintic interpolant indeed satisfies our expectation of performing extremely well if we confine the data to $\nu = x/N < 0.125$ by a 4-fold zero padding of the data before the DFT. The multiplicative error $E_0(x/N)$ is $< 5 \times 10^{-4}$ in this case.

Figure 1 plots $\tilde{K}(1+u)$ for the interpolants, which determines the relative brightness of the first ghost image. We find essentially the same criteria on padding and choice of interpolant as the size of E_0. This is assured, as $E_0(\nu) \approx \tilde{K}(1-u) + \tilde{K}(1+u)$, given that all filters beyond the linear one have $|\tilde{K}(2+u)| < 0.001$ for $|u| < 0.25$. With 4-fold zero-padding, the polynomial filters are again the most efficient at reducing the amplitude of the ghost images. The 4-element cubic filter limits the ghost amplitude to 0.006 of the original a_j, and the 6-element quintic filter reduces the ghost amplitude to 0.0012.

Limiting both the amplitude of the ghost images and the multiplicative error on the central image of $\hat{F}(x)$ to be < 0.001 is hence achieved by 4-fold zero padding of the input array, with a DFT followed by interpolation in u-space with the 6-point quintic interpolant. Similar performance is possible with the more compact 4-point cubic interpolant if the initial DFT has 6-fold zero padding.

Figure 2 shows the errors induced in reconstruction of a 2d bullseye image using three different u-space interpolation schemes.

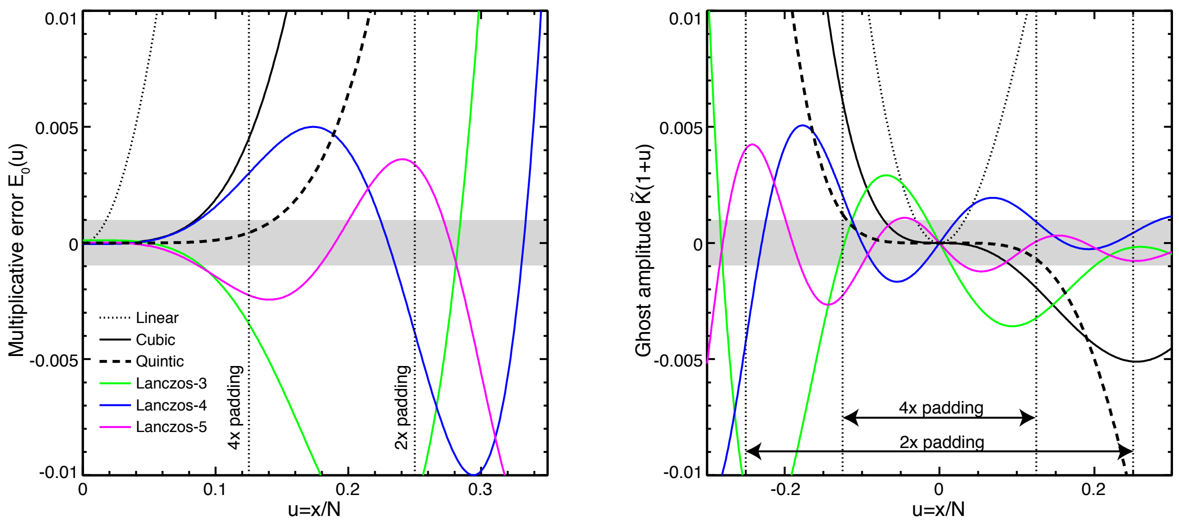

Fig. 1.— Error functions for several common interpolants. The left panel plots the multiplicative error $E_0(x/N)$ induced by approximation of the Fourier-domain sinc interpolation with the selected interpolant. The right panel plots the function $\tilde{K}(1 + x/N)$ that gives the amplitude of the first, dominant, ghost image that is produced by using the chosen u-space interpolant. The vertical dotted lines show the maximum x/N that will be present when the input data are zero-padded by a factor 2 or 4 before initial DFT. The horizontal grey band shows, for reference, errors of ± 1 part per thousand. The polynomial filters (in black) obtain lower errors than the Lanczos filters (in colors) for given kernel size if $|u| < 0.125$ is enforced by 4-fold zero padding.

