Advances in Applied Mathematics and Mechanics
Adv. Appl. Math. Mech., Vol. **5**, No. 4, pp. 528-547

DOI: 10.4208/aamm.13-13S09
August 2013

B-Spline Gaussian Collocation Software for Two-Dimensional Parabolic PDEs

Zhi Li and Paul Muir*

Mathematics and Computing Science, Saint Mary's University, Halifax, Nova Scotia, Canada B3H 3C3

Received 5 June 2012; Accepted (in revised version) 8 September 2012

Available online 7 June 2013

Abstract. In this paper we describe new *B*-spline Gaussian collocation software for solving two-dimensional parabolic partial differential equations (PDEs) defined over a rectangular region. The numerical solution is represented as a bi-variate piecewise polynomial (using a tensor product *B*-spline basis) with time-dependent unknown coefficients. These coefficients are determined by imposing collocation conditions: the numerical solution is required to satisfy the PDE and boundary conditions at images of the Gauss points mapped onto certain subregions of the spatial domain. This leads to a large system of time-dependent differential algebraic equations (DAEs) which is solved using the DAE solver, DASPK. We provide numerical results in which we use the new software, called BACOL2D, to solve three test problems.

AMS subject classifications: 65M20, 65M70

Key words: Collocation, *B*-splines, two-dimensional partial differential equations, differential-algebraic equations.

1 Introduction

In this paper, we discuss a numerical algorithm that uses high order methods in time and space to solve a system of n parabolic partial differential equations (PDEs) in two space dimensions. We assume a problem class having the form

$$u_t(x,y,t)=f\big(x,y,t,u(x,y,t),u_x(x,y,t),u_y(x,y,t),u_{xx}(x,y,t),u_{xy}(x,y,t),u_{yy}(x,y,t)\big), \quad (1.1)$$

for $(x,y,t)\in\Omega\times(t_0,t_{out}]$, where $\Omega=\{(x,y)|a<x<b,\ c<y<d\}$ and $u(x,y,t)$ is a vector function with n components. The boundary conditions at $x=a$ and $x=b$ are

$$g_a(y,t,u(a,y,t),u_x(a,y,t),u_y(a,y,t))=0, \quad g_b(y,t,u(b,y,t),u_x(b,y,t),u_y(b,y,t))=0,$$

*Corresponding author.
Email: lizhiupc@gmail.com (Z. Li), muir@smu.ca (P. Muir)

for $t\in(t_0,t_{out}]$, while the boundary conditions at $y=c$ and $y=d$ are

$$g_c(x,t,u(x,c,t),u_x(x,c,t),u_y(x,c,t))=0,\quad g_d(x,t,u(x,d,t),u_x(x,d,t),u_y(x,d,t))=0,$$

for $t\in(t_0,t_{out}]$. The initial conditions at $t=t_0$ are given by

$$u(x,y,t_0)=u_0(x,y),\quad (x,y)\in\Omega\cup\partial\Omega.$$

In the above, f, g_a, g_b, g_c, g_d and u_0 are given vector function with n components.

The numerical approach we employ uses two-dimensional (2D) B-spline Gaussian collocation to simultaneously discretize the x and y directions. The approximate solution is represented as a bi-variate piecewise polynomial (of degree p in x and degree q in y where $3\leq p,q\leq 11$) implemented in terms of a tensor product B-spline basis [3], with time-dependent unknown coefficients. We require the approximate solution to satisfy the PDE and boundary conditions at images of the Gauss points mapped onto certain subregions of the spatial domain, and this leads to a system of differential algebraic equations (DAEs). Since this DAE system is usually somewhat large, we use the DAE solver, DASPK [5], which is designed to efficiently solve large scale DAEs. DASPK uses a family of Backward Differentiation Formulas (BDFs) of orders 1 to 5 for the time integration. Our implementation of this approach is called BACOL2D. The BACOL2D software is a generalization of the software package, BACOL [30–32], designed for the numerical solution of systems of one-dimensional (1D) parabolic PDEs.

In Section 2, we provide a brief review of the related literature, focusing on work that features the use of collocation methods for 2D PDEs. We also review, in detail, the BACOL package since the 1D B-spline Gaussian collocation algorithm it employs is the basis for the 2D B-spline Gaussian collocation algorithm we consider in this paper. Section 3 discusses the BACOL2D implementation; this involves a description of the 2D B-spline Gaussian collocation algorithm, a discussion of the use of the DASPK package to solve the large DAE system arising from the collocation spatial discretization, and a description of a fast block LU algorithm for the treatment of certain structured matrices that arise during the computation. In Section 4 we present numerical results obtained by using BACOL2D to solve several test problems; these results allow us to experimentally demonstrate the order of convergence of the 2D collocation solutions. Section 5 provides our summary and identifies some areas for future work.

2 Background

2.1 Collocation methods for 2D PDEs

There is of course a very large body of literature on the numerical solution of PDEs–see, e.g., the recently published research texts [11, 18–20, 22] and references within. Here we focus on literature that considers collocation methods for 2D PDEs.

One of the papers most relevant to our work is [25] in which the authors considered the use of Hermite cubic spline collocation combined with a tensor product spatial basis to solve 2D elliptic PDEs. Another important paper in this area is [28] in which the author discussed the use of tensor product B-spline collocation for (non-time-dependent) 2D elasticity problems. The Ph.D. thesis [33] described the use of a parallel B-spline collocation method for 2D linear parabolic, separable PDEs. The paper [34] described the use of a 2D B-spline finite element method for the numerical solution of 2D PDEs.

The collocation methods mentioned above usually represent the numerical solution in terms of a spline basis and apply collocation at the Gaussian points on each element of the spatial domain; such methods are known as orthogonal spline collocation methods. A different class of collocation methods for 2D PDEs are the optimal spline collocation methods; these include the Quadratic Spline Collocation (QSC) and Cubic Spline Collocation (CSC) methods. These methods were used to solve 2D elliptic problems on rectangular domains in the papers [8, 12]. The advantage of these methods is that since they use only one collocation point per subinterval, the linear systems that arise are the smallest among all types of piecewise polynomial collocation methods for this problem class. Other related work has involved the development of special collocation software (GENCOL [13], HERMCOL/INTCOL [14]) for elliptic problems on rectangular domains.

For a 2D PDE whose solution exhibits rapid spatial variation, a useful approach for determining an adaptive spatial mesh is the moving mesh method (MMM). A MMM typically controls the mesh movement using a moving mesh PDE (MMPDE) [18]. The papers [6, 15–17] discussed the MMM for 2D problems. Other work in this area is the Ph.D. thesis [23] in which the CSC method is used together with the MMM approach.

2.2 Overview of BACOL

BACOL is designed to handle a system of n 1D PDEs of the form

$$u_t(x,t)=f(x,t,u(x,t),u_x(x,t),u_{xx}(x,t)), \tag{2.1}$$

for $x\in(a,b)$, $t\in(t_0,tout)$, where $u(x,t)$ is a vector function with n components. The initial conditions and separated boundary conditions are given by

$$u(x,t_0)=u_0(x), \quad b_L(t,u(a,t),u_x(a,t))=0, \quad b_R(t,u(b,t),u_x(b,t))=0,$$

where $x\in[a,b]$, $t\in(t_0,tout)$. The functions f, u_0, b_L and b_R are given vector functions with n components.

