A parallel Crank–Nicolson predictor-corrector method for many subdomains

Felix Kwok[1]

1 Introduction

In this paper, we propose a fast parallel solver for the parabolic equation

$$\begin{aligned} \partial_t u &= \mathscr{L}u + g(x,t), \qquad x \in \Omega, \\ u = u_\Gamma(x,t) \quad &\text{on } \Gamma = \partial\Omega, \qquad u(x,0) = f(x) \quad \text{on } \Omega, \end{aligned} \tag{1}$$

where Ω is an open connected subset of $\mathbb{R}^2$ and $\mathscr{L}u = \sum_{i,j} \partial_{x_i}\left(\kappa_{ij}(x)\partial_{x_j}u\right) - c(x)u$, with $c(x) \geq 0$ and $\kappa_{ij}(x)$ symmetric and uniformly positive definite, i.e., we have $\kappa_{ij}(x) = \kappa_{ji}(x)$ for $i \neq j$ and $\sum_{i,j} \kappa_{ij}(x)\xi_i\xi_j \geq \lambda \sum_i \xi_i^2$ for all choices of ξ_i, where the constant $\lambda > 0$ is independent of x. Our method is based on the predictor-corrector method introduced by [15]. In that work, the authors consider nonlinear reaction-diffusion equations posed on branched structures, which model the evolution of the electric potential in neurons, see Fig. 1. In such problems, the nodal points are natural separators of the computational domain, meaning that the solution within the individual branches can be solved independently if the electric potential at the nodes are known. Based on this observation, the authors proposed the Crank–Nicolson predictor-corrector (CNPC) method: they first use forward Euler to predict the nodal values, and then backward Euler to solve for the solution within the branches. To maintain stability, they then correct the nodal values using a backward Euler step, and the whole solution is extrapolated to obtain formal second-order accuracy in time. The main advantage of this method is that a fixed amount of computation is performed at each time step, and no iteration is necessary. This is unlike classical domain decomposition (DD) algorithms such as Schwarz methods [3, 14, 11, 1] or waveform relaxation methods [10, 8, 9, 7], where one must iterate to convergence (or to some fixed tolerance), and the number of iterations generally increases as the grid is refined. Thus, a suitable extension of the CNPC method for 2D and 3D problems can be useful for parallel-in-time methods such as Parareal [13, 6], where fast coarse integrators are needed. Other DD-type methods with a fixed cost per time step have been proposed in [4] and [16]; both are only first order accurate under simultaneous refinement in space and time.

Our main goal is to present in detail a generalization of the CNPC method that can be used to solve 2D problems with many subdomains in parallel. This is done in Section 2. In particular, we show how the backward Euler correction step for the interface can be implemented efficiently, even in cases where the subdomain inter-

[1] Université de Genève, 2-4 Rue du Lièvre, 1211 Genève, Switzerland, e-mail: felix.kwok@unige.ch

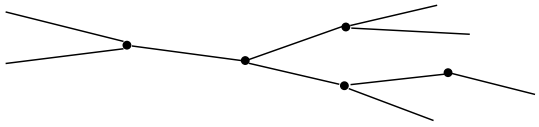

Fig. 1 A branched structure, with nodes indicated.

faces are coupled through cross points. To fix ideas, we have chosen a finite volume discretization in space, although similar techniques can be used for other discretizations. In Section 3, we examine the convergence of the CNPC method. We will see that the method indeed converges as the mesh size $h \to 0$ the time step τ satisfies $\tau = O(h^\alpha)$ for $\alpha \geq 1$. In fact, the method attains full second order accuracy for $\alpha \geq 3/2$; it is however only first order accurate when $\tau = O(h)$. Finally, numerical results in Section 4 illustrate the behavior of the method for many subdomains.

