ECCOMAS Congress 2016
VII European Congress on Computational Methods in Applied Sciences and Engineering
M. Papadrakakis, V. Papadopoulos, G. Stefanou, V. Plevris (eds.)
Crete Island, Greece, 5–10 June 2016

A DIRECT SOLVER FOR THE ADVECTION-DIFFUSION EQUATION USING GREEN'S FUNCTIONS AND LOW-RANK APPROXIMATION

Jonathan R. Bull[1], **Sverker Holmgren**[1] **and Stefan Engblom**[1]

[1]Division of Scientific Computing
Department of Information Technology, Uppsala University
Polacksbacken, Uppsala, Sweden
e-mail: {jonathan.bull,sverker.holmgren,stefan.engblom}@it.uu.se

Keywords: advection diffusion, Green's function, semi-implicit, low-rank approximation

Abstract. *A new direct solution method for the advection-diffusion equation is presented. By employing a semi-implicit time discretisation, the equation is rewritten as a heat equation with source terms. The solution is obtained by discretely approximating the integral convolution of the associated Green's function with advective source terms. The heat equation has an exponentially decaying Green's function, allowing for a reduction of effort via low-rank matrix approximation. Simple low-rank approximations of the Green's function matrix are investigated as a precursor to using the Fast Multipole Method in higher dimensions. Results show that fast, stable and accurate computations can be achieved by this method. Low-rank approximation saves computational time at the expense of some accuracy. The new method is a template for developing fast, scalable preconditioners for advection-dominated problems including the unsteady Navier-Stokes equations.*

1 Introduction

One can express the solution to a PDE as an integral convolution of the associated Green's function with forcing and boundary terms. Upon discretising, the solution is the product of a discrete Green's function matrix with a vector right hand side; in general the matrix is full-rank. However, if the Green's function has a decaying kernel then the matrix can be well-represented by a low-rank approximation. This concept is sometimes used to precondition elliptic and parabolic equations arising from in a variety of applications [1, 2]. However, it has not been applied to problems featuring advection until now. The presence of advective terms in a PDE operator introduces directionality and non-locality: the effect of conditions far upstream is felt strongly at a particular location. Therefore, increasing separation does not imply decreasing influence, and the degree to which the matrix rank can be reduced is limited.

This paper presents a new direct solver for the advection-diffusion equation based on discrete Green's functions and low-rank matrix approximation. The problem of non-locality is dealt with by using a semi-implicit time discretisation to remove the advective term from the discrete system matrix. Specifically, the advective part is treated explicitly in time and the diffusive part implicitly in time, thereby transforming the equation into a steady heat equation with source terms at each timestep. The associated Green's function has the required decaying property, allowing the use of low-rank approximations.

Discretisation of the integral convolution can be achieved by a number of methods. In this paper, two simple midpoint-rule approximations are employed and their stability and accuracy assessed. Computational cost is then reduced by using several straightforward low-rank approximations of the discrete Green's function matrix. A series of numerical experiments in one dimension (1D) verify the analysis and demonstrate proof of concept. The method described in this paper is a template for more advanced strategies, in particular the use of the Fast Multipole Method (FMM) to compute optimal low-rank approximations in 2D and 3D bounded domains.

FMM is a hierarchical algorithm for computing matrix-vector products involving N unknowns in $\mathcal{O}(N \log N)$ or even $\mathcal{O}(N)$ arithmetic operations to a given error tolerance [3]. It is commonly applied to N-body problems in astrophysics and molecular dynamics, and is beginning to be applied to solving elliptic PDEs arising in many areas of computational physics as well [4, 5, 6]. Recently, it has been applied as a preconditioner for elliptic PDEs in a technique known as Inverse FMM (IFMM) [7, 8]. In all existing applications, the governing physics are such that the mutual influence of one spatial location on another decays with separation distance. FMM exploits this quality by hierarchically compressing interactions between well-separated points into lower-dimensional spaces, reducing the number of interactions to be computed. The analogy with the multigrid (MG) method is strong: both are hierarchical and act mainly on low-frequency error components. It is envisioned that FMM could be competitive with MG for preconditioning the system of equations arising from unsteady 3D Navier-Stokes problems.

2 Method

2.1 Discretisation of advection-diffusion equation

We consider the advection-diffusion equation in one dimension (1D) on a finite domain $x \in V : [0, L]$ with boundary S

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} = 0, \qquad x \in V, \tag{1a}$$

$$u(x, 0) = u_0(x), \qquad x \in V, \tag{1b}$$

$$u(x, t) = g(x, t), \qquad x \in S, \tag{1c}$$

where u is a smooth scalar field, a is the (positive) advection coefficient and ν is the diffusion coefficient. Dirichlet, Neumann and Robin boundary conditions can be specified for the problem and the domain L may be periodic.

We subdivide the domain with N uniform intervals of size $\Delta x = L/N$ and choose timestep Δt. Now we write (1) in semi-discrete form using matrix notation:

$$\frac{d\mathbf{u}}{dt} + A\mathbf{u} - D\mathbf{u} = \mathbf{b}, \tag{2}$$

where $\mathbf{u}$ is the solution vector, A and D are the discrete advection and diffusion operators and $\mathbf{b}$ is the vector containing boundary terms. We now discretise in time using a semi-implicit splitting scheme for the advective and diffusive terms:

$$\frac{d\mathbf{u}}{dt} - D\mathbf{u}^{n+1} \;=\; \mathbf{b} - A\mathbf{u}^n. \tag{3}$$

For simplicity, let $\mathbf{b} = 0$ and impose periodic boundary conditions. The spatial derivatives are discretised by finite differences with an upwind scheme for advection:

$$u_i^{n+1} - \frac{\nu\Delta t}{\Delta x^2}(u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}) = u_i^n - \frac{a\Delta t}{\Delta x}(u_i^n - u_{i-1}^n). \tag{4}$$

The advective and diffusive CFL numbers are denoted $c_A = \frac{a\Delta t}{\Delta x}$ and $c_D = \frac{\nu\Delta t}{\Delta x^2}$ respectively. This is the simplest, first-order semi-implicit method for time-dependent PDEs; higher-order methods could also be used for greater accuracy [9].

