A Linear System-Free Gaussian RBF Method for the Gross-Pitaevskii equation on unbounded domains

Scott A. Sarra
Marshall University

April 28, 2010

Abstract

Gaussian Radial Basis Function (RBF) interpolation methods are theoretically spectrally accurate. However, in applications this accuracy is seldom realized due to the necessity of solving a very poorly conditioned linear system in order to evaluate the methods. Recently, by using approximate cardinal functions and restricting the method to a uniformly spaced grid (or a smooth mapping thereof), it has been shown that the Gaussian RBF method can be formulated in a matrix free framework that does not involve solving a linear system [1]. In this work we differentiate the linear system-free Gaussian (LSFG) method and use it to solve Partial Differential Equations on unbounded domains that have solutions that decay rapidly and that are negligible at the ends of the grid. As an application, we use the LSFG collocation method to numerically simulate Bose-Einstein Condensates.

Keywords: RBF interpolation, RBF collocation for PDEs, Numerical PDEs, Bose-Einstein Condensates

1 Introduction

Over the last 25 years, RBF methods have become an important tool for the interpolation of scattered data and for solving Partial Differential Equations

[2]. RBF methods that employ infinitely differentiable basis functions that contain a free parameter are theoretically spectrally accurate. The implementation of RBF methods involves solving a linear system that is extremely ill-conditioned when the parameters of the method are such that the best accuracy is theoretically realized. Thus, in applications, RBF methods are not able to produce as accurate of results as they are theoretically capable of.

For appropriately chosen interpolation sites in 1d, the non-polynomial RBF methods are known to be equivalent to polynomial methods in the limit as the shape parameter goes to zero [3]. RBF methods with small values of the shape parameter have been evaluated with "bypass" algorithms [4] that evaluate the method without solving the associated ill-conditioned linear systems. The bypass algorithms are applicable only for use with a small number of interpolation sites and thus are not well suited for applications. However, the bypass algorithms have been used to show that RBF methods are often more accurate than polynomial based methods when a small, non-zero value of the shape parameter is used.

Unlike polynomial based methods, with RBF methods it is not possible to rearrange the basis functions into an equivalent cardinal basis which reduces the interpolation matrix in the new basis to the identity matrix. It has been recently shown [1] that if the Gaussian RBF interpolation method is restricted to a uniform grid, that an approximate cardinal basis can be used to efficiently implement the method without any loss of accuracy. With the approximate cardinal approach, the Gaussian RBF method can be accurately implemented with very small values of the shape parameter where it is most accurate. In this work we use the approximate cardinal approach to approximate derivatives and to numerically solve nonlinear time-dependent PDEs. As a particular application, we use the linear system-free Gaussian (LSFG) collocation method to numerically simulate the motion of Bose-Einstein Condensates.

We note that there also exists a linear system-free Gaussian method for use with equally spaced centers on bounded domains [5]. The method in [5] is based on the connection of the RBF method to polynomial methods and on potential theory rather than on the approximate cardinal function approach of [1].

2 Gaussian RBF interpolation

The RBF interpolation method uses linear combinations of translates of one function $\phi(r)$ of a single real variable. Given a set of ***centers*** $\mathbf{x}_1^c, \ldots, \mathbf{x}_N^c$ in $\mathbb{R}^d$, the RBF interpolant takes the form

$$s(\mathbf{x}) = \sum_{j=1}^{N} \lambda_j \phi(\left\| \mathbf{x} - \mathbf{x}_j^c \right\|_2). \tag{1}$$

Many different basis functions $\phi(r)$ have been used, but we concentrate on the Gaussian RBF

$$\phi(r) = e^{-\epsilon^2 r^2} \tag{2}$$

where $\varepsilon > 0$ is a free parameter called the ***shape parameter***. The coefficients, λ, are chosen by enforcing the interpolation condition

$$s(\mathbf{x}_i) = f(\mathbf{x}_i) \tag{3}$$

at a set of nodes that typically coincide with the centers. Enforcing the interpolation condition at N centers results in a $N \times N$ linear system

$$B\lambda = f \tag{4}$$

to be solved for the RBF expansion coefficients λ. The matrix B with entries

$$b_{ij} = \phi(\left\| \mathbf{x}_i^c - \mathbf{x}_j^c \right\|_2), \qquad i, j = 1, \ldots, N \tag{5}$$

is called the ***interpolation matrix*** or the ***system matrix***. For distinct center locations, the system matrix for the GA RBF is known to be nonsingular [6] if a constant shape parameter is used. To evaluate the interpolant at M points $\mathbf{x}_i$ using (1), the $M \times N$ ***evaluation matrix*** H is formed with entries

$$h_{ij} = \phi(\left\| \mathbf{x}_i - \mathbf{x}_j^c \right\|_2), \qquad i = 1, \ldots, M \text{ and } j = 1, \ldots, N. \tag{6}$$

Then the interpolant is evaluated at the M points by the matrix multiplication

$$f_a = H\lambda. \tag{7}$$

Theoretically, RBF methods are most accurate when the shape parameter is small. However, the use of small shape parameters results in system matrices that are very poorly conditioned. The by now very established fact that in RBF methods we cannot have both good accuracy and good conditioning at the same is known as the ***uncertainty principle*** [7]. Recent books [8, 9, 2, 10] on RBF methods can be consulted for more information.