2 The CNPC algorithm

To define the CNPC algorithm, we will assume that the domain Ω is divided into shape regular, quasi-uniform and conforming control volumes V_i, $i = 1,\dots,n$, with diameter $h_i \leq h$, see Fig. 2. If we discretize (1) in space using a finite volume method, we get a semi-discrete ODE system of the form

$$M\partial_t \mathbf{u}(t) + A\mathbf{u}(t) + B\mathbf{u}_\Gamma(t) = Mg(\cdot,t). \tag{2}$$

Here, $\mathbf{u}(t)$ are the unknown values at the nodal points at time t, $A \in \mathbb{R}^{n\times n}$ is a sparse, symmetric positive definite matrix whose entries a_{ij} are non-zero and of $O(1)$ (constant with respect to h) if and only if volumes i and j are neighbors. $B \in \mathbb{R}^{n\times n_\Gamma}$ contains the dependence on the Dirichlet boundary values; its entries are also $O(1)$. $\mathbf{u}_\Gamma(t) \in \mathbb{R}^{n_\Gamma}$ contains the Dirichlet boundary values at time t. M is a diagonal mass matrix whose (i,i) entry is the area of V_i; thus, the elements of M are of size $O(h^2)$. $g(\cdot,t)$ is a vector whose elements are the values of g at the nodes; we will use this dot notation to denote the vectors of samples of other functions elsewhere in this paper.

We now divide the unknowns into two subsets, the *interface unknowns* $\mathcal{V}_1$ and the *interior unknowns* $\mathcal{V}_2$. We also define two corresponding projectors $X_1, X_2 \in \mathbb{R}^{n\times n}$ such that $X_1\mathbf{u}$ projects onto $\mathcal{V}_1$, i.e., it leaves all the values in $\mathcal{V}_1$ unchanged and sets all the other entries to zero, and X_2 does the opposite. Thus, we have $X_2 = I - X_1$ and $X_1X_2 = X_2X_1 = 0$. Note that X_1 and X_2 commute with M, since the latter is diagonal.

We are now ready to define the CNPC algorithm. For a given time-step size τ and an approximation $\mathbf{u}^n \approx u(\cdot,t_n)$, one step of the CNPC method proceeds as follows:

1. Predict the interface values at $t = t_{n+1/2}$ using forward Euler: calculate $\mathbf{u}^*$ using

$$\frac{M(\mathbf{u}^* - \mathbf{u}^n)}{\tau/2} = -X_1(A\mathbf{u}^n + B\mathbf{u}_\Gamma(t_n)) + X_1 Mg(\cdot,t_{n+1/2}),$$

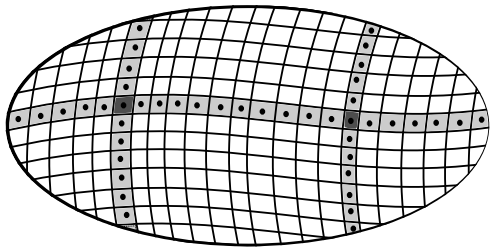

Fig. 2 Decomposition into interface (light and dark gray) and interior (white) cells and their corresponding unknowns. Light gray corresponds to edge nodes and dark gray to cross points.

Note that $X_2(\mathbf{u}^* - \mathbf{u}^n) = 0$, so interior node values are not altered by this step.

2. Using the predicted values $X_1\mathbf{u}^*$ as boundary values, solve for $\mathbf{u}^{**}$ in

$$\frac{M(\mathbf{u}^{**} - \mathbf{u}^n)}{\tau/2} = -X_2\left[A(X_1\mathbf{u}^* + X_2\mathbf{u}^{**}) + B\Big(\frac{\mathbf{u}_\Gamma(t_n) + \mathbf{u}_\Gamma(t_{n+1})}{2}\Big)\right] + X_2 Mg(\cdot, t_{n+1/2}),$$

where both $\mathbf{u}_\Gamma(t_n)$ and $\mathbf{u}_\Gamma(t_{n+1})$ are known. This corresponds to a backward Euler step for the interior unknowns $\mathcal{V}_2$; the interface values are not updated. Note that this step requires solving a linear system with the matrix $M + \frac{\tau}{2}X_2AX_2$.

3. Compute $\mathbf{u}^{n+1/2}$ by correcting the interface values at $t = t_{n+1/2}$ with backward Euler, using $\mathbf{u}^{**}$ as boundary values:

$$\frac{M(\mathbf{u}^{n+1/2} - \mathbf{u}^{**})}{\tau/2} = -X_1\Big[A(X_1\mathbf{u}^{n+1/2} + X_2\mathbf{u}^{**}) + B\Big(\frac{\mathbf{u}_\Gamma(t_n) + \mathbf{u}_\Gamma(t_{n+1})}{2}\Big)\Big] + X_1 Mg(\cdot, t_{n+1/2}).$$

This is a backward Euler step for the interface nodes, since their values have not been updated in the previous steps, i.e., we have $X_1\mathbf{u}^{**} = X_1\mathbf{u}^n$. For the other nodes, we have $X_2\mathbf{u}^{n+1/2} = X_2\mathbf{u}^{**}$, i.e. we reproduce the values obtained in step 2. Here one needs to solve a linear system with matrix $M + \frac{\tau}{2}X_1AX_1$.