2.2 Green's function

The semi-implicit discrete equation (3) is in the form of an initial-boundary value problem governed by the 1D heat equation:

$$\left(\frac{\partial}{\partial t} - \nu\frac{\partial^2}{\partial x^2}\right) u(x, t) = f(x, t), \qquad x \in V, \tag{5a}$$

$$u(x, 0) = u_0(x), \qquad x \in V, \tag{5b}$$

$$u(x, t) = g(x, t), \qquad x \in S, \tag{5c}$$

where in our case the source term $f(x, t) = -a\frac{\partial u^n}{\partial x}$ and u^n denotes the solution at the previous timestep. The solution of (5) is expressed in terms of a Green's function $G(x - x', t - t')$ as the

sum of three integrals accounting for source terms, boundary and initial conditions [10]:

$$\begin{aligned} u(x,t) &= \int_0^T \int_V G(x-x',t-t')f(x',t')dx'dt' \\ &\quad + \int_V G(x-x',t-0)u_0(x')dx' \\ &\quad - \int_0^T \int_S \frac{\partial G}{\partial n} g(x',t')dS(x')dt', \quad x \in \mathbb{R}, t > t'. \end{aligned} \tag{6}$$

The Green's function satisfies

$$\left(\frac{\partial}{\partial t} - \nu \frac{\partial^2}{\partial x^2}\right) G(x-x',t-t') = \delta(x-x',t-t'), \tag{7}$$

where $\delta(x-x',t-t')$ is the Dirac delta function. The expression for Green's function in one dimension is [10]:

$$G(x-x',t-t') = (4\pi\nu(t-t'))^{-1/2} e^{-\frac{(|x-x'|)^2}{4\nu(t-t')}}. \tag{8}$$

Hereafter, only periodic (free-space) problems are considered so the third term is dropped. Let us treat the $n+1$th timestep as an initial-boundary value problem from time t to $t+\Delta t$ with initial conditions $u(x,t) = u^n(x)$. The solution $u(t+\Delta t) = u^{n+1}(x)$ is given by

$$\begin{aligned} u^{n+1}(x) &= \int_t^{t+\Delta t} \int_V G(x-x',t-t')f(x',t')dx'dt' \\ &\quad + \int_V G(x-x',t+\Delta t-t)u^n(x')dx'. \end{aligned} \tag{9}$$

Taking a backwards-in-time approximation of the first integral, we obtain

$$\begin{aligned} u^{n+1}(x) &\approx \Delta t \int_V G(x-x',t+\Delta t-t)f(x',t)dx' \\ &\quad + \int_V G(x-x',t+\Delta t-t)u^n(x')dx' \\ &= \int_V (4\pi\nu\Delta t)^{-1/2} e^{-\frac{(|x-x'|)^2}{4\nu\Delta t}} [u^n + \Delta t f(x',t)]dx'. \end{aligned} \tag{10}$$

Note that the forcing term $u^n + \Delta t f(x',t)$ is identical to the right-hand side of the semi-implicit finite difference relation (4). The Green's function can be expressed in terms of a scaled wavenumber $\epsilon = (4\nu\Delta t)^{-1/2}$ and radial distance $r = |x-x'|$ as

$$G(r,\epsilon) = \frac{\epsilon}{\sqrt{\pi}} e^{-\epsilon^2 r^2}. \tag{11}$$

2.3 Integral approximation

The continuous integral (10) is approximated as a sum of integrations over each interval using the midpoint rule[1]. The mesh nodes are the points $x_i = \{0, \Delta x, \ldots, L-\Delta x, L\}$. Intervals can

[1]Caveat: the exponential function is not smooth at $x = x'$. In approximations that sample this point, the error may be dominated by this value.

be defined with the nodes either as their midpoints (*node-centred* scheme) or endpoints (*face-centred* scheme). Figure 1 (a) illustrates the node-centred scheme. In the case of non-uniform mesh spacing and in higher dimensions, it can be generalised as integration on a dual mesh. Using the node-centred scheme the solution at a node x_i is:

$$\begin{aligned} u_i^{n+1} &= M_{ij}(u_j^n + \Delta t f_j) && (12)\\ &= \sum_{j=1}^{N} \frac{\epsilon\Delta x}{\sqrt{\pi}} e^{-(|j-i|\epsilon\Delta x)^2}[(1-c_A)u_j^n + c_A u_{j-1}^n]. && (13) \end{aligned}$$

The $[N \times N]$ matrix M with entries $m_{ij} = \frac{\epsilon\Delta x}{\sqrt{\pi}} e^{-(|j-i|\epsilon\Delta x)^2}$ is symmetric positive-definite, diagonally dominant and circulant. Periodicity is imposed on j such that if $j - i > N/2$ then $j = j - N$ and similarly, if $j - i < N/2$ then $j = j + N$.

The face-centred scheme is illustrated in Figure 1 (b). Using this scheme, the solution at a node x_i is defined as:

$$\begin{aligned} u_i^{n+1} &= M_{ij}^F(u_j^n + \Delta t f_j) && (14)\\ &= \sum_{j=1}^{N} \frac{\epsilon\Delta x}{\sqrt{\pi}} e^{-((|j-i|+1/2)\epsilon\Delta x)^2}[(1-c_A)u_j^n + c_A u_{j-1}^n]. && (15) \end{aligned}$$

The salient difference between these two methods is that the latter does not include the peak of the Green's function. Although not critical for our Green's function, similar PDEs have Green's functions with singularities. In those cases, the face-centred scheme is preferable. Using either scheme, the error η_i at a point x_i is the sum of the errors in each interval:

$$\eta_i = \left| \int_L G(x,x')dx' - \sum_{j=1}^{N} \Delta x G(x_j, x_i) \right| \leq \sum_{j=1}^{N} \frac{\Delta x^3}{24} \left\| \frac{d^2 G}{dx^2} \right\|_{L_\infty(j)}. \quad (16)$$