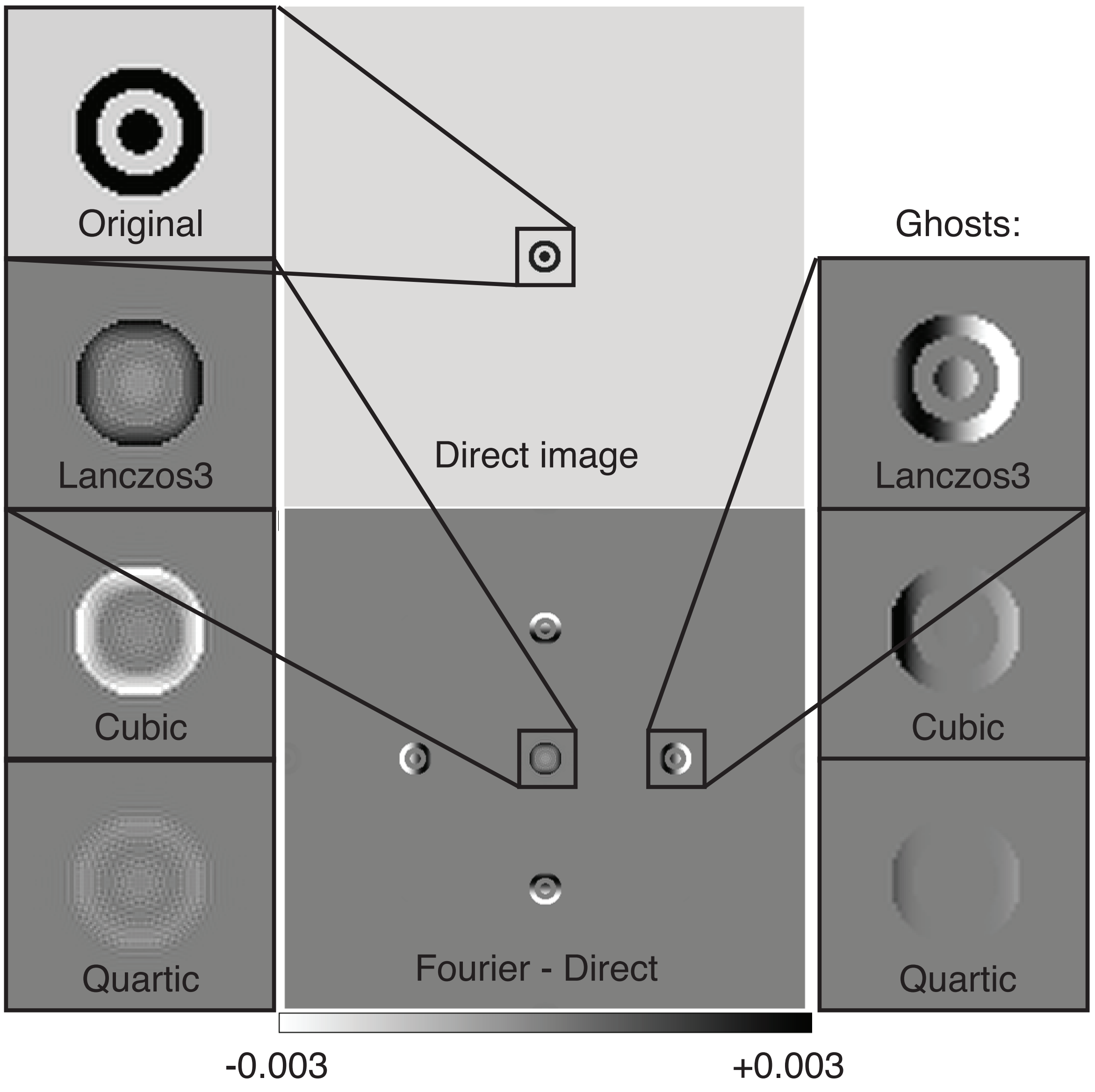

Fig. 2.— Errors induced by interpolation approximations in Fourier domain reconstruction of a 2d image. The original image is a 32×32 pixel bullseye pattern with unit amplitude. The upper middle image, with close-up in upper left, is a $2\times$ oversampled rendition using 3rd-order Lanczos interpolation. The original image is then zero-padded to 128×128 pixels, DFT'ed to Fourier domain, interpolated to a new grid in Fourier domain with a 3rd-order Lanczos filter, and transformed back to real space. The bottom middle panel is the difference between this reconstruction and the direct-interpolation image, showing the scaling error on the central image and the 4 dominant ghost images. The greyscale on the error images extends to $\pm 0.003\times$ the input image's brightness. At left is the error in the central image; at right is a closeup of the right-hand ghost image. Changing the Fourier-domain interpolant from the 6×6 Lanczos3 kernel, to a 4×4 cubic interpolant, to a 6×6 quintic interpolant leads to progressively smaller interpolation errors.