3 Linear system-free Gaussian (LSFG) RBF interpolation

The LSFG interpolation method [1] is restricted to application on a 1d infinite grid with uniform spacing h. This simplification is so that all cardinal functions are simply translations of a master basis function, such as is the case for the sinc basis [11, 12]. The Gaussian RBF (2) is reformulated to incorporate the grid spacing as

$$\phi(r) = e^{-[\alpha^2/h^2]r^2}. \tag{8}$$

The shape parameter ε in the direct method and the parameter α are related by the equation

$$\alpha = h\varepsilon. \tag{9}$$

The motivation behind deriving approximate cardinal functions for the Gaussian RBFs is that RBF methods exhibit ***error saturation*** [9]. In particular, for a method employing a basis function of the form (8), the error in approximating a given function converges to a nonzero function

$$E_s(\alpha) \approx 4\exp(-\pi^2/\alpha^2)$$

as $h \to 0$ for fixed α. The error saturation is smaller than machine epsilon ($\epsilon_m = 2.2\times 10^{-16}$) for $\alpha < 1/2$. Due to the error saturation, an exact cardinal function is unnecessary. All that is needed is an approximate cardinal function that approximates the exact cardinal function with an error less than E_s. In [1], the approximate cardinal function

$$C(X) \approx \frac{\alpha^2}{\pi}\frac{\sin(\pi X)}{\sinh(\alpha^2 X)} \tag{10}$$

is derived that approximates the exact cardinal function with error

$$E_c \approx 4\exp(-2\pi^2/\alpha^2)$$

which is the square of the saturation error and is less than machine epsilon for $\alpha < 0.726$. The approximate cardinal function (10) reduces to the sinc function

$$\text{sinc}(X) = \frac{\sin(\pi X)}{\pi X}$$

as $\alpha \to 0$ [1]. Because the error in the approximation of the master cardinal function is the square of the error saturation, there is no penalty for using the approximations to obtain linear system-free interpolating RBF approximations. Thus, the linear system-free Gaussian RBF interpolant is

$$f(x) \approx \frac{\alpha^2}{\pi} \sum_{j=1}^{N} f\left(x_j^c\right) \frac{\sin\left(\pi[x - x_j^c]/h\right)}{\sinh\left(\alpha^2[x - x_j^c]/h\right)} \qquad \alpha < 0.726, \tag{11}$$

where the N centers x^c have uniform spacing h.

To apply the LSFG method on a finite interval, the infinite domain is truncated to a large finite grid that is chosen sufficiently wide so that the function being approximated has negligible amplitude at the ends of the grid. The function being approximated must be sufficiently small near boundaries since the cardinal functions near the endpoints of a finite grid, for which no explicit approximation is known, are different from those for an unbounded grid. If the function being approximated is sufficiently small near the boundary, the cardinal functions near the boundary have negligible coefficients, and the function can be approximated entirely and accurately by the interior approximate cardinal functions. The LSFG method can be applied on a nonuniform grid by using a mapping of the uniform grid to a nonuniform grid as is commonly done with pseudospectral methods [11]. The LSFG method is applied on tensor product grids in higher dimension.

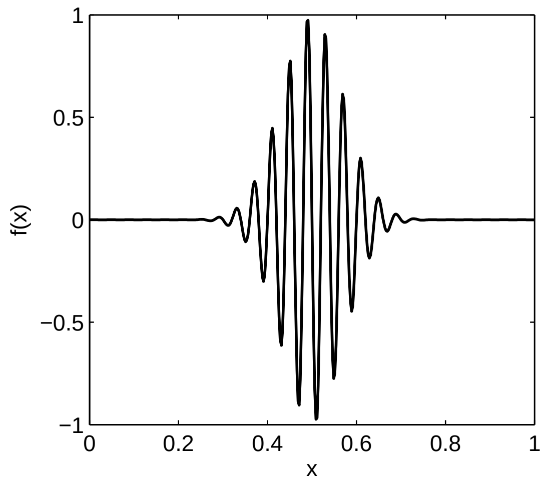

Figure 1: The graph of function (13)

4 Derivative approximation

Formulas for the elements of the LSFG differentiation matrices, D, are found by differentiating the interpolant (11) and then evaluating the result at the N centers. Derivatives of a function $f(x)$ at the centers are approximated by the matrix-vector product $Df(x)$.