4. Extrapolate to obtain $\mathbf{u}^{n+1}$:

$$\mathbf{u}^{n+1} = 2\mathbf{u}^{n+1/2} - \mathbf{u}^n.$$

Note that there is no iteration to convergence, since each of the above is only performed once per time step.

Parallelization. We only need consider how to solve linear systems with matrices $A_i = M + \frac{\tau}{2}X_iAX_i$ $(i = 1, 2)$ in parallel, since the other operators are local in nature and easy to parallelize. For the matrix $A_2 = M + \frac{\tau}{2}X_2AX_2$ (step 2), we note that the interior nodes $\mathcal{V}_2$ are naturally decomposed into disconnected "subdomains" whose only connections are through the interface nodes $\mathcal{V}_1$. Thus, A_2 is block diagonal, with blocks corresponding to subdomains or to individual nodes in $\mathcal{V}_1$. As a result, if we assign each subdomain to its own processor, step 2 can be solved in parallel.

Next, we need to solve systems involving $A_1 = M + \frac{\tau}{2} X_1 A X_1$ (step 3). This is a block diagonal matrix whose largest block is of the same size as $\mathscr{V}_1$, so it is much smaller than the original system. Also note that $X_1 A X_1$ (and hence A_1) is *sparse*, with nonzero entries corresponding to neighboring interface nodes *only*. This is unlike a Schur complement approach, where the elimination of interior nodes introduces additional connections between non-neighboring interface nodes. However, the unknowns corresponding to edges from different subdomains are coupled through cross points, see Fig. 2, leading to a system that is globally coupled.

We now show how we can overcome this bottleneck by reducing the interface system to an even smaller one that has only as many variables as there are *cross points* in the domain. Let N be the number of subdomains, i.e., the number of connected components of $\mathscr{V}_2$. We partition the set $\mathscr{V}_1$ of interface nodes into edges $\{\mathscr{E}_1, \dots, \mathscr{E}_m\}$ between subdomains and $\mathscr{C}$, the set of cross points, so that $\mathscr{V}_1 = \mathscr{C} \cup \left(\cup_{j=1}^m \mathscr{E}_j\right)$. We now permute the blocks of A_1 so that edges are ordered first and the cross points last. If we let $\mathbf{u}_j$ be the unknowns corresponding to $\mathscr{E}_j$ and $\mathbf{v}$ be those belonging to cross points, we get

$$\begin{bmatrix} E_1 & & & G_1 \\ & E_2 & & G_2 \\ & & \ddots & \vdots \\ & & E_m & G_m \\ G_1^T & G_2^T & \cdots & G_m^T & C \end{bmatrix} \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_m \\ \mathbf{v} \end{pmatrix} = \sum_{i=1}^N \mathbf{f}_i,$$

where E_j are sparse matrices corresponding to couplings within $\mathscr{E}_j$, G_j are the connections between $\mathscr{E}_j$ and the cross points, and C represents the connections among cross points themselves (typically $C = 0$). The $\mathbf{f}_i$ represent contributions of subdomain i to the right-hand side, e.g., contributions from nodes in subdomain i that are adjacent to $\mathscr{E}_j$. Then the Schur complement with respect to the cross points becomes

$$\left(C - \sum_{j=1}^m G_j^T E_j^{-1} G_j\right) \mathbf{v} = R_C \sum_{i=1}^N \sum_{j=1}^m (I - R_j^T G_j^T E_j^{-1} R_j) \mathbf{f}_i, \tag{3}$$

where R_j is the restriction from $\mathscr{V}_1$ to $\mathscr{E}_j$, $j = 1, \dots, m$ and R_C the restriction from $\mathscr{V}_1$ to $\mathscr{C}$.Thus, each term in the sum on the right-hand side can be computed independently by subdomain i; moreover, since edges are one-dimensional, E_j is typically a tridiagonal matrix that can be factored easily. In addition, $R_j \mathbf{f}_i$ is nonzero only if $\mathscr{E}_j$ is an edge of subdomain i, so the inner sum contains only as many terms as there are edges in the subdomain boundary. Thus, the contribution $G_j^T E_j^{-1} G_j$ and the corresponding right-hand side can be calculated in parallel, and it remains to solve the Schur complement system, whose size is typically comparable to the number of subdomains. Once $\mathbf{v}$ is known, the $\mathbf{u}_j$ can be calculated in parallel by back substitution, which completes Step 3 in the CNPC algorithm. Thus, the cost of the coarse solve is low, similar to the cost of one coarse grid correction step in other domain decomposition methods, such as FETI-DP [5].