In BACOL, it is assumed that $[a,b]$ is partitioned by a spatial mesh, $a=x_0<x_1<\cdots<x_N=b$. The approximate solution is expressed in terms of piecewise polynomials, of a given degree p $(3\le p\le 11)$, associated with this mesh, with C^1-continuity imposed at the internal mesh points. The dimension of this piecewise polynomial subspace is $NC=N(p-1)+2$. To represent these piecewise polynomials, BACOL employs a B-spline

basis which is implemented using a collection of Fortran subroutines called the B-spline package [3]. The approximate vector solution, $U(x,t)$, is expressed in the form

$$U(x,t)=\sum_{j=1}^{NC} w_j(t)B_j(x), \tag{2.2}$$

where $w_j(t)$ is the vector of time-dependent B-spline coefficients multiplying the j-th B-spline basis function, $B_j(x)$. An important property of the B-spline basis is that on any subinterval $[x_{i-1},x_i)$, $i=1,\cdots,N$, at most $p+1$ basis functions, $\{B_m(x)\}_{m=(i-1)(p-1)+1}^{i(p-1)+2}$, are non-zero. This implies that, for $x\in[x_{i-1},x_i)$, the collocation solution can be expressed in the form

$$U(x,t)=\sum_{m=(i-1)(p-1)+1}^{i(p-1)+2} w_m(t)B_m(x). \tag{2.3}$$

Let $h_i=x_i-x_{i-1}$ and let $\{\rho_j\}_{j=1}^{p-1}$ be the images of the Gaussian points on $[0,1]$ with $0<\rho_0<\rho_1<\cdots<\rho_{p-1}<1$. Let the collocation points in the x direction be

$$\xi_1=x_0=a,\quad \xi_l=x_{i-1}+h_i\rho_j,\quad \xi_{NC}=x_N=b,$$

where $l=1+(i-1)\cdot(p-1)+j$, for $i=1,\cdots,N$, $j=1,\cdots,p-1$. The spatial discretization scheme employed by BACOL requires the approximate solution to satisfy the PDEs at the collocation point sequence, $\{\xi_l\}_{l=2}^{NC-1}$ (ξ_1, ξ_{NC} correspond to requiring $U(x,t)$ to satisfy the boundary conditions–see below). Substituting the approximate solution (2.3) and its derivatives, evaluated at ξ_l, $l=1+(i-1)\cdot(p-1)+j$, $i=1,\cdots N$, $j=1,\cdots,p-1$, into (2.1) gives the collocation conditions

$$\begin{aligned}&\sum_{m=(i-1)(p-1)+1}^{i(p-1)+2} w_m'(t)B_m(\xi_l)\\ &=f\left(\xi_l,t,\sum_{m=(i-1)(p-1)+1}^{i(p-1)+2} w_m(t)B_m(\xi_l),\sum_{m=(i-1)(p-1)+1}^{i(p-1)+2} w_m(t)B_m'(\xi_l)\sum_{m=(i-1)(p-1)+1}^{i(p-1)+2} w_m(t)B_m''(\xi_l)\right).\end{aligned} \tag{2.4}$$

An important feature of the BACOL code is that the boundary conditions are treated in their original form. Substituting the approximate solution (2.3) and its first derivative into the boundary conditions gives the algebraic equations

$$0=b_L\left(t,\sum_{m=1}^{p+1} w_m(t)B_m(a),\sum_{m=1}^{p+1} w_m(t)B_m'(a)\right), \tag{2.5a}$$

$$0=b_R\left(t,\sum_{m=(N-1)(p-1)+1}^{N(p-1)+2} w_m(t)B_m(b),\sum_{m=(N-1)(p-1)+1}^{N(p-1)+2} w_m(t)B_m'(b)\right). \tag{2.5b}$$

Considering the boundary conditions and collocation conditions together (in the order (2.5a), (2.4), (2.5b)), gives a DAE system of the form

$$A_x\underline{W}_t(t)=\underline{F}(t,\underline{W}(t)). \tag{2.6}$$

The vector $\underline{W}(t)$ is

$$\underline{W}(t)=\begin{bmatrix} w_1(t) \\ w_2(t) \\ \vdots \\ w_{NC}(t) \end{bmatrix},$$

and the right hand side vector $\underline{F}(t,\underline{W}(t))$ is

$$\underline{F}(t,\underline{W}(t))=\begin{bmatrix} b_L\left(t,\sum_{m=1}^{p+1}w_m(t)B_m(a),\sum_{m=1}^{p+1}w_m(t)B_m'(a)\right) \\ \vdots \\ f\left(\xi_l,t,\sum_{m=(i-1)(p-1)+1}^{i(p-1)+2}w_m(t)B_m(\xi_l),\sum_{m=(i-1)(p-1)+1}^{i(p-1)+2}w_m(t)B_m'(\xi_l),\right. \\ \left.\sum_{m=(i-1)(p-1)+1}^{i(p-1)+2}w_m(t)B_m''(\xi_l)\right) \\ \vdots \\ b_R\left(t,\sum_{m=(N-1)(p-1)+1}^{N(p-1)+2}w_m(t)B_m(b),\sum_{m=(N-1)(p-1)+1}^{N(p-1)+2}w_m(t)B_m'(b)\right) \end{bmatrix},$$

where $\xi_l\in[x_{i-1},x_i)$. The top and bottom blocks of rows of A_x correspond to the boundary conditions and are zero. The other block rows of A_x are associated with the left hand side of (2.4) and involve the evaluation of the B-spline basis functions at the collocation points. This matrix A_x, shown in Fig. 1, has a structure known as almost block diagonal (ABD)–see, e.g., [10]. The ith subblock, S_i, appearing in Fig. 1, is an $n(p-1)\times n(p+1)$ matrix given by

$$S_i=\begin{bmatrix} B_k(\xi_{k+1})I_n & B_{k+1}(\xi_{k+1})I_n & \cdots & B_{k+p}(\xi_{k+1})I_n \\ B_k(\xi_{k+2})I_n & B_{k+1}(\xi_{k+2})I_n & \cdots & B_{k+p}(\xi_{k+2})I_n \\ \vdots & \vdots & \ddots & \vdots \\ B_k(\xi_{k+p-1})I_n & B_{k+1}(\xi_{k+p-1})I_n & \cdots & B_{k+p}(\xi_{k+p-1})I_n \end{bmatrix}, \tag{2.7}$$

where $k=1+(i-1)(p-1)$ and I_n is the $n\times n$ identity matrix. The block rows of S_i correspond to the $p-1$ collocation points contained in $[x_{i-1},x_i)$. The block columns of S_i correspond to the $p+1$ B-spline basis functions that can be non-zero when evaluated at points contained in $[x_{i-1},x_i)$, as mentioned above.

BACOL employs a modification of DASSL [4], which uses a family of BDFs (of orders 1 to 5) for the time integration of the DAE system. An estimate of the local *temporal* error for each time step is controlled by DASSL so that it is less than a given user tolerance. Since the matrices that arise during the computation of the solution of the DAE systems have an ABD structure, DASSL was modified to use the software package, COLROW [10], which is appropriate for the efficient treatment of such matrices. Once the B-spline coefficients are obtained from the solution of the DAE system, the collocation solution is then computable from (2.2).

In addition to computing $U(x,t)$, BACOL also computes a high order estimate of the spatial error associated with $U(x,t)$. BACOL obtains this high order spatial error estimate by comparing $U(x,t)$ with $\overline{U}(x,t)$, a second global collocation solution it also computes,

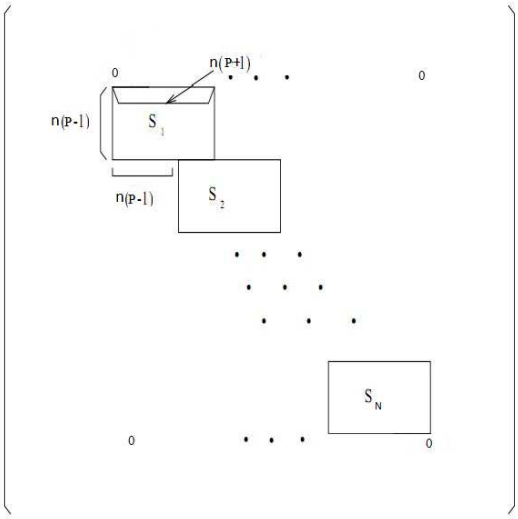

Figure 1: The structure of the matrix A_x that appears in (2.6). The zeros appearing in the top and bottom rows represent $n\times n$ blocks of zeros, where n is the number of PDEs. Each block, S_i (2.7), is an $n(p-1)$ by $n(p+1)$ matrix, where p is the degree of the piecewise polynomials representing the collocation solution. The overlap between the S_i blocks is $2n$. N is the number of subintervals of the spatial mesh.

using degree $p+1$ piecewise polynomials on the same spatial mesh that was used to obtain $U(x,t)$. If the estimated error does not satisfy the user provided tolerance, BACOL will generate a new spatial mesh by approximately equidistributing the estimated spatial error over each subinterval of the new mesh. The total number of mesh points may also be changed during this mesh adaptation process in order to improve the accuracy/efficiency of the next collocation solution computed on the new mesh.