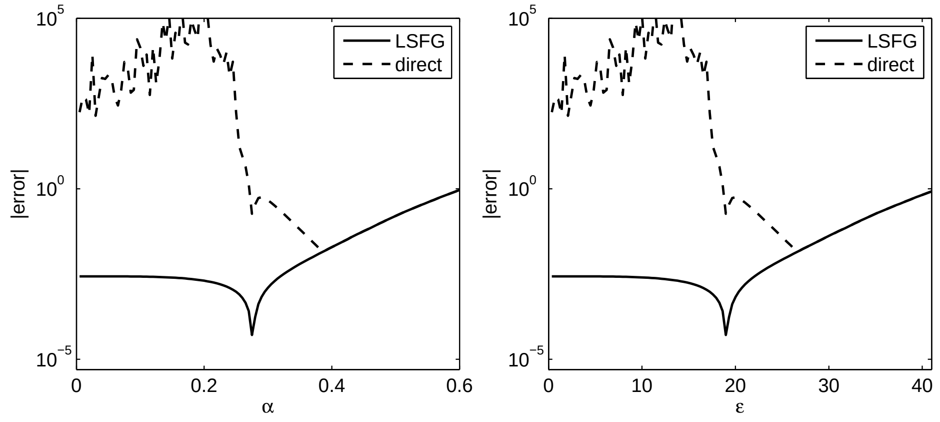

Figure 2: $N = 70$, maximum error in calculating derivatives of function (13) by the direct method and the LSFG method. Left: error versus the α parameter. Right: error versus the equivalent shape parameters.

Let

$$r_{ij} = x_i^c - x_j^c = h(i-j) \tag{12}$$

be the signed distance between centers, $\mu_{ij} = (\alpha^2 r_{ij})/h$, and $\rho_{ij} = (\pi r_{ij})/h$. The first-order LSFG differentiation matrix D_1 has elements

$$d_{ij} = \frac{\alpha^2}{\pi}\text{csch}(\mu_{ij})\left[\frac{\pi\cos(\rho_{ij}) - \alpha^2\text{coth}(\mu_{ij})\sin(\rho_{ij})}{h}\right]$$

for $i \neq j$ and $d_{ij} = 0$ for $i = j$. The LSFG second-order differentiation matrix D_2 has elements for $i \neq j$

$$d_{ij} = \frac{\alpha^2}{\pi}\text{csch}(\mu_{ij})\left[\frac{-2\alpha^2\pi\cos(\rho_{ij})\text{coth}(\mu_{ij}) + (\alpha^4 - \pi^2 + 2\alpha^4\text{csch}^2(\mu_{ij}))\sin(\rho_{ij})}{h^2}\right]$$

and for $i = j$

$$d_{ij} = -\frac{\alpha^4 + \pi^2}{3h^2}.$$

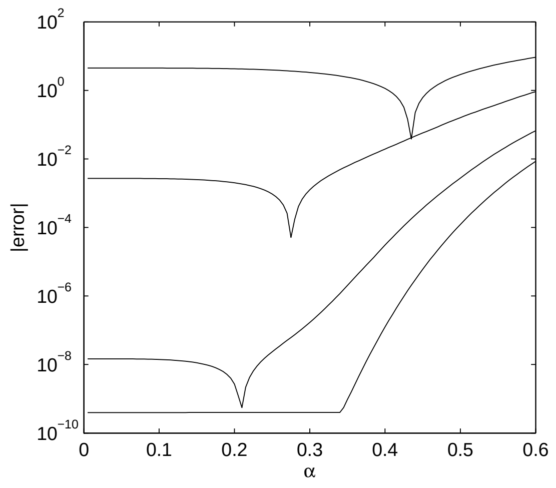

Figure 3: Convergence trend in calculating the first derivative of function (13) with $N = 60, 70, 80, 90$ respectively from top to bottom.

Due to the structure of the LSFG differentiation matrices, it is not necessary to form the entire differentiation matrix, nor is a $\mathcal{O}\left(N^2\right)$ operation count matrix by vector product necessary to evaluate a derivative. LSFG differentiation matrices have a Toeplitz structure (constant along the diagonals). Additionally, odd order LSFG differentiation matrices are skew-symmetric and even order LSFG differentiation matrices are symmetric. Thus, only the first row or column of the differentiation matrix needs to be formed and then the Fast Fourier Transform (FFT) algorithm can be efficiently used to evaluate a derivative approximation in $\mathcal{O}\left(N \log N\right)$ floating point operations. Reference [13] can be consulted for details. A Matlab function that uses the FFT to evaluate the first and second derivatives of a function $f(x)$ is in listing 1. The Toeplitz structure and symmetry properties of the LSFG differentiation matrices are enjoyed by the sinc method differentiation matrices as well.