Table 1. Properties of Interpolants

Name	$N_{\rm points}$	$u_{\rm max}$:	Max reconstruction error for:		
			$x/N < 1/4$	$x/N < 1/8$	$x < 1/12$
Nearest	1	317.5	(large)	(large)	(large)
Linear	2	9.6	0.18	0.049	0.022
Cubic	4	2.74	0.061	0.0061	0.0016
Quintic	6	3.62	0.037	0.0012	0.00015
Lanczos $n = 3$	6	1.49	0.014	0.0035	0.0035
Lanczos $n = 4$	8	1.35	0.005	0.0030	0.0019
Lanczos $n = 5$	10	1.08	0.004	0.0022	0.0012
Sinc	∞	0.5	0	0	0

Note. — Interpolants are defined in the Appendix. $N_{\rm points}$ is the number of grid points per dimension summed during the interpolation; $u_{\rm max}$ is the largest $|u|$ for which $|\tilde{K}(u)| > 0.001$, giving some estimate of the power beyond the Nyquist frequency $u = 0.5$ induced by the interpolant. The maximum of $|E_0(u)|$ and $|\tilde{K}(1 \pm u)|$ attained for $u < x/N$ is listed for three values of x/N, corresponding to 2×, 4×, and 6× zero-padding of the data, respectively.

4. Effect of wrapping

The above section considers the transform from u space back to x space to be continuous, but a discrete transform is necessary in practice. Consider this DFT to produce N_u samples of the x-space image at spacing Δx. The DFT will sample the u space at intervals $\Delta u = (N_u \Delta x)^{-1}$ and produce an output image that has been wrapped with period $P = N_u \Delta x$, such that the output point $\hat{F}_j = \sum_k \hat{F}(j\Delta x + kP)$. The input image $F(x)$ is confined to the extent $\pm N_{\rm in}/2$ of the original input array a_j plus the radius of the non-zero region for the interpolant K_x. Hence as long as the output DFT has extent P that would contain the input image, the wrapped regions are nominally zero.

There is, however, the issue of the ghost images, which appear primarily at the locations $\pm N_x = \pm m N_{\rm in}$, where N_x is the size of the $x \rightarrow u$ input DFT, and m is the zero-padding factor applied to the input data array with $N_{\rm in}$ samples. If $P = N_x/n$ for some integer n, then the ghosts will be folded directly atop the primary image. This might seem damaging, but in fact means that the reconstructed $\hat{F}_j$ will be *better*, in fact *perfect*, because the summed ghost images will exactly cancel the multiplicative error $E_0(x/N_x)$ that affects the primary image. Another way to see this is that if $P = N_x/n$, then the output DFT is using exactly the u values produced by the input DFT, and no interpolation is being done at all.

Wrapping of the ghost images can be a major problem, though, in the following important application: suppose our goal is to take $F(x)$, dilate it to $G(x) = F[x/(1+\epsilon)]$, then convolve with some PSF function. This is, for example, exactly what one wants to do to simulate a sky that has been sheared by weak gravitational lensing—the image must be dilated by $\epsilon = \gamma$ in one direction and contracted with $\epsilon = -\gamma$ along a perpendicular axis. If we wish to execute this dilation in the Fourier domain, setting $\tilde{G}(u) = \tilde{F}[(1+\epsilon)u]$, then the ghosts will appear at locations $x \approx \pm(1-\epsilon)N_x$. If we have done our inverse DFT with $P = N_x$, as might be common, then the ghosts are wrapped to positions $x = \pm\epsilon N_x = \pm \epsilon m N_{\rm in}$, just slightly displaced. Figure 3 illustrates the results.