3 The BACOL2D implementation

In this section we discuss the main algorithms employed by BACOL2D. First we describe the tensor product B-spline Gaussian collocation algorithm used to simultaneously discretize the x and y directions of (1.1). This collocation process yields a large system of DAEs for the B-spline coefficients. Next we describe how DASPK is used to solve this DAE system. Then we describe a fast block LU algorithm for the treatment of certain structured matrix systems that arise during the computation.

3.1 2D B-spline Gaussian collocation

We consider the case of a single PDE ($n=1$); however, the algorithm easily generalizes to arbitrary n. We assume a rectangular grid based on a mesh of $N+1$ points in $[a,b]$ and a mesh of $M+1$ points in $[c,d]$ such that $a=x_0<x_1<\cdots<x_N=b$ and $c=y_0<y_1<\cdots<y_M=d$. In the x direction, we employ C^1-continuous piecewise polynomials of degree

p $(3\leq p\leq 11)$, i.e., we have a polynomial of degree p on the ith subinterval, $[x_{i-1},x_i]$, $i=1,\cdots,N$, with C^1-continuity imposed at the internal mesh points. The dimension of this piecewise polynomial subspace is $NC=N(p-1)+2$. Similarly, in the y direction, we use C^1-continuous piecewise polynomials of degree q $(3\leq q\leq 11)$. The dimension of this piecewise polynomial subspace is $MC=M(q-1)+2$.

To represent the piecewise polynomials, we employ B-spline bases. Let $\{B_i(x)\}_{i=1}^{NC}$ and $\{D_i(y)\}_{i=1}^{MC}$ be the B-splines bases in the x and y directions. We then write the numerical solution, $U(x,y,t)$, as a linear combination of the tensor product of the B-spline basis functions with time-dependent coefficients, $w_{ij}(t)$:

$$U(x,y,t)=\sum_{i=1}^{NC}\sum_{j=1}^{MC}w_{ij}(t)B_i(x)D_j(y). \tag{3.1}$$

As in the 1D case, the B-spline basis for the x direction has the property that on the subinterval $[x_{i-1},x_i)$, at most $p+1$ basis functions, $\{B_r(x)\}_{r=(i-1)(p-1)+1}^{i(p-1)+2}$, are non-zero. Similarly, the B-spline basis for the y direction has the property that on any subinterval $[y_{j-1},y_j)$, at most $q+1$ basis functions, $\{D_s(x)\}_{s=(j-1)(q-1)+1}^{j(q-1)+2}$, are non-zero. This implies that, for $x\in[x_{i-1},x_i)$ and $y\in[y_{j-1},y_j)$, (3.1) can be rewritten as

$$U(x,y,t)=\sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r(x)D_s(y). \tag{3.2}$$

Let $h_i=x_i-x_{i-1}$, $k_i=y_i-y_{i-1}$ and $\{\rho_i\}_{i=1}^{p-1}$ and $\{\eta_i\}_{i=1}^{q-1}$ be the images of the Gaussian points on $[0,1]$, with $0<\rho_1<\cdots<\rho_{p-1}<1$, and $0<\eta_1<\cdots<\eta_{q-1}<1$. The collocation points in the x direction are defined to be $\xi_1=x_0=a$, $\xi_l=x_{i-1}+h_i\rho_j$, $l=1+(i-1)\cdot(p-1)+j$, $i=1,\cdots,N$, $j=1,\cdots,p-1$, $\xi_{NC}=x_N=b$, and the collocation points in the y direction are defined to be $\gamma_1=y_0=c$, $\gamma_l=y_{i-1}+k_i\eta_j$, $l=1+(i-1)\cdot(q-1)+j$, $i=1,\cdots,M$, $j=1,\cdots,q-1$, $\gamma_{MC}=y_M=d$.

The PDE is discretized in the x and y directions by collocating at the points $\{\xi_i\}_{i=2}^{NC-1}$ in x and the points $\{\gamma_j\}_{j=2}^{MC-1}$ in y. This gives the following ODEs in time:

$$\begin{aligned}U_t(\xi_i,\gamma_j,t)=&f\big(\xi_i,\gamma_j,t,U(\xi_i,\gamma_j,t),U_x(\xi_i,\gamma_j,t),U_y(\xi_i,\gamma_j,t),\\&U_{xx}(\xi_i,\gamma_j,t),U_{xy}(\xi_i,\gamma_j,t),U_{yy}(\xi_i,\gamma_j,t)\big),\end{aligned} \tag{3.3}$$

where $i=2,\cdots,NC-1$, and $j=2,\cdots,MC-1$. The collocation conditions involving ξ_1, ξ_{NC}, γ_1, γ_{MC} are associated with applying collocation to the boundary conditions. These conditions are

$$0=g_a(\gamma_j,t,U(\xi_1,\gamma_j,t),U_x(\xi_1,\gamma_j,t),U_y(\xi_1,\gamma_j,t)),\qquad j=1,\cdots,MC, \tag{3.4a}$$

$$0=g_b(\gamma_j,t,U(\xi_{NC},\gamma_j,t),U_x(\xi_{NC},\gamma_j,t),U_y(\xi_{NC},\gamma_j,t)),\qquad j=1,\cdots,MC, \tag{3.4b}$$

$$0=g_c(\xi_i,t,U(\xi_i,\gamma_1,t),U_x(\xi_i,\gamma_1,t),U_y(\xi_i,\gamma_1,t)),\qquad i=1,\cdots,NC, \tag{3.4c}$$

$$0=g_d(\xi_i,t,U(\xi_i,\gamma_{MC},t),U_x(\xi_i,\gamma_{MC},t),U_y(\xi_i,\gamma_{MC},t)),\qquad i=1,\cdots,NC. \tag{3.4d}$$

The above collocation conditions (3.3)-(3.4d) are collected together to form a DAE system. They are shown, in the order they appear within the DAE system, in (3.5)