The function

$$f(x) = \sin(50\pi x)e^{-100(x-1/2)^2}, \qquad x \in [0, 1] \tag{13}$$

is an example of a function for which the LSFG method is applicable as it has a negligible size of $\mathcal{O}\left(10^{-10}\right)$ near the boundaries. The graph of function (13) is shown in figure 1. Figure 2 compares the accuracy in calculating the first derivative of function (13) by the LSFG method and the direct

Listing 1: LsfgDiffFFT.m

```
function Df = mfg_diff_fft(f, a, xc, d)

% Inputs:
%    f     function values on a evenly spaced grid
%    a     the RBF alpha parameter in definition (8)
%   xc     N evenly spaced centers
%    d     derviative order, d = 1 or 2
% Output:
%   Df     LSFG derivative approximation of f at the centers xc

     f = f(:).';     xc = xc(:).';    % Ensure f and xc are row vectors
     N = length(f);
     h = abs(xc(2)-xc(1));
     r = xc(1) - xc;
                         % Compute first row of the differention matrix
if d==1
    row1 = ((a^2)/(h*pi))*csch(((a^2)*r)/h).*( pi*cos((pi*r)/h) - ...
           (a^2)*coth(((a^2)*r)/h).*sin((pi*r)/h) );
    row1(find(isnan(row1))) = 0;
elseif d==2
    row1 = ((a^2)/(pi*h^2))*csch(((a^2)*r)/h).* ...
           ( -2*(a^2)*pi*cos((pi*r)/h).*coth(((a^2)*r)/h) + ...
         (a^4 - pi^2 + 2*(a^4)*csch(((a^2)*r)/h).^2).*sin((pi*r)/h)   );
    row1(find(isnan(row1))) = -(a^4 + pi^2)/(3*h^2);
end

% Imbed first row of the Toeplitz matrix into a bigger circulant matrix:

 row = [row1 zeros(1,2^nextpow2(2*N)-2*N+1) ((-1)^d)*fliplr(row1(2:N))];

% Multiply circulant matrix times vector using the FFT:

     M = length(row);
    ew = M*ifft(row);                % eigenvalues of circulant matrix
  fhat = fft([f zeros(1,M-N)]);      % FFT of a padded data vector
    Df = ifft(ew.*fhat);             % multiply the result by ew's
    Df = Df(1:N)';                   % take inverse FFT

if max(abs(imag(f))) == 0;           % Real data in, real derivative out
   Df = real(Df);
end
```

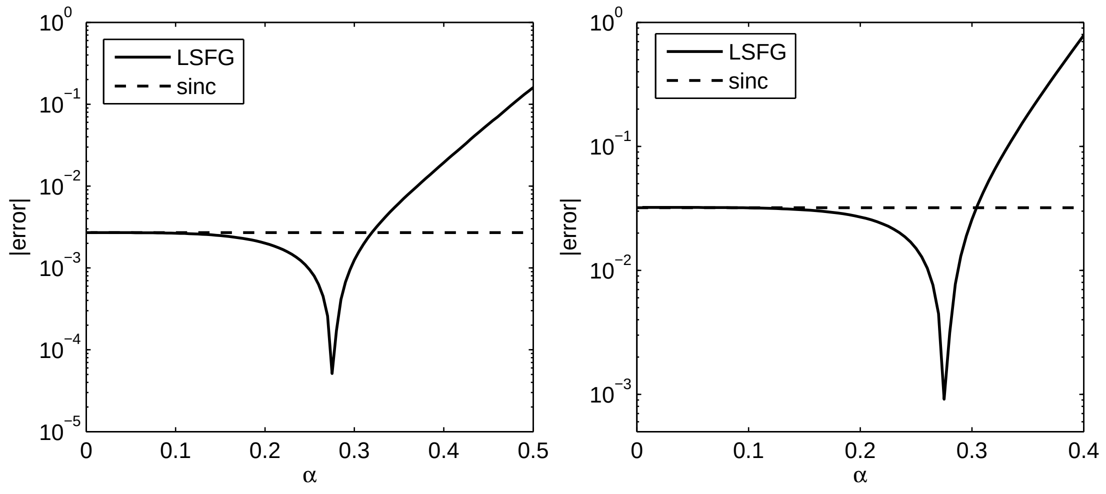

Figure 4: $N = 70$, maximum error over a range of α parameter in calculating derivatives of function (13). Left: first derivative. Right: second derivative.

Gaussian RBF method that solves linear system (4). The direct method has a minimum error of 1.5e-2 at $\alpha = 0.385$ ($\varepsilon = 26.565$) while the LSFG method has a minimum error of 5.1e-5 at $\alpha = 0.275$ ($\varepsilon = 18.975$). The direct method is unable to accurately evaluate the solution for $\varepsilon < 26.565$ due to the system matrix of the method being very ill-conditioned. The convergence trend in approximating the first derivative of function (13) as N is increased is illustrated in figure 3. The error is measured in the infinity norm.