These displaced ghosts can generate spurious broadening of the reconstructed image $\hat{G}_j$. This is a problem when simulating sheared 2d sky images, because the interpolation errors induce a quadrupole moment atop that induced by the shearing of the image. We take our "bullseye" galaxy and calculate the ellipticity from unweighted quadrupole moments of the image:

$$M_{xx} \equiv \sum_{jk} \hat{G}_{jk} x_j x_k \tag{31}$$

$$M_{yy} \equiv \sum_{jk} \hat{G}_{jk} y_j y_k \tag{32}$$

$$e \equiv \frac{M_{xx} - M_{yy}}{M_{xx} + M_{yy}}. \tag{33}$$

The original bullseye pattern has a diameter of 32 pixels, and is 4× zero-padded to 128 pixels before its DFT. We construct a sheared image of this galaxy by interpolating in u space and executing a DFT back to an x-space image on a 512×512 grid with $\Delta x = 0.25$. The quadrupole moment

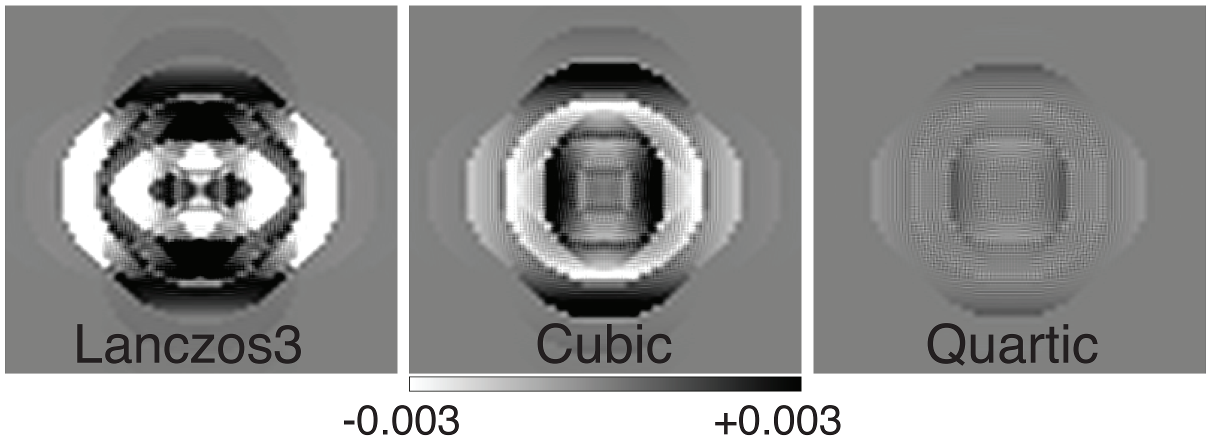

Fig. 3.— Errors induced by Fourier-domain interpolation in the case that we are simulating a shear of amplitude 0.1 on the bullseye image. The output DFT is done with period $P = 128$ such that the ghost images are wrapped atop the central bullseye, and these images show the resultant errors of the central image in closeup. The residuals are dominated by 4 copies of the ghost image, slightly shifted in the $\pm x$ and $\pm y$ directions because of the shear applied before the $u \rightarrow x$ DFT. The greyscale spans ± 0.003 of the input bullseye brightness. The quadrupole patterns from the folded ghosts can confuse weak-lensing shear measurements in this case of unfortunate choice of DFT period, but the quintic filter is a great improvement on the Lanczos and cubic interpolants.

of this reconstruction is compared to that of an image constructed by direct interpolation of the x-space image with a 3rd-order Lanczos kernel. If the applied shear produces an ellipticity e on the x-domain interpolation of the sheared image, we find that the reconstruction from cubic u-space interpolation produces an ellipticity that is systematically biased by about $0.04e$. The quintic filter is better, with biases of $\approx 0.004e$. If we increase the input zero-padding from $4\times$ to $6\times$, we find the quintic filter reduces spurious ellipticities to $< 0.001e$, sufficient for highest-precision simulation of applied shear.

Note that the bullseye image is a worst case in the respect that it has full amplitude out to the edge of the initial domain. In practice we can expect postage-stamp images of galaxies to have near-zero flux at their edges. Since the ghosts of the quintic interpolation rise very rapidly at the edges, a typical galaxy or PSF cutout that has very little flux at the borders will have lower spurious contribution from wrapping of the ghost images.

5. Summary

We seek a recipe for performing high-precision Fourier-domain image simulation and analysis of galaxies and PSFs given as pixelized data, with values between the pixel samples specified by some interpolant K_x.