$$\left[\begin{array}{c}
0=g_a(\gamma_1,t,U(\xi_1,\gamma_1,t),U_x(\xi_1,\gamma_1,t),U_y(\xi_1,\gamma_1,t)) \\
\vdots \\
0=g_a(\gamma_{MC},t,U(\xi_1,\gamma_{MC},t),U_x(\xi_1,\gamma_{MC},t),U_y(\xi_1,\gamma_{MC},t)) \\
\hline
0=g_c(\xi_2,t,U(\xi_2,\gamma_1,t),U_x(\xi_2,\gamma_1,t),U_y(\xi_2,\gamma_1,t)) \\
U_t(\xi_2,\gamma_2,t)=f\big(\xi_2,\gamma_2,t,U(\xi_2,\gamma_2,t),U_x(\xi_2,\gamma_2,t),U_y(\xi_2,\gamma_2,t), \\
\qquad U_{xx}(\xi_2,\gamma_2,t),U_{xy}(\xi_2,\gamma_2,t),U_{yy}(\xi_2,\gamma_2,t)\big) \\
\vdots \\
U_t(\xi_2,\gamma_{MC-1},t)=f\big(\xi_2,\gamma_{MC-1},t,U(\xi_2,\gamma_{MC-1},t),U_x(\xi_2,\gamma_{MC-1},t), \\
\qquad U_y(\xi_2,\gamma_{MC-1},t),U_{xx}(\xi_2,\gamma_{MC-1},t), \\
\qquad U_{xy}(\xi_2,\gamma_{MC-1},t),U_{yy}(\xi_2,\gamma_{MC-1},t)\big) \\
0=g_d(\xi_2,t,U(\xi_2,\gamma_{MC},t),U_x(\xi_2,\gamma_{MC},t),U_y(\xi_2,\gamma_{MC},t)) \\
\hline
\vdots \\
\hline
0=g_c(\xi_{NC-1},t,U(\xi_{NC-1},\gamma_1,t),U_x(\xi_{NC-1},\gamma_1,t),U_y(\xi_{NC-1},\gamma_1,t)) \\
U_t(\xi_{NC-1},\gamma_2,t)=f\big(\xi_{NC-1},\gamma_2,t,U(\xi_{NC-1},\gamma_2,t),U_x(\xi_{NC-1},\gamma_2,t),U_y(\xi_{NC-1},\gamma_2,t), \\
\qquad U_{xx}(\xi_{NC-1},\gamma_2,t),U_{xy}(\xi_{NC-1},\gamma_2,t),U_{yy}(\xi_{NC-1},\gamma_2,t)\big) \\
\vdots \\
U_t(\xi_{NC-1},\gamma_{MC-1},t)=f\big(\xi_{NC-1},\gamma_{MC-1},t,U(\xi_{NC-1},\gamma_{MC-1},t),U_x(\xi_{NC-1},\gamma_{MC-1},t), \\
\qquad U_y(\xi_{NC-1},\gamma_{MC-1},t),U_{xx}(\xi_{NC-1},\gamma_{MC-1},t), \\
\qquad U_{xy}(\xi_{NC-1},\gamma_{MC-1},t),U_{yy}(\xi_{NC-1},\gamma_{MC-1},t)\big) \\
0=g_d(\xi_{NC-1},t,U(\xi_{NC-1},\gamma_{MC},t),U_x(\xi_{NC-1},\gamma_{MC},t),U_y(\xi_{NC-1},\gamma_{MC},t)) \\
\hline
0=g_b(\gamma_1,t,U(\xi_{NC},\gamma_1,t),U_x(\xi_{NC},\gamma_1,t),U_y(\xi_{NC},\gamma_1,t)) \\
\vdots \\
0=g_b(\gamma_{MC},t,U(\xi_{NC},\gamma_{MC},t),U_x(\xi_{NC},\gamma_{MC},t),U_y(\xi_{NC},\gamma_{MC},t))
\end{array}\right]. \tag{3.5}$$

The DAE system begins with MC algebraic equations (3.4a) involving collocation of the boundary condition g_a, followed by $NC-2$ groups of equations. The lth group consists of collocation conditions involving ξ_l and includes MC equations: one algebraic equation from the set of Eqs. (3.4c) involving collocation of the boundary condition g_c at $x=\xi_l$, $MC-2$ ODEs (3.3) associated with collocation of the PDE at $x=\xi_l$, and one algebraic equation from the set of Eqs. (3.4d) involving collocation of the boundary condition g_d at $x=\xi_l$. The last set of MC equations appearing in the DAE system are the algebraic equations (3.4b) involving collocation of the boundary condition g_b.

By substituting (3.2) into (3.5), we can rewrite the DAE system in the terms of the unknowns $w_{ij}(t)$. The DAE system (3.5), in matrix form, becomes

$$A\underline{W}_t(t)=\underline{F}(t,\underline{W}(t)). \tag{3.6}$$

In (3.6), $\underline{W}(t)$ and $\underline{F}(t,\underline{W}(t))$ are each of size $NC\cdot MC$ and have the form,

$$\underline{W}(t)=\begin{bmatrix}\underline{w}_1(t)\\ \underline{w}_2(t)\\ \vdots\\ \underline{w}_{NC}(t)\end{bmatrix},\quad \text{where } \underline{w}_i(t)=\begin{bmatrix}w_{i1}(t)\\ w_{i2}(t)\\ \vdots\\ w_{iMC}(t)\end{bmatrix},\quad \underline{F}(t,\underline{W}(t))=\begin{bmatrix}F_1(t,\underline{W}(t))\\ \vdots\\ F_{NC}(t,\underline{W}(t))\end{bmatrix},$$

where each $\underline{F}_l(t,\underline{W}(t))$, $l=1,\cdots,NC$, has MC components. The vector $\underline{F}_1(t,\underline{W}(t))$ has the form

$$\begin{bmatrix}
g_a\big(\gamma_1,t,\sum_{r=1}^{p+1}\sum_{s=1}^{q+1}w_{rs}(t)B_r(\xi_1)D_s(\gamma_1),\\
\sum_{r=1}^{p+1}\sum_{s=1}^{q+1}w_{rs}(t)B_r'(\xi_1)D_s(\gamma_1),\\
\sum_{r=1}^{p+1}\sum_{s=1}^{q+1}w_{rs}(t)B_r(\xi_1)D_s'(\gamma_1)\big)\\
\vdots\\
g_a\big(\gamma_l,t,\sum_{r=1}^{p+1}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r(\xi_1)D_s(\gamma_l),\\
\sum_{r=1}^{p+1}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r'(\xi_1)D_s(\gamma_l),\\
\sum_{r=1}^{p+1}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r(\xi_1)D_s'(\gamma_l)\big)\\
\vdots\\
g_a\big(\gamma_{MC},t,\sum_{r=1}^{p+1}\sum_{s=(M-1)(q-1)+1}^{M(q-1)+2}w_{rs}(t)B_r(\xi_1)D_s(\gamma_{MC}),\\
\sum_{r=1}^{p+1}\sum_{s=(M-1)(q-1)+1}^{M(q-1)+2}w_{rs}(t)B_r'(\xi_1)D_s(\gamma_{MC}),\\
\sum_{r=1}^{p+1}\sum_{s=(M-1)(q-1)+1}^{M(q-1)+2}w_{rs}(t)B_r(\xi_1)D_s'(\gamma_{MC})\big)
\end{bmatrix},$$

where $l=2,\cdots,MC-1$ and $\gamma_l\in[y_{j-1},y_j)$ for some j. The vector $F_{NC}(t,\underline{W}(t))$ has the form

$$\begin{bmatrix}
g_b\big(\gamma_1,t,\sum_{r=(N-1)(p-1)+1}^{N(p-1)+2}\sum_{s=1}^{q+1}w_{rs}(t)B_r(\xi_{NC})D_s(\gamma_1),\\
\sum_{r=(N-1)(p-1)+1}^{N(p-1)+2}\sum_{s=1}^{q+1}w_{rs}(t)B_r'(\xi_{NC})D_s(\gamma_1),\\
\sum_{r=(N-1)(p-1)+1}^{N(p-1)+2}\sum_{s=1}^{q+1}w_{rs}(t)B_r(\xi_{NC})D_s'(\gamma_1)\big)\\
\vdots\\
g_b\big(\gamma_l,t,\sum_{r=(N-1)(p-1)+1}^{N(p-1)+2}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r(\xi_{NC})D_s(\gamma_l),\\
\sum_{r=(N-1)(p-1)+1}^{N(p-1)+2}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r'(\xi_{NC})D_s(\gamma_l),\\
\sum_{r=(N-1)(p-1)+1}^{N(p-1)+2}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r(\xi_{NC})D_s'(\gamma_l)\big)\\
\vdots\\
g_b\big(\gamma_{MC},t,\sum_{r=(N-1)(p-1)+1}^{N(p-1)+2}\sum_{s=(M-1)(q-1)+1}^{M(q-1)+2}w_{rs}(t)B_r(\xi_{NC})D_s(\gamma_{MC}),\\
\sum_{r=(N-1)(p-1)+1}^{N(p-1)+2}\sum_{s=(M-1)(q-1)+1}^{M(q-1)+2}w_{rs}(t)B_r'(\xi_{NC})D_s(\gamma_{MC}),\\
\sum_{r=(N-1)(p-1)+1}^{N(p-1)+2}\sum_{s=(M-1)(q-1)+1}^{M(q-1)+2}w_{rs}(t)B_r(\xi_{NC})D_s'(\gamma_{MC})\big)
\end{bmatrix},$$