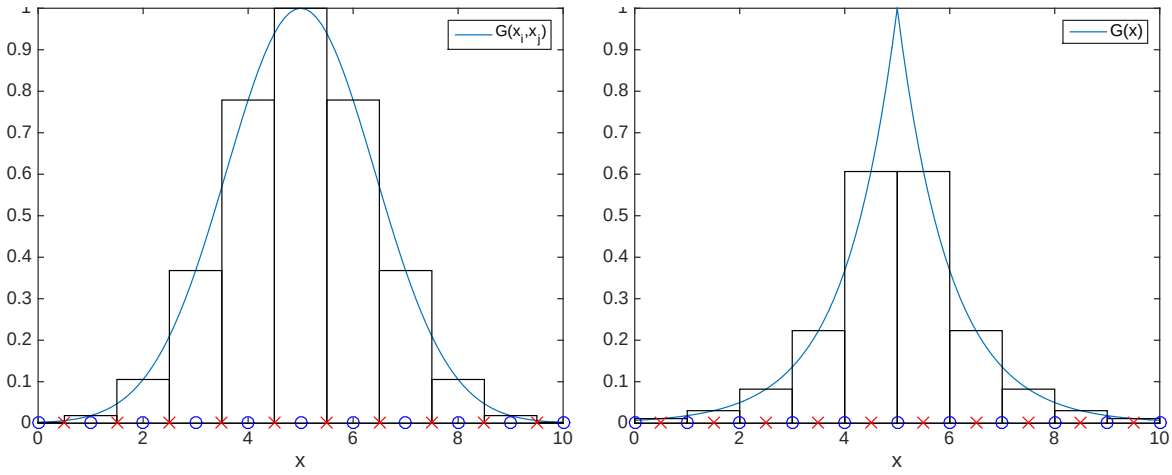
With the node-centred scheme the error is

$$\eta_i \leq \sum_{j=1}^{N} \frac{\Delta x^3}{24} \left\| -\frac{\epsilon}{2\sqrt{\pi}} e^{-(|j-i|\epsilon\Delta x)^2}(2(|j-i|\epsilon\Delta x)^2 - 1) \right\|_{L_\infty(j)} \leq \frac{L\epsilon\Delta x^2}{48\sqrt{\pi}}. \quad (17)$$

Therefore second-order convergence of the solution is expected. The face-centred scheme has the error

$$\begin{aligned} \eta_i &\leq \sum_{j=1}^{N} \frac{\Delta x^3}{24} \left\| -\frac{\epsilon}{2\sqrt{\pi}} e^{-(|j-i+1/2|\epsilon\Delta x)^2}(2(|j-i+1/2|\epsilon\Delta x)^2 - 1) \right\|_{L_\infty(j)}\\ &\leq \frac{L\epsilon\Delta x^2}{48\sqrt{\pi}} e^{-\epsilon\Delta x/2}(1-(\epsilon\Delta x)^2)\\ &= \frac{L\epsilon\Delta x^2}{48\sqrt{\pi}}(1 - \epsilon\Delta x/2 + \mathcal{O}(\Delta x^2))(1-(\epsilon\Delta x)^2)\\ &= \frac{L\epsilon\Delta x^2}{48\sqrt{\pi}}, \end{aligned} \quad (18)$$

where $e^{-\epsilon\Delta x/2}$ has been expanded as a Taylor series about $\Delta x = 0$. Keeping the leading-order term only, the face-centred error (18) reduces to the same expression as for the node-centred scheme (17). Higher-order approximations may also be employed for the integral. Note that this is not the only source of error: the spatial and temporal discretisations also contribute.



(a) Integration on dual mesh (node-centred) (b) Integration on primary mesh (face-centred)

Figure 1: Approximations of Green's function with $N = 10$ centred at 5th interval. Primary mesh nodes shown in blue and midpoints in red.

2.4 Stability analysis

2.4.1 Node-Centred Scheme

Consider a uniform initial flow field $u(x) = C$: the solution is constant throughout time. Therefore a necessary, but not sufficient, condition for stability of the integral equation (12) (and also (14)) is that the row sum of matrix M (resp. M^F) must not be greater than one. For the sake of accuracy the row sum must be as close to one from below as possible. Periodic boundaries and uniform Δx are assumed. Thanks to the circulant property of M, the sum is identical for all rows l. The node-centred matrix row sum stability constraint is written:

$$\begin{aligned} \frac{\epsilon\Delta x}{\sqrt{\pi}} \sum_{j=1}^{N} e^{-(|j-l|\epsilon\Delta x)^2} &\leq 1 \\ \frac{\epsilon\Delta x}{\sqrt{\pi}} \left[1 + 2 \sum_{j=1}^{(N-1)/2} e^{-(j\epsilon\Delta x)^2} \right] &\leq 1. \end{aligned} \tag{19}$$

Now $\epsilon\Delta x = c_D^{-1/2}$ is substituted into (19) to obtain a function $f(c_D)$:

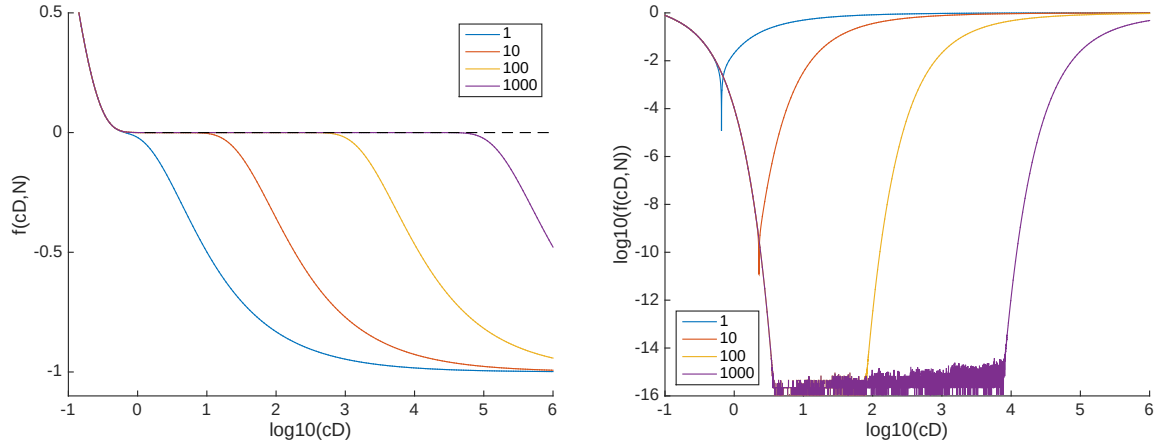
$$f(c_D) = \frac{1}{\sqrt{\pi}} c_D^{-1/2} \left[1 + 2 \underbrace{\sum_{j=1}^{(N-1)/2} (e^{1/c_D})^{-j^2}}_{S_N} \right] - 1 \leq 0. \tag{20}$$

The sum S_N is convergent for $e^{1/c_D} > 1$, i.e. $c_D > 0$. However, no expression can be found for the limit of the sum as $N \to \infty$. Numerical calculations with large N provide the following information:

1. as $c_D \to 0, S_N \to 0$,
2. as $c_D \to \infty, S_N \to N$,

3. at $c_D = 1, S_N = 0.3863$.

The stability function (20) is plotted (20) in Figure 2 on (a) semi-log and (b) log-log scales (real part) for several values of N. The function lies below the c_D axis above a certain value, i.e. there is a *minimum* bound on c_D for the inequality to be satisfied. The values of c_D at which the function crosses the axis (spikes on the log-log plot) increases with N. In the simplest case of $N = 1$, the stability condition is $c_D \geq 0.66$. At $N = 10$, $c_D > 2.28$; at $N = 100$, $c_D > 3.73$, and at $N = 1000$, $c_D > 3.73$. The reason for the limited stability of the node-centred scheme is the presence of the central term at $j = i$ in (19). Accuracy also depends on c_D: there is a range of c_D in which $f(c_D) \approx 0$ and $f(c_D) \leq 0$ with the zero-crossing defining the lower bound. The extent of the range increases with N. At larger N, the row sum equals one to machine precision across several orders of magnitude in c_D.



(a) Linear axes

(b) Re$(f(c_D, N))$ Log axes

Figure 2: Stability condition on node-centred scheme as a function of c_D for $N = \{1, 10, 100, 1000\}$.

2.4.2 Face-Centred Scheme

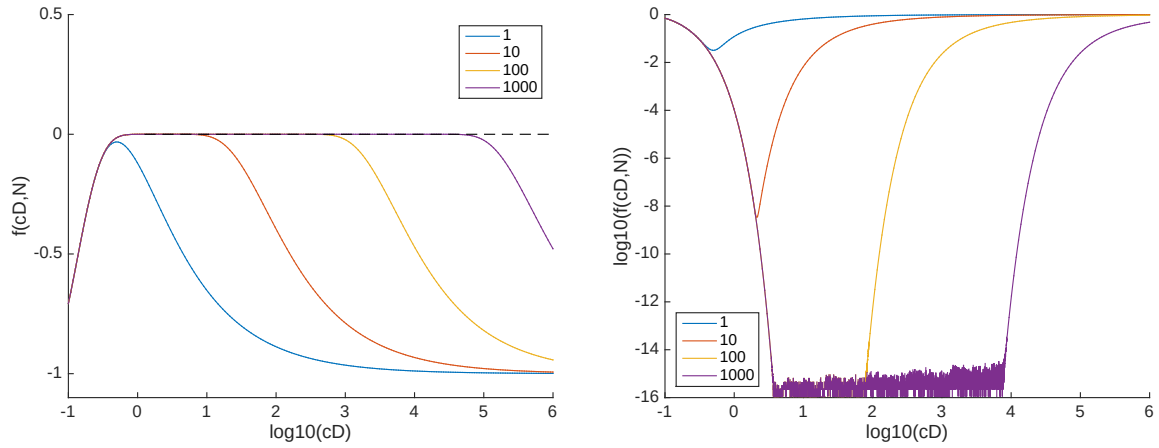
The matrix row sum using the face-centred scheme is:

$$\begin{aligned} \frac{\epsilon\Delta x}{\sqrt{\pi}} \sum_{j=1}^{N} e^{-((|j-l|+1/2)\epsilon\Delta x)^2} &\leq 1 \\ 2\frac{\epsilon\Delta x}{\sqrt{\pi}} \sum_{j=1}^{N/2} e^{-((j-1/2)\epsilon\Delta x)^2} &\leq 1. \end{aligned} \tag{21}$$

Again, we substitute $\epsilon\Delta x = c_D^{-1/2}$ into (21) to obtain:

$$f(c_D) = \frac{2}{\sqrt{\pi}} c_D^{-1/2} \sum_{j=1}^{N/2} (e^{1/c_D})^{-(j-1/2)^2} - 1 \leq 0. \tag{22}$$

The stability function is plotted in Figure 3. Now, the inequality is satisfied for all c_D. Nevertheless, the accuracy is still contingent upon c_D. As with the node-centred scheme, it is clear from Figure 3 (a) that the face-centred scheme is accurate over a finite region that increases rapidly with N. At $N = 1000$, machine-precision accuracy is achieved from $c_D = 3.75$ to $c_D \approx 10000$.



(a) Semi-log axes

(b) Log-log axes

Figure 3: Stability condition on face-centred scheme as a function of c_D for $N = \{1, 10, 100, 1000\}$.