Next we compare the accuracy, in approximating the first and second derivative of function (13), of the LSFG method and the sinc method, to which the LSFG method is equivalent to as $\alpha \to 0$. Matlab code for implementing the sinc method is found in [14]. The results are illustrated in figure 4. The first derivative results are in the left image of the figure where the smallest LSFG error is 5.1e-5 with $\alpha = 0.275$ and the sinc error is 2.7e-3. The second derivative results are shown in the right image of the figure. The smallest LSFG method error is 9.1e-4 with $\alpha = 0.275$ and the sinc error is 3.2e-2. RBF methods are often more accurate, with some small shape parameter greater than zero, then are their limiting method as the shape parameter goes to zero.

5 Time-dependent PDEs

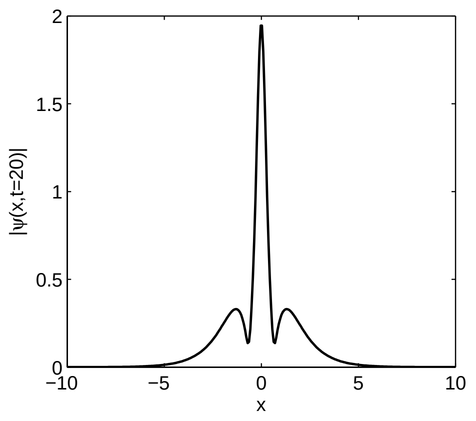

Figure 5: The solution of equation (14) at time $t = 20$.

After the space derivatives of a time-dependent PDE are discretized with the LSFG method, a method of lines approach is taken, and the remaining system of ordinary differential equations

$$\psi_t = F(\psi, t)$$

is advanced in time with an ODE method. In the numerical examples we have used a fourth-order Runge-Kutta method.

The dimensionless Gross–Pitaevskii (also called a nonlinear Schrödinger equation) equation is

$$i\psi_t + \tau\nabla^2\psi - V(x)\psi - \beta \,|\, \psi|^2 \,\psi = 0 \tag{14}$$

where $V(x)$ is an external trap potential and the quantity $|\,\psi(x)|^2$ represents a particle density. The Gross–Pitaevskii equation can be used to model Bose–Einstein condensates (BEC) which are a state of matter of a dilute gas of weakly interacting bosons confined in an external potential and cooled to temperatures very near to absolute zero. Under such conditions, a large fraction of the bosons collapse into the lowest quantum state of the external potential, and all wave functions overlap each other, at which point quantum effects become apparent on a macroscopic scale. The state of matter was first predicted by Satyendra Nath Bose and Albert Einstein in 1924. The

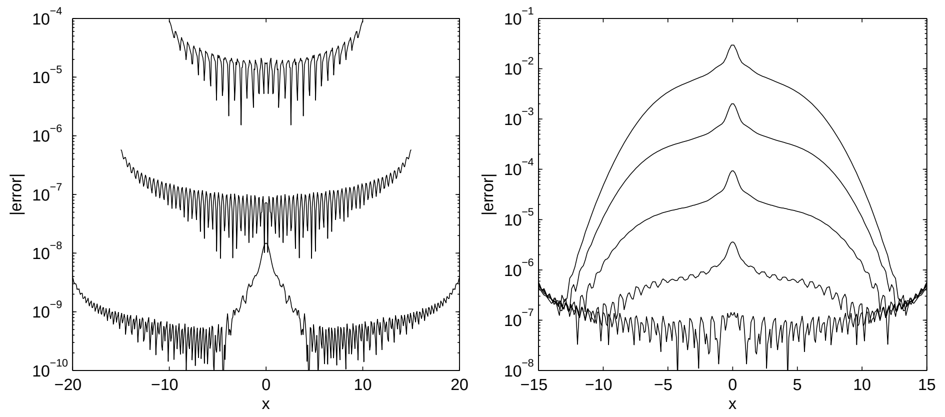

Figure 6: Left: Pointwise errors of 1d Gross–Pitaevskii problem at $t = 1$ on intervals of three different sizes with $h = 1/15$ and $\varepsilon = 1.25$ ($\alpha = 0.0835$). Right: Convergence as N is increased and the shape parameter is held fixed at $\varepsilon = 1.25$.

phenomenon was observed for the first time in 1995, and is now the subject of intense theoretical and experimental study [15].