where $l=2,\cdots,MC-1$ and $\gamma_l\in[y_{j-1},y_j)$ for some j. For $k=2,\cdots,NC-1$, and assuming $\xi_k\in[x_i,x_{i+1})$, we have

$$\underline{F}_k(t,\underline{W}(t))=\begin{bmatrix} g_c\big(\xi_k,t,\sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=1}^{q+1}w_{rs}(t)B_r(\xi_k)D_s(\gamma_1),\\ \sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=1}^{q+1}w_{rs}(t)B_r'(\xi_k)D_s(\gamma_1),\\ \sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=1}^{q+1}w_{rs}(t)B_r(\xi_k)D_s'(\gamma_1)\big)\\ \vdots\\ f\big(\xi_k,\gamma_l,t,\sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r(\xi_k)D_s(\gamma_l),\\ \sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r'(\xi_k)D_s(\gamma_l),\\ \sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r(\xi_k)D_s'(\gamma_l),\\ \sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r''(\xi_k)D_s(\gamma_l),\\ \sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r'(\xi_k)D_s'(\gamma_l),\\ \sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=(j-1)(q-1)+1}^{j(q-1)+2}w_{rs}(t)B_r(\xi_k)D_s''(\gamma_l)\big)\\ \vdots\\ g_d\big(\xi_k,t,\sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=(M-1)(q-1)+1}^{M(q-1)+2}w_{rs}(t)B_r(\xi_k)D_s(\gamma_{MC}),\\ \sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=(M-1)(q-1)+1}^{M(q-1)+2}w_{rs}(t)B_r'(\xi_k)D_s(\gamma_{MC}),\\ \sum_{r=(i-1)(p-1)+1}^{i(p-1)+2}\sum_{s=(M-1)(q-1)+1}^{M(q-1)+2}w_{rs}(t)B_r(\xi_k)D_s'(\gamma_{MC})\big)\end{bmatrix},$$

where $l=2,\cdots,MC-1$, and $\gamma_l\in[y_j,y_{j+1})$ for some j. The matrix A appearing in (3.6) has the form

$$A=A_x\otimes A_y, \tag{3.7}$$

where $\otimes$ is the Kronecker product, A_x is a matrix, associated with B-spline collocation in the x direction, similar to that given in Fig. 1, but with the following differences. Since we are assuming $n=1$, the top and bottom rows of zeros of A_x are simply rows of zeros rather than block rows of zeros, and each matrix, S_i, is a $(p-1)\times(p+1)$ matrix (thus in (2.7), I_n is replaced by 1). The matrix A_y appearing in (3.7) is the corresponding matrix associated with B-spline collocation in the y direction–see Fig. 2. In Fig. 2, the ith subblock, R_i, $i=1,\cdots,M$, is a $(q-1)\times(q+1)$ matrix given by

$$R_i=\begin{bmatrix} D_k(\gamma_{k+1}) & D_{k+1}(\gamma_{k+1}) & \cdots & D_{k+q}(\gamma_{k+1})\\ D_k(\gamma_{k+2}) & D_{k+1}(\gamma_{k+2}) & \cdots & D_{k+q}(\gamma_{k+2})\\ \vdots & \vdots & \ddots & \vdots\\ D_k(\gamma_{k+q-1}) & D_{k+1}(\gamma_{k+q-1}) & \cdots & D_{k+q}(\gamma_{k+q-1})\end{bmatrix}, \tag{3.8}$$

where $k=1+(i-1)(q-1)$. The rows of R_i correspond to the $q-1$ collocation points contained in $[y_{i-1},y_i)$ and the columns of R_i correspond to the $q+1$ B-spline basis functions that can be non-zero when evaluated at collocation points contained in $[y_{i-1},y_i)$.

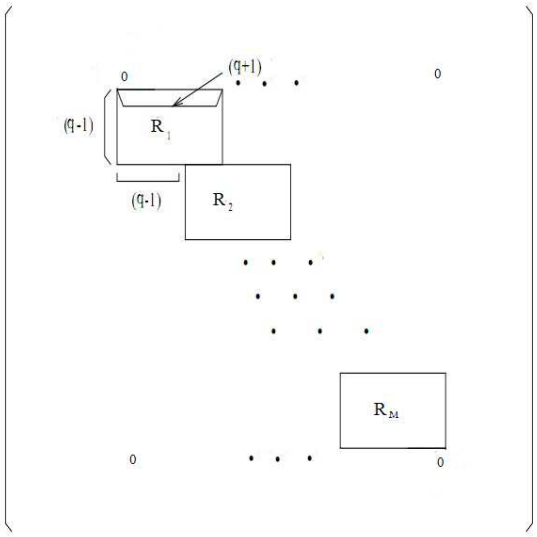

Figure 2: The structure of the matrix A_y associated with B-spine collocation in the y direction. Here we assume the case $n=1$ (where n is the number of PDEs). The matrix has a row of zeros along the top and the bottom. Each block, R_j (3.8), is of size $(q-1)\times(q+1)$, where q is the degree of the piecewise polynomials used in the y direction. The overlap between the R_j blocks is 2. M is the number of subintervals in the y direction.

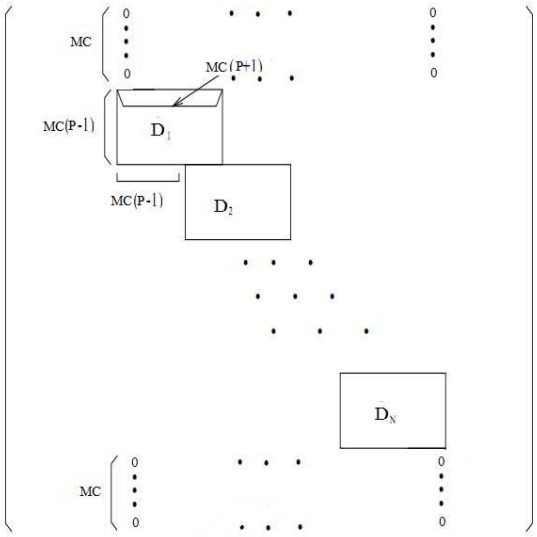

Figure 3: The structure of the matrix A appearing in (3.6). Each submatrix, D_i (3.9), is a block matrix having $(p-1)$ block rows and $(p+1)$ block columns where each internal block of D_i is a matrix of size $MC\times MC$. Thus D_i is a matrix of size $MC(p-1)\times MC(p+1)$. The overlap between the D_i blocks is $2MC$, p is the degree of the piecewise polynomials used in the x direction, N is the number of subintervals in the x direction, MC is the dimension of the piecewise polynomial subspace in the y direction.

The structure of the matrix A, appearing in (3.6), is shown in Fig. 3. Each matrix, D_i, is a block matrix having $(p-1)$ block rows and $(p+1)$ block columns where each internal block of D_i is a matrix of size $MC \times MC$. The matrix D_i has the form

$$D_i = \begin{bmatrix} B_k(\xi_{k+1})A_y & B_{k+1}(\xi_{k+1})A_y & \cdots & B_{k+p}(\xi_{k+1})A_y \\ B_k(\xi_{k+2})A_y & B_{k+1}(\xi_{k+2})A_y & \cdots & B_{k+p}(\xi_{k+2})A_y \\ \vdots & \vdots & \ddots & \vdots \\ B_k(\xi_{k+p-1})A_y & B_{k+1}(\xi_{k+p-1})A_y & \cdots & B_{k+p}(\xi_{k+p-1})A_y \end{bmatrix}, \tag{3.9}$$

where $k=1+(i-1)(p-1)$.