2.5 von Neumann analysis

The above conditions are necessary but not sufficient for stability of the node- and face-centred schemes. Von Neumann's method is used to derive a second condition which, together with the first condition, is sufficient for stability of the method. We analyse the node-centred scheme first. The error term at time level $n+1$ and location l is related to the error at the previous time level by:

$$\begin{aligned} e^{a(t+\Delta t)}e^{ik_m x} &= \frac{\epsilon\Delta x}{\sqrt{\pi}}\Bigg[[(1-c_A)e^{at}e^{ik_m x} + c_A e^{at}e^{ik_m(x-\Delta x)}] \\ &+ \sum_{j=1}^{(N-1)/2} e^{-(j\epsilon\Delta x)^2}[(1-c_A)e^{at}e^{ik_m(x-j\Delta x)} + c_A e^{at}e^{ik_m(x-(j-1)\Delta x)}] \\ &+ \sum_{j=1}^{(N-1)/2} e^{-(j\epsilon\Delta x)^2}[(1-c_A)e^{at}e^{ik_m(x+j\Delta x)} + c_A e^{at}e^{ik_m(x+(j-1)\Delta x)}]\Bigg], \end{aligned} \tag{23}$$

where the first term is the contribution from point $j = l$ and the next two terms are the contributions from points to the left and right of l respectively. Upon dividing through by $e^{at}e^{ik_m x}$ and substituting the relations $e^{-(j+1)ik_m\Delta x} = e^{-jik_m\Delta x}e^{-ik_m\Delta x}$ and $e^{(j-1)ik_m\Delta x} = e^{jik_m\Delta x}e^{-ik_m\Delta x}$, we obtain

$$\begin{aligned} e^{a\Delta t} &= \frac{\epsilon\Delta x}{\sqrt{\pi}}\left[(1-c_A+c_A e^{-ik_m\Delta x})\left(1+\sum_{j=1}^{(N-1)/2} e^{-(j\epsilon\Delta x)^2}\left(e^{-jik_m\Delta x}+e^{jik_m\Delta x}\right)\right)\right] \\ &= (1-c_A+c_A e^{-ik_m\Delta x})\underbrace{\frac{\epsilon\Delta x}{\sqrt{\pi}}\left(1+2\sum_{j=1}^{(N-1)/2} e^{-(j\epsilon\Delta x)^2}\cos(jk_m\Delta x)\right)}_{M}. \end{aligned} \tag{24}$$

For stability it is required that $|e^{a\Delta t}| \leq 1$. For the most restrictive condition on c_A term M must be maximised. Using $\max(|\cos(jk_m\Delta x)|) = 1$, M is reduced to the matrix row sum (19). Choosing c_D such that the stability function in (20) equals zero to a sufficient level of precision,

we have $M \approx 1$. Thus (24) reduces to

$$\left|1 - c_A + c_A e^{-ik_m \Delta x}\right| \leq 1. \tag{25}$$

This represents a circle of radius c_A and centre $1 - c_A$. Finally, we have the stability condition:

$$2c_A - 1 \leq 1 \quad \therefore c_A \leq 1. \tag{26}$$

This condition is typical of the upwind finite difference discretisation of the advection term and does not depend on the attributes of the Green's function method. An identical condition can also be derived for the face-centred scheme.

2.6 Low-rank approximation

In the two matrix methods defined above, entries far from the diagonal make a relatively small contribution to the solution so low-rank approximations to M or M^F can be justifiably used to reduce computational effort. The primary method of interest here is the fast multipole method (FMM), although this paper treats only the 1D case for which FMM is not applicable. After a generic description of FMM, two simple methods of constructing a low-rank matrix are presented. This work is a precursor step to using FMM for the 2D and 3D advection-diffusion equations.

Fast multipole method: The computational degrees of freedom are treated as a cloud of points (we do not specify the number of spatial dimensions d to keep the description generic). A k-level tree mesh is constructed in which level one consists of one box containing all points and the kth level consists of k^{2d} boxes. Each box at level k contains a small number of points, say p on average. A multipole expansion (polynomial series) of degree m is employed at the box centroid to describe the equivalent source due to all point sources in the box.

The interactions between any two boxes on the same level are classified as strong or weak based on their radii, on the separation between their centroids and on a threshold value θ. Strong interactions are computed directly between points. Weak interactions are computed by 'shifting' a multipole expansion from one box to the other. At levels 1 to $k-1$, a multipole expansion in a box is computed from multipole expansions inside its children. In this manner, only local interactions at level k are computed directly and all other interactions are computed approximately via multipole expansions and shifts. This saves considerable effort especially for very large N.

In an abstract sense FMM is a method for hierarchical sparsification of a dense matrix. Strong interactions are clustered around the diagonal. Away from the diagonal, zeros are introduced by the multipole representation. The degree of sparsification depends on the user-defined parameters p, m and θ. Accuracy and computational cost can be fine-tuned via these parameters.

Threshold value: A minimum value is specified, below which the matrix entry is set to zero. The sparse matrix M^{L1} is constructed according to

$$m_{ij}^{L1} = \begin{cases} m_{ij}, & m_{ij} > 1.0/N, \\ 0, & \text{otherwise} \end{cases} \tag{27}$$

Likewise, the very sparse matrix M^{L2} is constructed with the threshold value of $1.0/\sqrt{N}$.

Specified bandwidth: Let P be the low-rank matrix bandwidth; in this case $P = \text{round}(N/8)$. Then the entries of sparse matrix M^{L3} are given by

$$m_{ij}^{L3} = \begin{cases} m_{ij}, & i - P \leq j \leq i + P, \\ 0, & \text{otherwise} \end{cases} \tag{28}$$

with the same periodic indexing as the full-rank matrix.