As an accuracy check we start with a 1d example that has an exact solution. We take equation (14) with $V(x) = 0$, $\tau = 1$, and $\beta = 8$. The initial condition is $\psi(x, 0) = sech(x)$. The problem is defined on the real line, but numerically the domain is truncated to a finite interval $\Omega = [-A, A]$ that is large enough so that the rapidly decaying solution does not reach the boundary. The choice of A is discussed later. The exact solution of the problem is $\psi(x, t) = a + bi$ where

$$a = \frac{\cosh(x)\left[4\cos(t)\cosh^2(x) + 3\cos(t)\cos(8t) - 3\cos(t) - 3\sin(t)\sin(8t)\right]}{4\cosh^4(x) - 3 + 3\cos^2(4t)}$$

and

$$b = \frac{\cosh(x)\left[3\cos(t)\sin(8t) + 4\sin(t)\cosh^2(x) + 3\sin(t)\cos(8t) - 3\sin(t)\right]}{4\cosh^4(x) - 3 + 3\cos^2(4t)}.$$

The solution is called a breather solution which is a localized periodic solution with a soliton structure. The solution at time $t = 20$ is shown in figure 5.

The next numerical experiment is to gain insight about the effect of the size of the computational interval. The problem is discretized on intervals of

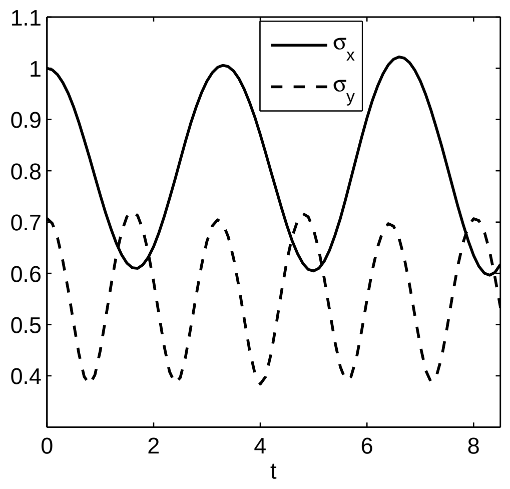

Figure 7: 2d Gross–Pitaevskii equation (14) solution. BEC widths versus time with $N = 80$, $\Delta t = 0.0025$ and $\alpha = 0.1$.

three different sizes with $A = 10$, $A = 15$, and $A = 20$. The shape parameter is taken to be $\varepsilon = 1.25$ and with each interval N is chosen so that $h = 1/15$. With $A = 10$, the magnitude of the initial condition at the interval endpoints is 9.1e-5, with $A = 15$ it is 6.1e-7, and for $A = 20$ it is 4.1e-9. In figure 6 it can be seen that the error in boundary regions is proportional to the size of the PDE solution at the boundary in this example.

Next, we take $A = 15$ and fix the shape parameter at $\varepsilon = 1.25$ and examine the convergence of the method as N is increased. As N is increased α is adjusted according to equation (9) so that ε remains fixed. The plot of the pointwise error as N takes the values $N = 150, 200, 250, 300, 250$, is shown in the right image of figure 6. For each N, the errors are virtually identical near the boundary and the errors decay on the interior as N is increased.

5.1 2d example

The next two examples, taken from reference [16], are examples of the Gross–Pitaevskii equation modeling BECs. In [16], a split-step Fourier method was used. To quantify the numerical results we calculate the condensate widths along the x, y, and z axes as defined by

$$\sigma_\eta^2 = \int_{\mathcal{R}_d} \eta^2 \, |\, \psi(\mathbf{x}, t)|^2 \, d\mathbf{x} \qquad \eta = x, y, z. \tag{15}$$

The trapezoid rule has been used to evaluate the integrals.

The 2d example takes equation (14) with

$$V(x, y) = (x^2 + 2y^2)/2,$$

$\tau = 1/2$, and $\beta = 2$. The initial condition is

$$\psi(x, y, 0) = \frac{2^{1/4}}{\sqrt{2\pi}} e^{-[(x^2+2y^2)/4]}.$$

The domain is $\Omega = [-8, 8]^2$. The condensate widths are shown in figure 7. The example is the same as example 2, case IV, from reference [16]. An analytical solution to the problem is not known. The LSFG method solution with $N = 80$, $\Delta t = 0.0025$, and $\alpha = 0.1$ is visually identical to the split-step Fourier solution with $N = 512$ and $\Delta t = 0.001$ that is given in [16]. Contour plots for the 2d BEC example are shown in figure 8. For the contour plots, only the portion of the domain ($[-3, 3]^2$) where the BEC is trapped is shown. Despite not knowing the contour levels that were used in [16], the contour plots from the LSFG solution are qualitatively similar to the plot given in reference [16].

5.2 3d example

The 3d example sets up equation (14) with

$$V(x, y, z) = (x^2 + 4y^2 + 16z^2)/2,$$

$\tau = 1/2$, and $\beta = 1/10$. The initial condition is

$$\psi(x, y, z, 0) = \frac{8^{1/4}}{\sqrt{(\pi/4)^{3/4}}} e^{-2(x^2+2y^2+4z^2)}$$

and the domain is $\Omega = [-8, 8]^3$. The setup is the same as case I of example 4 in reference [16]. The condensate widths for $N = 120$ are shown in figure 9. In moving from $N = 80$, to $N = 100$, and to $N = 120$, it is apparent that the solution has become independent of the grid. The results are qualitatively the same as the results in the left image of figure 10 in reference [16].