The DAE system (3.6) must be solved by DASPK in order to obtain the B-spline coefficients, $\underline{W}(t)$. During the computation of the solution of the DAE system, it is obviously very important to take advantage of the matrix structure present in the DAE system. We discuss this further in Subsection 3.3

3.2 The numerical solution of the DAE system using DASPK

A major task in the development of the BACOL2D software involved interfacing the 2D B-spline collocation algorithm that generates the large DAE system (3.6) with the DAE solver, DASPK, that is used to solve the DAE system. DASPK was obtained from a modification of DASSL. DASPK also uses a family of BDFs of orders 1 to 5 to solve the DAE system. This computation involves the solution of a nonlinear system; DASPK uses an inexact Newton method [9] to treat the nonlinear system and a large linear system arises during each Newton step. Since these linear systems are too large to be solved with a direct method, an iterative method must be used; DASPK uses (a Krylov subspace method) the scaled preconditioned incomplete GMRES method [26]. DASPK employs a banded preconditioner matrix and tries to reuse this preconditioner for as many time steps as possible, since the costs for building this preconditioner are high. The linear system involving the preconditioner matrix is solved using an incomplete LU factorization [27] based on routines from the SPARSKIT library–see [27] and references within.

3.3 A efficient block LU algorithm for the 2D B-spline projection matrix

From the discussion of Subsection 3.1, it is clear that the matrices that arise during the computations have substantial structure. The efficient treatment of these matrices using algorithms that take advantage of this structure is central to the efficiency of the BACOL2D software. Here we describe a block LU algorithm associated with computing projections of the collocation solution onto the 2D B-spline basis. The M.SC. thesis [21] provides further details on these computations.

At certain points in the BACOL2D algorithm we need to project the collocation solution, evaluated at all combinations of the collocation points, $U(\xi_i,\gamma_j,t)$, $i=1,\cdots,NC$, $j=1,\cdots,MC$, onto the 2D B-spline basis. Let $\underline{U}(\underline{\xi},\underline{\gamma},t)$ be the vector representing the

evaluation of the collocation solution at all combinations of the collocation points; the lth components of the vectors $\underline{\xi}$ and $\underline{\gamma}$ are, respectively, ξ_l and γ_l. The determination of the projection of these collocation solution values onto the 2D B-spline basis can then be expressed as computing the solution of the linear system

$$(M_x \otimes M_y)\underline{W}(t) = \underline{U}(\underline{\xi},\underline{\gamma},t), \tag{3.10}$$

where M_y is the same as the matrix A_y appearing in Fig. 2 except that the zeros in the upper left hand corner and lower right hand corner are replaced by ones, and M_x is the corresponding matrix for the x direction. (Note that M_x and M_y are ABD matrices of dimensions $NC \cdot NC$ and $MC \cdot MC$, respectively.)

We assume that we have factored $M_x = L_x U_x$ and $M_y = L_y U_y$ (The factorization performed by COLROW also includes permutation matrices associated with alternating row and column pivoting and is based on row and column elimination but in order to simplify the presentation we do not include these components of the factorization here. Slight modifications to some of the COLROW routines were required in order to implement the following algorithm). Eq. (3.10) can be rewritten as

$$((L_x U_x) \otimes (L_y U_y))\underline{W}(t) = \underline{U}(\underline{\xi},\underline{\gamma},t), \tag{3.11}$$

and, employing a property of the Kronecker product, we can rewrite (3.11) in the form

$$((L_x \otimes L_y)(U_x \otimes U_y))\underline{W}(t) = \underline{U}(\underline{\xi},\underline{\gamma},t). \tag{3.12}$$

The linear system (3.12) can be solved using the following four step fast block LU algorithm:

1. Solve $(L_x \otimes I_{MC})\underline{\bar{V}}(t) = \underline{U}(\underline{\xi},\underline{\gamma},t)$ for $\underline{\bar{V}}(t)$
   (where I_{MC} is the identity matrix of dimension $MC \times MC$).
2. Then solve $(I_{NC} \otimes L_y)\underline{V}(t) = \underline{\bar{V}}(t)$ for $\underline{V}(t)$
   (where I_{NC} is the identity matrix of dimension $NC \times NC$).
3. Then solve $(U_x \otimes I_{MC})\underline{\bar{W}}(t) = \underline{V}(t)$ for $\underline{\bar{W}}(t)$.
4. Then solve $(I_{NC} \otimes U_y)\underline{W}(t) = \underline{\bar{W}}(t)$ for $\underline{W}(t)$.

Steps 1 and 2 solve the system $(L_x \otimes L_y)\underline{V}(t) = \underline{U}(\underline{\xi},\underline{\gamma},t)$ for $\underline{V}(t)$. Steps 3 and 4 solve the system $(U_x \otimes U_y)\underline{W}(t) = \underline{V}(t)$ for $\underline{W}(t)$. This algorithm implies that instead of factoring and solving a large $(MC \cdot NC) \times (MC \cdot NC)$ linear system (3.10), we only need to factor one matrix of size $NC \times NC$ and one matrix of size $MC \times MC$ and then solve a sequence of $MC \times MC$ and $NC \times NC$ triangular linear systems.

For the case where $N = M$ and $p = q$ (which implies $NC = MC$), it can be shown that the direct treatment of (3.10) has a cost that is $\mathcal{O}(N^4 p^6)$. The fast block LU algorithm first requires the factorization of the two ABD matrices, M_x and M_y, and these factorizations have costs that are $\mathcal{O}(Np^3)$. For each step of the fast block LU algorithm it can be shown that the cost is $\mathcal{O}(N^2 p^3)$. Therefore the overall cost for the fast block LU algorithm is

$\mathcal{O}(N^2p^3)$. Thus, even for modestly sized values of N or p, this algorithm can provide substantial efficiency improvements over the standard approach.

Similar algorithms have been discussed in a number of previous papers–see, e.g., [24, 25].

4 Numerical experiments

In this section, we will consider three 2D parabolic PDE test problems. We use GNU Fortran77 (GCC) 4.4.3 under ubuntu (Kernel Linux 2.6.32-40-server) running on an 7 Intel(R) Xeon(R) CPUs(E5420 @ 2.50GHz) system for which the accessible memory is 2GB.

The following notation will be used in representing the numerical results:

- $KCOL$: the number of collocation points per subinterval, $KCOL=p-1=q-1$;
- NINT: the number of subintervals, NINT$=N=M$;
- ATOL: the absolute tolerance (used by DASPK);
- RTOL: the relative tolerance (used by DASPK);
- TOL: the tolerance for the nonlinear solver in DASPK;
- t_{out}: the output time;
- GE: the true error at a set of sample points equally distributed over the problem domain at time t_{out}.

In order to obtain the convergence results, we employed a number of different choices for TOL, ATOL, and RTOL. These were chosen by experimentally so that the temporal error was smaller than the observed spatial error for each collocation solution we computed.

4.1 Numerical solution of three test problems

Problem 4.1. The 2D Burgers' equation [29],

$$\frac{\partial u}{\partial t}=\epsilon\frac{\partial^2 u}{\partial x^2}+\epsilon\frac{\partial^2 u}{\partial y^2}-u\frac{\partial u}{\partial x}-u\frac{\partial u}{\partial y}. \tag{4.1}$$

The problem domain is $(x,y)\in(0,1)\times(0,1)$, $t>0$; the boundary and initial conditions are chosen so that the true solution is

$$u(x,y,t)=\left(1+e^{\frac{x+y-t}{2\epsilon}}\right)^{-1}.$$

We set $\epsilon=0.01$ and $t_{out}=1$. The numerical solution is plotted in Fig. 4 for the case $KCOL=3$, NINT = 64, RTOL = ATOL = 10^{-6}, TOL = 10^{-7}.

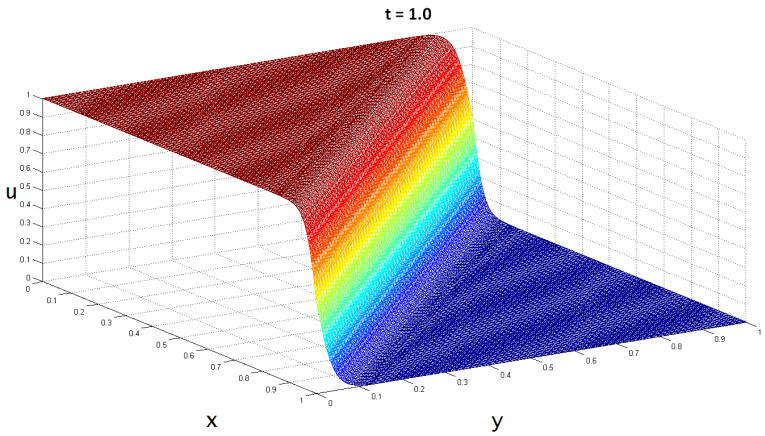

Figure 4: Approximate solution for Problem 4.1, $\epsilon = 0.01$.