2.7 Comparison to iterative method

To compute a reference solution, an iterative method is employed based on the same implicit-explicit flux splitting:

$$Au_i^{n+1} = \left(1 - \nu\Delta t \frac{d^2}{dx^2}\right) u_i^{n+1} = (1 - c_A)u_i^n + c_A u_{i-1}^n, \tag{29}$$

where the matrix A is

$$A = \begin{bmatrix} 1 + 2c_D & -c_D & \dots & -c_D \\ -c_D & 1 + 2c_D & -c_D & \\ & \ddots & \ddots & \ddots \\ & -c_D & 1 + 2c_D & -c_D \\ -c_D & \dots & -c_D & 1 + 2c_D \end{bmatrix}. \tag{30}$$

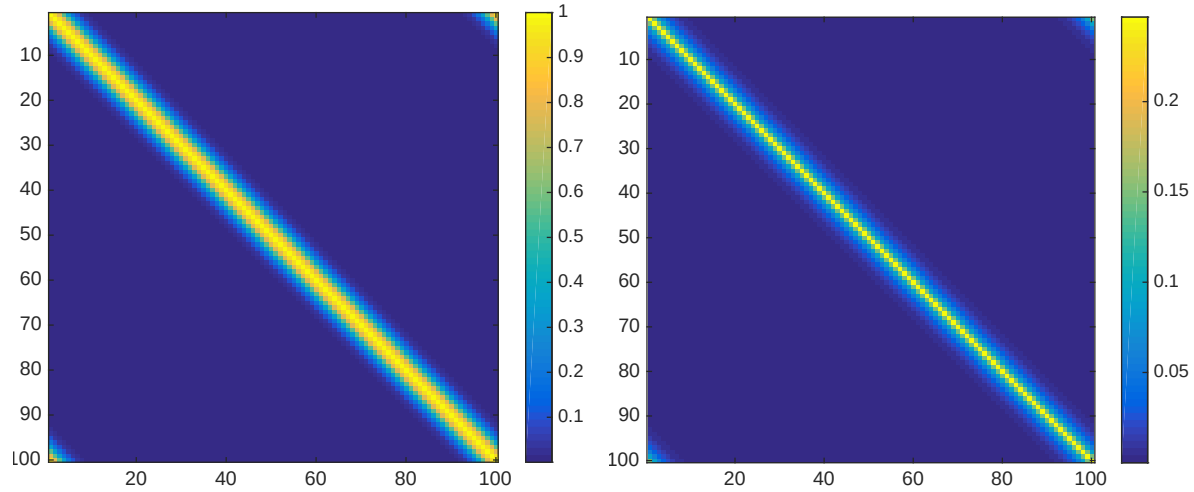
The Matlab `linsolve` function was used with `opts.SYM=true,opts.POSDEF=true` and all other options set to false. It is not a very efficient method but serves to provide a reference solution.

3 Numerical tests

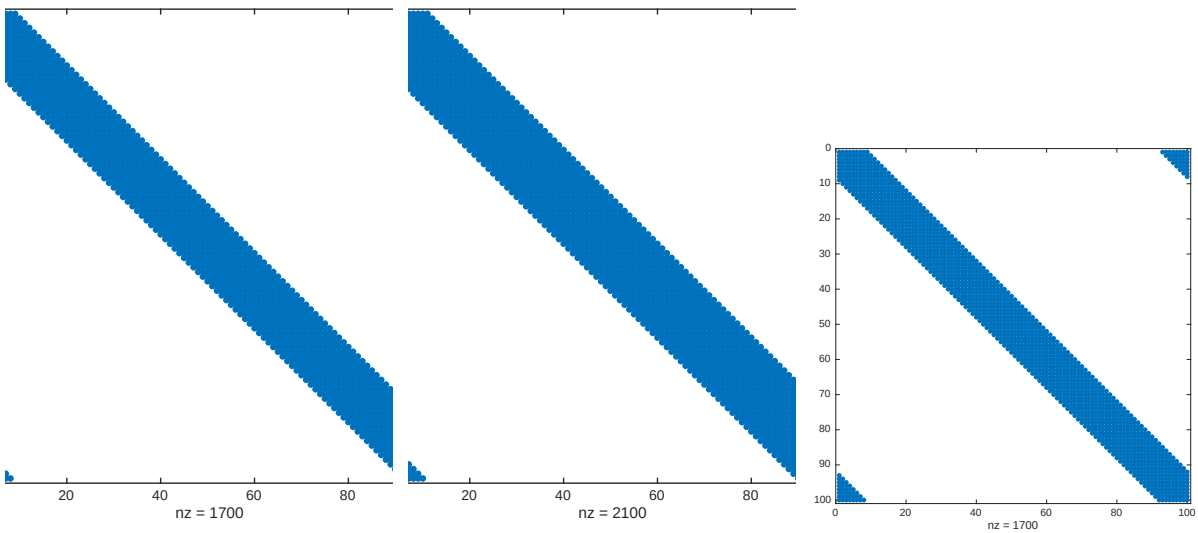
The full-rank and sparse matrix-vector products and the iterative method were implemented in Matlab to solve the 1D advection-diffusion equation with periodic boundary conditions. The advective and diffusive constants were set to $a = 1.0$ and $\nu = 0.1$ and simulations were run for 100 timesteps. Discrete optimisation of the diffusive CFL number was performed for the node- and face-centred schemes. The smallest values of c_D that minimised the functions in Figures 2 and 3 were found for each value of N. These in turn dictated the timesteps, which were sufficiently small that the advective CFL condition (26) was also satisfied. Table 1 lists the optimised timesteps Δt for each N and each scheme. The values scale approximately with N^{-2} and the node-centred scheme has an optimal timestep slightly larger than that of the face-centred scheme.

Figure 4 (a) shows the full-rank node-centred matrix M coloured by magnitude. For comparison, Figure 4 (b) shows the inverse of matrix A in (30). Figures (c,d,e) show the non-zeros of sparse matrices M^{L1}, M^{L2} and M^{L3}. Table 2 shows the row sum and number of nonzeros in each matrix.

Figure 5 plots the solutions starting from (a) sinusoidal and (b) Gaussian initial conditions with N=100. Node-centred solutions are compared to the reference solution computed at nodes, and the face-centred scheme to the reference solution computed at interval midpoints. The M^{L2} solution displays a large amplitude (dissipative) error, as expected from the fact that it has a row sum well below one. All the other node-centred solutions overlie the reference solution. The face-centred solution has a phase lag (dispersion error).