6 Conclusions

Methods for PDE problems on infinite domains fall into three broad categories [11, Chapter 17]: domain truncation, basis functions suited to an infinite interval such as sinc or Hermite functions, or mapping the unbounded interval to a bounded interval and then using a bounded interval approach such as the Chebyshev or Fourier pseudospectral method. Most, but not all methods, require the solution of the problem to decay sufficiently fast for large $|\mathbf{x}|$.

Of the methods for infinite interval problems, the LSFG method is most closely related to the sinc method to which it is equivalent in the limit $\alpha \to 0$. The LSFG method shares the desirable properties of the sinc method, such as being able to efficiently approximate derivatives using the FFT in $\mathcal{O}(N \log N)$ flops. The LSFG method may be viewed as a sinc method with a tunable parameter that often affords greater accuracy than the standard sinc method. In many cases, such as in the example in section 4, the LSFG method is more accurate with some small $\alpha > 0$, than is the sinc method.

In comparison with the Gaussian RBF method from which the LSFG method was derived, the LSFG method is less flexible in that it can only be applied on uniformly spaced grids in 1d and on tensor product grids of the 1d grid in higher dimensions. However, the LSFG method may be mapped to non-uniform grids if desired. The LSFG method eliminates the $\mathcal{O}(N^3)$ operation count setup phase of solving a linear system that is required by the direct method. The LSFG method accurately handles small shape parameters, for which the RBF method is most accurate, that are unreachable by the direct RBF method. The direct RBF method evaluates a derivative in a $\mathcal{O}(N^2)$ matrix multiplication while the LSFG method needs only $\mathcal{O}(N \log N)$ flops.

References

[1] J. P. Boyd and L. Wang. An analytic approximation to the cardinal functions of gaussian radial basis functions on an infinite lattice. *Applied Mathematics and Computation*, 215(6):2215–2223, 2009. (document), 1, 3, 3, 3

[2] S. A. Sarra and E. J. Kansa. *Multiquadric Radial Basis Function Approximation Methods for the Numerical Solution of Partial Differential Equations*. Tech Science Press, 2010. 1, 2

[3] R. Schaback. Limit problems for interpolation by analytic radial basis functions. *Journal of Computational and Applied Mathematics*, 212:127–149, 2008. 1

[4] B. Fornberg and G. Wright. Stable computation of multiquadric interpolants for all values of the shape parameter. *Computers and Mathematics with applications*, 48:853–867, 2004. 1

[5] R. Platte and T. Driscoll. Polynomials and potential theory for gaussian radial basis function interpolation. *SIAM Journal on Numerical Analysis*, 43(2):750–766, 2005. 1

[6] C. Micchelli. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constructive Approximation*, 2:11–22, 1986. 2

[7] R. Schaback. Error estimates and condition numbers for radial basis function interpolation. *Advances in Computational Mathematics*, 3:251–264, 1995. 2

[8] M. D. Buhmann. *Radial Basis Functions*. Cambridge University Press, 2003. 2

[9] G. E. Fasshauer. *Meshfree Approximation Methods with Matlab*. World Scientific, 2007. 2, 3

[10] H. Wendland. *Scattered Data Approximation*. Cambridge University Press, 2005. 2

[11] J. Boyd. *Chebyshev and Fourier Spectral Methods*. Dover, second edition, 2000. 3, 3, 6

[12] F. Stenger. Summary of sinc numerical methods. *Journal of Computational and Applied Mathematics*, 121:379420, 2000. 3

[13] G. Strang. A proposal for Toeplitz matrix calculations. *Studies in Applied Mathematics*, 74:171–176, 1986. 4

[14] J. Weideman and S. Reddy. A MATLAB differentiation matrix suite. *ACM Transactions on Mathematical Software*, 26:465–519, 2000. 4

[15] C. J. Pethick and H. Smith. *Bose-Einstein Condensation in Dilute Gases*. Cambridge University Press, second edition, 2008. 5

[16] W. Bao, D. Jaksch, and P. A. Markowich. Numerical solution of the Gross-Pitaevskii equation for Bose-Einstein condensation. *Journal of Computational Physics*, 187:318–342, 2003. 5.1, 5.1, 5.2