Problem 4.2. (see [33])

$$\frac{\partial u}{\partial t} = (L_1 + L_2)u + f(x,y,t), \quad L_1 = (x^2+1)\frac{\partial^2}{\partial x^2} + x, \quad L_2 = (y^2+1)\frac{\partial^2}{\partial y^2} + y\frac{\partial}{\partial y} + y,$$

on the spatial domain, $(0,1)\times(0,1)$, with the boundary and initial conditions (at $t=0$) and $f(x,y,t)$ chosen so that the true solution is

$$u = (e^{-t}+1)\sin(\pi x)\sin(\pi y).$$

We choose $t_{out}=1$. The numerical solution is plotted in Fig. 5 for the case $KCOL=3$, NINT = 16, RTOL = ATOL = 2×10^{-13}, TOL = 10^{-14}.

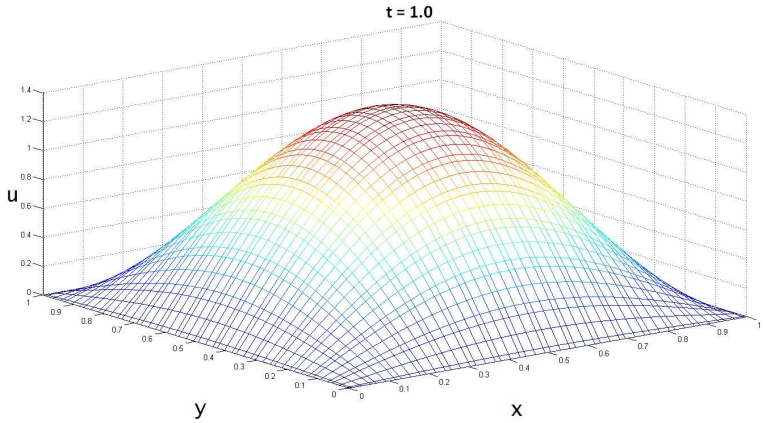

Figure 5: Approximate solution for Problem 4.2.

Problem 4.3. (see [33])

$$\frac{\partial u}{\partial t}=\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+f(x,y,t),$$

on the spatial domain $(0,1)\times(0,1)$. The function $f(x,y,t)$ and the boundary and initial conditions (at $t=0$) are chosen so that the true solution is

$$u(x,y,t)=(e^{-t}+1)(x^m+y^m+xy^{m-1}+1).$$

We set $m=6$ and $t_{out}=1$. The numerical solution is plotted in Fig. 6 for the case $KCOL=5$, NINT = 20, RTOL = ATOL = 5×10^{-13}, TOL = 5×10^{-14}.

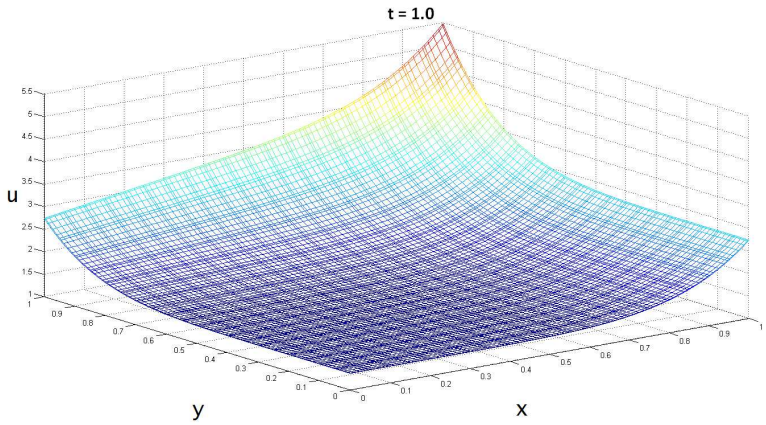

Figure 6: Approximate solution for Problem 4.3, $m=6$.

4.2 Order of convergence of the 2D collocation solution

Since each of the three test problems has a known solution, it is possible to estimate the maximum GE of a given numerical solution. In this subsection, we compute an estimate of the maximum GE for the collocation solutions computed by BACOL2D. By fixing $KCOL$ and considering a sequence of meshes obtained by doubling the NINT value we can compare the observed GE. By considering the ratio of the GE of the collocation solutions obtained using this sequence of meshes, we can experimentally determine the spatial order of convergence of the collocation solution.

The convergence results for the corresponding 1D case are known from the literature–see, e.g., [7]. The rate of convergence is $KCOL+2$, i.e., the spatial error is $\mathcal{O}(h^{KCOL+2})$ where h is the maximum spatial mesh spacing. Since we are using a tensor product framework, we anticipate that the corresponding result will hold for the 2D case.

In Tables 1-3, we present the observed GE, GE ratios, and corresponding approximate convergence rates for Problems 4.1-4.3, for several $KCOL$ and NINT values.

Table 1: Observed GE, GE ratios, and corresponding approximate convergence rates for Problem 4.1.

KCOL	NINT	GE	ratio	rate
3	8	9.95×10^{-2}	-	-
3	16	3.69×10^{-3}	26.96	4.75
3	32	1.16×10^{-4}	31.81	4.99
3	64	4.22×10^{-6}	27.49	4.78
4	16	5.56×10^{-4}	-	-
4	32	1.05×10^{-5}	52.95	5.73
4	64	1.90×10^{-7}	55.33	5.79
5	10	2.40×10^{-3}	-	-
5	20	2.06×10^{-5}	116.50	6.86
5	40	1.61×10^{-6}	130.38	7.03

Table 2: Observed GE, GE ratios, and corresponding approximate convergence rates for Problem 4.2.

KCOL	NINT	GE	ratio	rate
3	4	1.50×10^{-5}	-	-
3	8	4.51×10^{-7}	33.22	5.05
3	16	1.36×10^{-8}	33.23	5.05
3	32	4.14×10^{-10}	32.78	5.03
4	4	5.23×10^{-7}	-	-
4	8	8.58×10^{-9}	60.88	5.93
4	16	1.36×10^{-10}	63.26	5.98
4	32	2.13×10^{-12}	63.63	5.99
5	10	3.06×10^{-11}	-	-
5	20	1.97×10^{-13}	155.33	7.28

Table 3: Observed GE, GE ratios, and corresponding approximate convergence rates for Problem 4.3.

KCOL	NINT	GE	ratio	rate
3	32	1.99×10^{-9}	-	-
3	64	6.38×10^{-11}	31.19	4.96
4	20	1.67×10^{-10}	-	-
4	40	2.98×10^{-12}	56.04	5.81
5	15	1.71×10^{-11}	-	-
5	30	1.33×10^{-13}	128.57	7.01

From the three tables, we observe that the expected rates of convergence (based on the known results for the 1D case) are indeed observed in the 2D case. The GE is order $KCOL+2$; i.e., the rate of convergence is $\mathcal{O}(h^{KCOL+2})$ where $KCOL$ is the (equal) number of collection points per subinterval in the x and y directions and h is the (uniform and equal) mesh subinterval size in the x and y directions.