(a) M (b) A^{-1}



(c) M^{L1} (d) M^{L2} (e) M^{L3}

Figure 4: Colour plot of (a) full-rank matrix, (b) inverse iterative matrix, (c,d,e) non-zeros of sparse matrices.

N	Δt (node-centred)	Δt (face-centred)
100	$3.85e-3$	$3.75e-3$
200	$9.64e-4$	$9.37e-4$
400	$2.41e-4$	$2.34e-4$
800	$6.02e-5$	$5.86e-5$
1600	$1.51e-5$	$1.46e-5$
3200	$3.76e-6$	$3.66e-6$

Table 1: Optimal timesteps.

matrix	row sum	NNZ
A^{-1}	1.00000	10000
M	1.00000	10000
M^F	1.00000	10000
M^{L1}	0.99791	1700
M^{L2}	0.95359	1100
M^{L3}	0.99999	4900

Table 2: Matrix row sums and no. of nonzeros, N=100.

3.1 Error Convergence

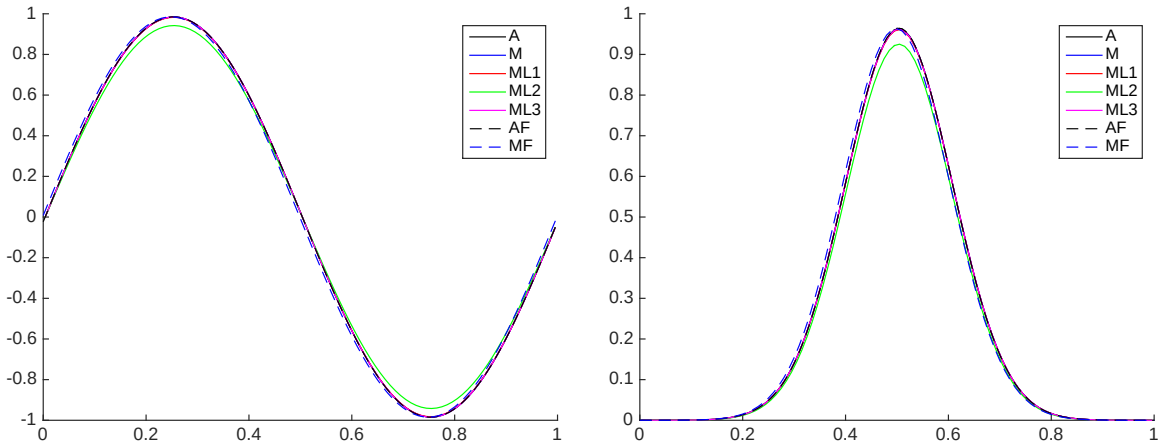
The L_2 norm of the solution errors with respect to the reference solutions (at the nodes or faces as appropriate) was calculated after one timestep, successively doubling N from 100 up to 3200. Figure 6 (a) shows the error convergence on a log-log plot, starting from the sinusoidal initial condition. The full-rank node-centred matrix M results in a convergence rate of approximately fifth order and approaches machine precision at large N. The M^{L3} error converges at an identical rate and has the same magnitude: this low-rank approximation does not reduce accuracy. The M^{L1} and M^{L2} errors converge at between first and second order. The full-rank face-centred matrix M^F error also converges at between first and second order. Figure 6 (b) shows the errors when starting from the Gaussian initial condition, with similar results to the (a).

3.2 CPU time

The time taken to compute one timestep with N=100 to N=3200 was found. The times to construct A, M, M^{L1} etc. were not included. Figure 7 shows the CPU times. Similar results are obtained from both initial conditions. The reference solution is much slower than the direct solutions and the time increases faster than N^2. Direct solution times increase at about first order. The low-rank matrices constructed using the minimum-value rule, M^{L1} and M^{L2}, are by far the fastest at large N. Matrix M^{L3} with prescribed rank is as slow as the full-rank matrices at large N. Given that the error is as small as the full-rank scheme, this low-rank scheme clearly does not alter the matrix noticeably. Specifying a smaller bandwidth would have a more pronounced effect.

4 Conclusions

This paper presents a simple scheme for direct solution of the one-dimensional advection-diffusion equation in a periodic setting. By using a semi-implicit time discretisation the advective term becomes a forcing function and the left-hand side becomes the heat equation.



(a) sinusoidal initial condition

(b) Gaussian initial condition

Figure 5: Computed solutions with (a) sinusoidal, (b) Gaussian initial conditions. —: reference (node), – –: reference (face), blue: M, red: M^{L1}, green: M^{L2}, magenta: M^{L3}, blue dashes: M^F.

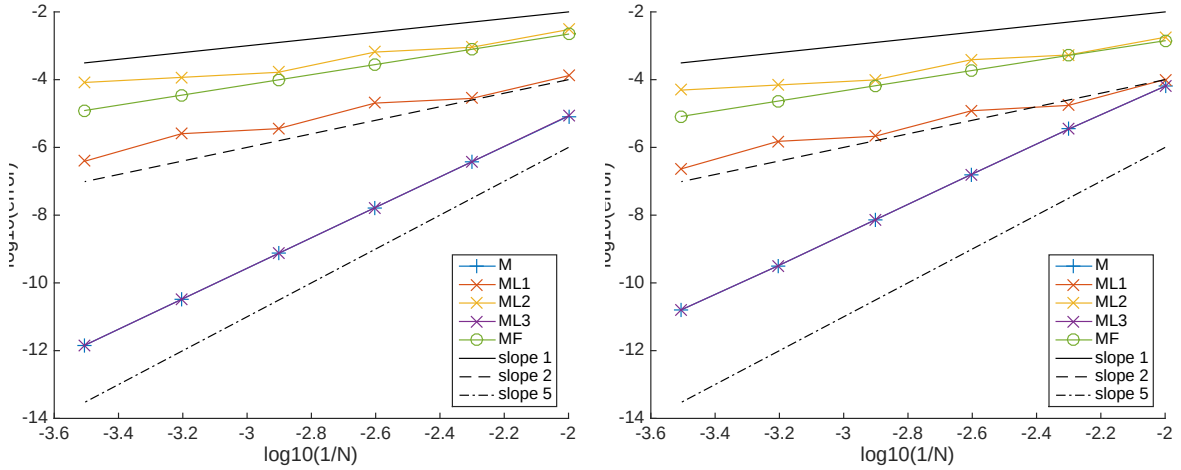
The associated Green's function has a decaying kernel, making it highly suitable for low-rank discrete approximation. Similar methods have been very successful in solving purely elliptic equations, but this is the first known application to a hyperbolic equation.