Our first question was: what is the exact expression for the Fourier transform $\tilde{F}(u)$ of such an interpolated, sampled image? Equation (11) gives the answer: first calculate the $\tilde{a}_k$ from a DFT of the input samples; then interpolate between the $u_k = k/N$ using a wrapped sinc function; then multiply by the transform $\tilde{K}_x(u)$ of the original interpolant. The maximum frequency $u_{\rm max}$ with non-zero $\tilde{F}(u)$ hence depends on the choice of interpolant: $K_x = \mathrm{sinc}(x)$ provides strictly band-limited $u_{\rm max} = 0.5$ but produces an $F(x)$ that extends to very large distance beyond the original samples. The Lanczos interpolants are a good choice to define $F(x)$ that is not much larger than the original samples, while not extending $\tilde{F}(u)$ far beyond the original Nyquist frequency.

The sinc interpolation of the $\tilde{a}_k$ is often too slow for practical implementation, so how could we efficiently obtain the values $\tilde{F}(u)$ at arbitrary u that are needed to implement simulation of sheared and convolved renditions of $F(x)$? One recipe would be to interpolate $F(x)$ to new grid such that a DFT will produce a sample at the desired u value(s), *i.e.* x interpolation followed by $x \rightarrow u$ DFT. If, however, we can reverse the order— $x \rightarrow u$ DFT followed by interpolation with a kernel K_u in u space. This would have the advantage that one DFT could be used to produce many simulated renditions of the object.

We have seen that sinc interpolation in u space would be exact, but impractical. Approximation of the sinc with a more compact kernel produces two errors in the simulated images: the first is a multiplicative error $E_0(x/N)$, where N is the size of the $x \rightarrow u$ DFT and E_0 is a function characteristic of the interpolant K_u. The second error is the appearance of a pair of ghost images located $\pm N$ units from the original image, each ghost mutiplied by the function $\tilde{K}_u(1 \pm x/N)$, which is nearly the same size as $E_0(x/N)$.

Figures 1 and Table 1 show the size of these errors for common interpolants. To attain sub-percent errors on the simulated images, we find this recipe effective:

1. Zero-pad the input data by a factor 4 before performing the $x \rightarrow u$ DFT.
2. Use a 4×4 pixel cubic interpolant in u space.

The cubic interpolant out-performs the Lanczos filters once the data are zero-padded, which inspires us to construct a 6×6 pixel 5-th order polynomial interpolant expressly designed to minimize the interpolation errors for $|x/N| < \frac{1}{4}$. With this quintic interpolant in place of the cubic K_u, the simulation errors are kept to amplitude < 0.001.

We therefore recommend use of the quintic filter for u-space interpolation after zero-padding and DFT. The cubic filter can be used to gain a factor $2 - -3$ in speed at expense of $\approx 5\times$ larger simulation errors.

There is one important caveat to this recipe: if the simulated u-domain image is transformed back to x domain using a DFT with period P that nearly evenly divides N, then the ghosts will be folded atop the primary image. A simulated shear or magnification will cause the ghosts to move relative to the primary image, which produces spurious broadening of the image, which can change the quadrupole moments of the image in a way that biases shear measurements. In a simple pessimistic trial case, we find that the standard $4\times$padding+cubic recipe induces 4% errors in a simple measure of applied shear, much too big for testing current-generation shear measurement techniques. Using the quintic interpolant reduces this error to 0.4%, good enough for current-generation tests. We find it possible to reduce the spurious shear to < 0.001 of the applied shear by combining the quintic K_u with $6\times$ zero-padding of the initial DFT, which should be sufficiently accurate for foreseeable cosmic-shear simulations. In fact this would be overkill for most galaxy images that one is likely to be simulating, since they are already nearly zero at their edges and hence not in need of $6\times$ padding.

A. Function and interpolant definitions

We adopt the following conventions for functions and Fourier transforms:

$$\Pi(x) \equiv \begin{cases} 1. & |x| < 0.5 \\ 0.5 & |x| = 0.5 \\ 0 & |x| > 0.5 \end{cases} \tag{A1}$$

$$\mathrm{sinc}(x) \equiv \frac{\sin \pi x}{\pi x} \tag{A2}$$

$$\mathrm{Si(x)} \equiv \int_0^x dt \frac{\sin t}{t} \tag{A3}$$

$$\tilde{f}(u) = \int_{-\infty}^{\infty} dx\, f(x) e^{-2\pi i u x} \tag{A4}$$

$$f(x) = \int_{-\infty}^{\infty} du\, \tilde{f}(u) e^{2\pi i u x} \tag{A5}$$

The interpolants considered in this paper are defined as follows:

- **Nearest-neighbor:** The real-space kernel is maximally compact, $K(x) = \Pi(x)$, but the Fourier domain $\tilde{K}(u) = \mathrm{sinc}(u)$ extends to infinity as $\sim (1/u)$.