5 Summary and future work

In this paper, we have described an extension of the 1D *B*-spline Gaussian collocation algorithm employed within the 1D PDE solver BACOL to a 2D *B*-spline Gaussian collocation algorithm for use in the 2D PDE solver, BACOL2D. The 2D collocation algorithm employs a tensor product *B*-spline basis, based on the corresponding 1D *B*-spline bases in x and y. The application of this collocation algorithm leads to a large DAE system. A major component of our work involved applying the DAE solver DASPK so that it could efficiently solve this large DAE system. It was necessary to take advantage of the structure present in the DAE system, arising from the use of a tensor product basis, in order to obtain an efficient implementation. Numerical results were provided to demonstrate the use of BACOL2D on three test problems. We also provided experimental convergence results that showed that the 2D collocation solution had the same convergence rate as is observed in the 1D case.

There are two important extensions of BACOL2D that are required. The first is that an efficient error estimate for the 2D collocation solution must be developed. Because of the use of the tensor product basis, it appears likely that the efficient interpolation based spatial error estimation schemes recently developed for BACOL in the 1D case [1, 2] can be extended to the 2D case. The second important extension of BACOL2D will involve the introduction of an adaptive mesh refinement algorithm applied after each successful time step, so that the spatial error estimate is less than the user tolerance. We plan to investigate an approach employed in the area of MMM methods; spatial adaptivity is based on a transformation (via a MMPDE) between the physical domain on which the PDE is defined and a computational domain where the collocation and time integration computations are performed [18].

Acknowledgments

The authors would like to thank Jack Pew for several helpful discussions during the writing of this paper.

References

[1] T. ARSENAULT, T. SMITH AND P.H. MUIR, *Superconvergent interpolants for efficient spatial error estimation in 1D PDE collocation solvers*, Can. Appl. Math. Q., 17 (2009), pp. 409–431.

[2] T. ARSENAULT, T. SMITH, P. H. MUIR AND J. PEW, *Asymptotically exact interpolation-based error estimates for collocation solutions of 1D PDEs*, to appear, Can. Appl. Math. Q., 2012.

[3] C. DE BOOR, *A practical guide to splines*, Applied Mathematical Sciences, 27, Springer-Verlag, New York, 1978.

[4] K. E. BRENAN, S. L. CAMPBELL AND L. R. PETZOLD, *Numerical solution of initial-value problems in differential-algebraic equations*, Society for Industrial and Applied Mathematics, Philadelphia, 1989.

[5] P. N. BROWN, A. C. HINDMARSH AND L. R. PETZOLD, *Using Krylov methods in the solution of large-scale differential-algebraic systems*, SIAM J. Sci. Comput., 15 (1994), pp. 1467–1488.
[6] W. CAO, W. HUANG AND R. D. RUSSELL, *A study of monitor functions for two-dimensional adaptive mesh generation*, SIAM J. Sci. Comput., 20 (1999), pp. 1978–1994.
[7] J. H. CERUTTI AND S. V. PARTER, *Collocation methods for parabolic partial differential equations in one space dimension*, Numer. Math., 26 (1976), pp. 227–254.
[8] C. C. CHRISTARA, *Quadratic spline collocation methods for elliptic partial differential equations*, BIT, 34 (1994), pp. 33–61.
[9] R. S. DEMBO, S. C. EISENSTAT AND T. STEIHAUG, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.
[10] J. C. DÍAZ, G. FAIRWEATHER AND P. KEAST, *FORTRAN packages for solving certain almost block diagonal linear systems by modified alternate row and column elimination*, ACM Trans. Math. Software, 9 (1983), pp. 358–375.
[11] M. S. GOCKENBACH, *Partial differential equations: analytical and numerical methods*, Society for Industrial and Applied Mathematics, Philadelphia, 2002.
[12] E. N. HOUSTIS, E. A. VAVALIS AND J. R. RICE, *Convergence of $\mathcal{O}(h^4)$ cubic spline collocation methods for elliptic partial differential equations*, SIAM J. Numer. Anal., 25 (1988), pp. 54–74.
[13] E. N. HOUSTIS, W. F. MITCHELL AND J. R. RICE, *Algorithm 637: GENCOL: collocation of general domains with bicubic hermite polynomials*, ACM Trans. Math. Software, 11 (1985), pp. 413–415.
[14] E. N. HOUSTIS, W. F. MITCHELL AND J. R. RICE, *Algorithm 638: INTCOL and HERMCOL: collocation on rectangular domains with bicubic hermite polynomials*, ACM Trans. Math. Software, 11 (1985), pp. 416–418.
[15] W. HUANG AND D. M. SLOAN, *A simple adaptive grid method in two dimensions*, SIAM J. Sci. Comput., 15 (1994), pp. 776–797.
[16] W. HUANG, *Practical aspects of formulation and solution of moving mesh partial differential equations*, J. Comput. Phys., 171 (2001), pp. 753–775.
[17] W. HUANG AND R. D. RUSSELL, *Moving mesh strategy based on a gradient flow equation for two-dimensional problems*, SIAM J. Sci. Comput., 20 (1998), pp. 998–1015.
[18] W. HUANG AND R. D. RUSSELL, *Adaptive moving mesh methods*, Applied Mathematical Sciences, 174, Springer, New York, 2011.
[19] W. HUNDSDORFER AND J. VERWER, *Numerical solution of time-dependent advection-diffusion-reaction equations*, Springer Series in Computational Mathematics, 33, Springer-Verlag, Berlin, 2003.
[20] R. J. LEVEQUE, *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*, Classics in Applied Mathematics, Society for Industrial and Applied Mathematics, Philadelphia, 2007.
[21] Z. LI, *B*-spline Collocation Methods for Two-Dimensional, Time-Dependent PDEs, M.Sc. thesis, Saint Mary's University, 2012.
[22] R. M. M. MATTHEIJ, S. W. RIENSTRA AND J. H. M. T. T. BOONKKAMP, *Partial differential equations: modeling, analysis, computation*, SIAM Monographs on Mathematical Modeling and Computation, Society for Industrial and Applied Mathematics, Philadelphia, 2005.
[23] K. S. NG, Spline Collocation on Adaptive Grids and Non-Rectangular Regions, Ph.D. thesis, University of Toronto, 2005.
[24] A. K. PANI, G. FAIRWEATHER AND R. I. FERNANDES, *ADI orthogonal spline collocation methods for parabolic partial integro-differential equations*, IMA J. Numer. Anal., 30 (2010), pp. 248–276.

[25] R. D. RUSSELL AND W. SUN, *Spline collocation differentiation matrices*, SIAM J. Numer. Anal., 34 (1997), pp. 2274–2287.
[26] Y. SAAD, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput., 14 (1993), pp. 461–469.
[27] Y. SAAD, *Iterative methods for sparse linear systems*, Society for Industrial and Applied Mathematics, Philadelphia, 2003.
[28] W. SUN, *B-spline collocation methods for elasticity problems*, in Scientific Computing and Applications, Adv. Comput. Theory Pract., 7 (2000), pp. 133–141.
[29] A. C. VELIVELLI AND K. M. BRYDEN, *Parallel performance and accuracy of lattice Boltzmann and traditional finite difference methods for solving the unsteady two-dimensional Burger's equation*, Physica A: Statistical Mechanics and its Applications, 362 (2006), pp. 139–145.
[30] R. WANG, P. KEAST AND P. H. MUIR, *BACOL: B-spline adaptive collocation software for 1-D parabolic PDEs*, ACM Trans. Math. Software, 30 (2004), pp. 454–470.
[31] R. WANG, P. KEAST AND P. H. MUIR, *A high-order global spatially adaptive collocation method for 1-D parabolic PDEs*, Appl. Numer. Math., 50 (2004), pp. 239–260.
[32] R. WANG, P. KEAST AND P. H. MUIR, *A comparison of adaptive software for 1D parabolic PDEs*, J. Comput. Appl. Math., 169 (2004), pp. 127–150.
[33] Y. WANG, A Parallel Collocation Method for Two Dimensional Linear Parabolic Separable Partial Differential Equations, Ph.D. thesis, Dalhousie University, 1995.
[34] S. WENDEL, H. MAISCH, H. KARL AND G. LEHNER, *Two-dimensional B-spline finite elements and their application to the computation of solitons*, Electrical Engineering (Archiv fur Elektrotechnik), 76 (1993), pp. 427–435.