Several different full- and low-rank discrete approximations of the kernel were analysed and found to have differing stability limits and lead to differing error convergence rates. All obtained approximately linear scaling of CPU time with N. The node-centred full-rank scheme was not unconditionally stable but had the best accuracy and the error converged at fifth order. Almost machine-precision accuracy (with respect to the reference iterative solution) was attained by this scheme at $N = 3200$ in a single matrix-vector multiplication. The face-centred scheme, although unconditionally stable, was less accurate and had a much lower convergence rate. Specifically, the numerical wave speed was under-predicted although the amplitude was well-predicted. Low-rank node-centred approximations in which a threshold matrix entry was defined were shown to save considerable computational effort but were less accurate and had a lower error convergence rate. The balance of accuracy and cost may be problem-dependent and more work is needed to determine robust sparsification strategies.

The new method is a template for finding fast approximate solutions of advection-dominated problems. FMM is proposed as the ideal low-rank approximation algorithm for this application (although it is not suitable for 1D problems). FMM allows control over the balance of accuracy and speed, and scales very well to large numbers of parallel processes. In terms of solving PDEs, FMM is somewhat analogous to multigrid, in that low-frequency error components (i.e. distant interactions) are rapidly diminished. Indeed, FMM could become a competitor to multigrid as a solver/preconditioner for hyperbolic PDEs including the Navier-Stokes equations. Future work will extend the low-rank direct solution method to 2D and 3D problems on bounded domains.

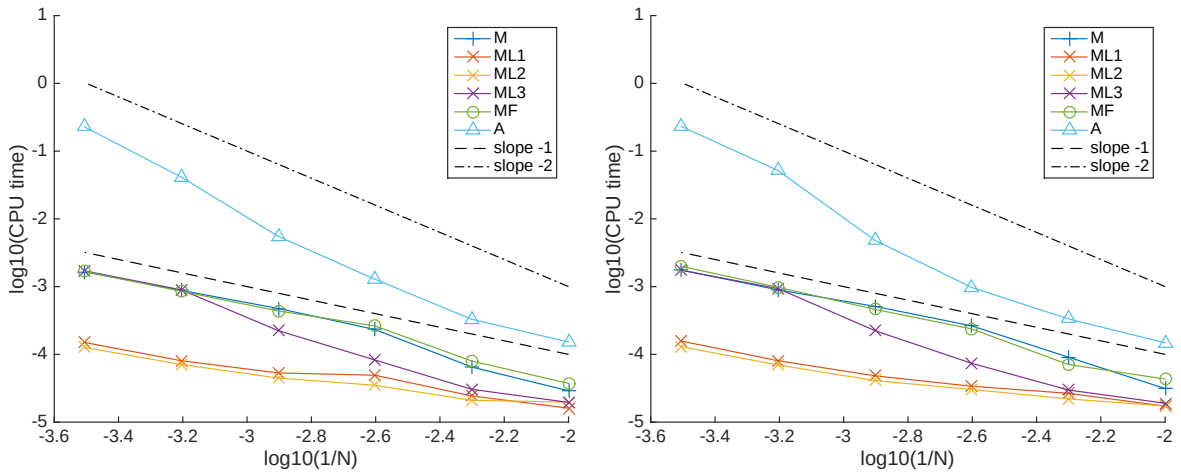
REFERENCES

[1] M. Bebendorf, Hierarchical lu decomposition-based preconditioners for BEM, Computing 74 (3) (2005) 225–247.



(a) sinusoidal initial condition (b) Gaussian initial condition

Figure 6: Error convergence of direct solutions with respect to reference solution, (a) sinusoidal and (b) Gaussian initial conditions.



(a) sinusoidal initial condition (b) Gaussian initial condition

Figure 7: Time to solution.

[2] H. Branden, P. Sundqvist, An algorithm for computing fundamental solutions of difference operators, Numerical Algorithms 36 (4) (2004) 331–343.

[3] L. Greengard, V. Rokhlin, A fast algorithm for particle simulations, Journal of Computational Physics 73 (1987) 325–348.

[4] F. Ethridge, L. Greengard, A new fast-multipole accelerated Poisson solver in two dimensions, SIAM J. Sci. Comput. 23 (3) (2001) 741–760.

[5] L. Greengard, D. Gueyffier, P.-G. Martinsson, V. Rokhlin, Fast direct solvers for integral equations in complex three-dimensional domains, Acta Numerica 18 (2009) 243–275.

[6] R. Yokota, H. Ibeid, D. Keyes, Fast multipole method as a matrix-free hierarchical low-rank approximation, arXiv:1602.02244 [cs.NA] (2016).

[7] S. Ambikasaran, E. Darve, The inverse fast multipole method, arXiv:1407.1572 (2014).

[8] P. Coulier, H. Pouransari, E. Darve, The inverse fast multipole method: using a fast approximate direct solver as a preconditioner for dense linear systems, arXiv:1508.01835 (2015).

[9] U. Ascher, S. Ruuth, B. Wetton, Implicit-explicit methods for time-dependent partial differential equations, SIAM Journal on Numerical Analysis 32 (3) (1995) 797–823.

[10] P. Olver, introduction to partial differential equations, Springer, 2013.