- **Linear:** Common interpolant with 2-point footprint in real space, and improved but still very broad Fourier behavior $\sim (1/u)^2$:

$$K(x) = \begin{cases} 1. - |x| & |x| \le 1 \\ 0 & |x| \ge 1 \end{cases} \tag{A6}$$

$$\tilde{K}(u) = \mathrm{sinc}^2(u). \tag{A7}$$

- **Cubic:** Piecewise-cubic polynomial interpolant with continuous first derivates designed to interpolate perfectly up to second order in a Taylor expansion. This implies $\tilde{K}'(j) = \tilde{K}''(j) = 0$ for integers $j \neq 0$. The Fourier domain expression is analytic but too complex to merit detailing:

$$K(x) = \begin{cases} \frac{3}{2}|x^3| - \frac{5}{2}x^2 + 1 & |x| \le 1 \\ -\frac{1}{2}|x^3| + \frac{5}{2}x^2 - 4|x| + 2 & 1 \le |x| \ge 2 \\ 0 & |x| \ge 2 \end{cases} \tag{A8}$$

- **Quintic:** A six-point interpolant that is exact to fourth order in a Taylor expansion, and hence has $\tilde{K}(j \pm \nu) \sim \nu^5$ for $j \neq 0$, can be produced from a piecewise-quintic polynomial kernel. This kernel also has continuous second derivatives:

$$K(x) = \begin{cases} 1 + \frac{x^3}{12}\left(-95 + 138x - 55x^2\right) & |x| \le 1 \\ \frac{(x-1)(x-2)}{24}\left(-138 + 348x - 249x^2 + 55x^3\right) & 1 \le |x| \le 2 \\ \frac{(x-2)(x-3)^2}{24}\left(-54 + 50x - 11x^2\right) & 2 \le |x| \le 3 \\ 0 & |x| \ge 3 \end{cases} \tag{A9}$$

- **Lanczos:** at order m truncates the sinc filter after its mth null:

$$K(x) \equiv \mathrm{sinc}(x)\mathrm{sinc}(x/m)\Pi(x/2m) \tag{A10}$$

$$\tilde{K}(u) \equiv 2m^2\Pi(u) * \Pi(u/m) * \mathrm{sinc}(2mu) \tag{A11}$$

$$= \pi(2mu - m - 1)\mathrm{Si}(2mu - m - 1) - \pi(2mu - m + 1)\mathrm{Si}(2mu - m + 1) \tag{A12}$$

$$-\pi(2mu + m - 1)\mathrm{Si}(2mu + m - 1) + \pi(2mu + m + 1)\mathrm{Si}(2mu + m + 1) \tag{A13}$$

Abramowitz & Stegun (1965) present approximations to the Si function which are very useful for calculating $\tilde{K}$ for the Lanczos interpolants or for estimating leading-order behavior of the interpolation error quantities described in this text. See their (5.2.6), (5.2.34), and (5.2.38).

As noted in §3.1, the Lanczos filters do not conserve background flux, but in practice can be modified to do so. The plots and table assume that we are using background-conserving versions of the Lanczos filters.

- **Sinc:** conjugate of the nearest-neighbor filter, with $K(x) = \mathrm{sinc}(x)$ and $\tilde{K}(u) = \Pi(u)$.

REFERENCES

Abramowitz, M., & Stegun, I. 1965, Handbook of Mathematical Functions, (New York: Dover)

Amara, A., & Réfrégier, A. 2008, MNRAS, 391, 228

Bernstein, G. M. 2010, MNRAS, 406, 2793

Bertin, E. 2011, Astronomical Data Analysis Software and Systems XX, 442, 435

Huterer, D., Takada, M., Bernstein, G., & Jain, B. 2006, MNRAS, 366 101

Mandelbaum, R., Hirata, C. M., Leauthaud, A., Massey, R. J., & Rhodes, J. 2011, MNRAS, 2107

This preprint was prepared with the AAS LaTeX macros v5.2.