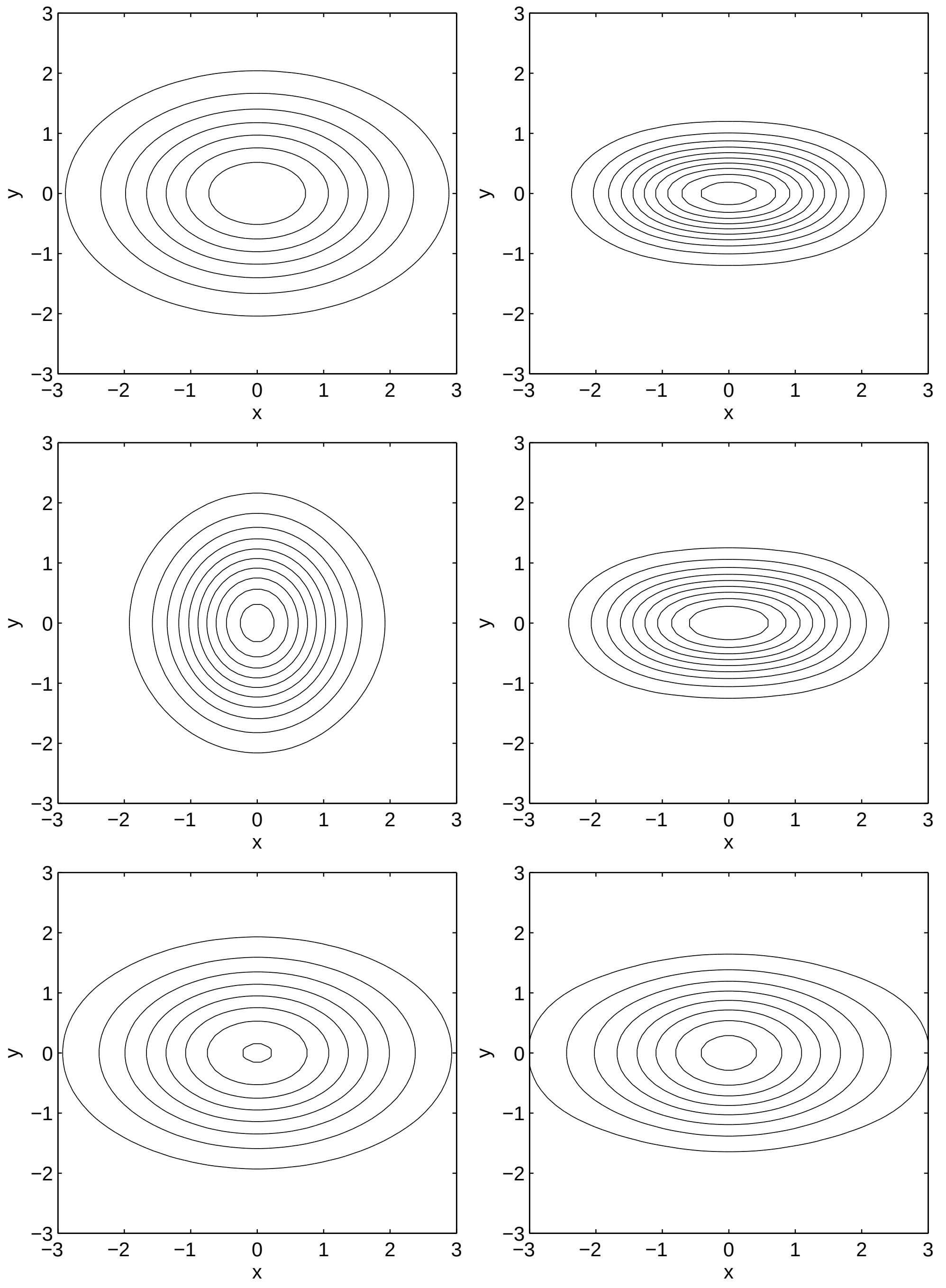

Figure 8: 2d Gross–Pitaevskii equation (14), contour plots of $|\psi(x,y,t)|^2$ at times: $t = 0$, $t = 0.85$, $t = 1.7$, $t = 2.55$, $t = 3.4$, and $t = 6.8$.

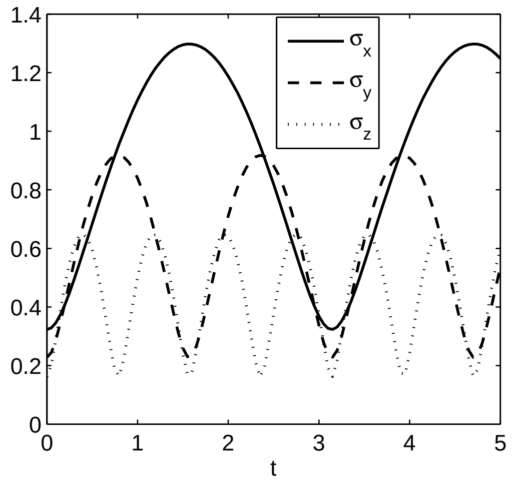

Figure 9: 3d Gross–Pitaevskii equation (14), BEC widths versus time with $N = 120$, $\Delta t = 0.001$, and $\alpha = 0.1$.