3 Convergence of the CNPC method

In this section, we outline the convergence analysis of the CNPC method under simultaneous time and spatial grid refinement. For more details, see [12]. For ease of presentation, we assume a uniform rectangular grid in which all control volumes are of size h^2, so that $M = h^2 I$. Then (2) is a second-order discretization of (1):

$$-\mathscr{L}u(\cdot,t) = \frac{1}{h^2}[Au(\cdot,t) + B\mathbf{u}_\Gamma(t)] + O(h^2).$$

We assume that the boundary data and source terms are sufficiently smooth, so that $u(x,t)$ has as many continuous spatial and temporal derivatives as needed.

Lemma 1. *The CNPC method can be written as*

$$D\mathbf{u}^{n+1} + \frac{k}{2}(I + \frac{k}{2}X_2AX_1)B\mathbf{u}_\Gamma(t_{n+1}) = C\mathbf{u}^n - \frac{k}{2}(I - \frac{k}{2}X_2AX_1)B\mathbf{u}_\Gamma(t_n) + \tau g(\cdot, t_{n+1/2}),$$

where $k = \tau/h^2$, $D = (I + \frac{k}{2}X_2A)(I + \frac{k}{2}X_1A)$ and $C = (I - \frac{k}{2}X_2A)(I - \frac{k}{2}X_1A)$. Moreover, the stability matrix $D^{-1}C$ satisfies $\|D^{-1}C\|_W < 1$ for any $\tau > 0$ and $h > 0$, where $\|\cdot\|_W$ is induced by the vector norm $\|\mathbf{u}\|_W^2 := \mathbf{u}^T(I + \frac{k}{2}AX_1)A(I + \frac{k}{2}X_1A)\mathbf{u}$.

Recall that the classical Crank–Nicolson method can be written as

$$(I + \frac{k}{2}A)\mathbf{u}^{n+1} + \frac{k}{2}B\mathbf{u}_\Gamma(t_{n+1}) = (I - \frac{k}{2}A)\mathbf{u}^n - \frac{k}{2}B\mathbf{u}_\Gamma(t_n) + \tau g(\cdot, t_{n+1/2}).$$

Thus, we see that CNPC and the classical Crank–Nicolson (CN) method differ by

$$\hat{\rho}_n := \frac{k^2}{4}X_2AX_1[A(\mathbf{u}^{n+1} - \mathbf{u}^n) + B(\mathbf{u}_\Gamma(t_{n+1}) - \mathbf{u}(t_n))] = -\frac{\tau^3}{4h^2}X_2AX_1\left(\mathscr{L}(\partial_t u(\cdot, t_{n+1/2})) + O(h^2)\right).$$

This observation, combined with the fact that the truncation error of CN is $O(\tau^2 + h^2)$, yields the following lemma.

Lemma 2. *The local truncation error ρ_n of the CNPC method at time step n satisfies*

$$\rho_n = \tau\left[-\frac{\tau^2}{4h^2}X_2AX_1\left(\mathscr{L}(\partial_t u(\cdot, t_{n+1/2})) + O(\tau^2) + O(h^2)\right) + O(\tau^2) + O(h^2)\right].$$

In particular, if $\tau = O(h^\alpha)$ with $\alpha \geq 1$, then $\rho_n = \tau \cdot [O(h^2) + O(h^{2\alpha-2})]$.

Note that the $O(h^{2\alpha-2})$ term comes from the term $\frac{\tau^2}{4h^2}X_2AX_1$. Fig. 3 shows the local truncation error for a two-subdomain decomposition with $\tau = O(h)$, for which Lemma 2 predicts $\rho_n/\tau = O(1)$. Although this is true near the interface, we observe that the error is much smaller away from the interface, where X_2AX_1 vanishes.

Let $\varepsilon_n := u(\cdot, t_n) - \mathbf{u}^n$ denote the global error of the method at step n. If $\varepsilon_0 = 0$, i.e., if the correct initial conditions are used, then a standard argument shows that

$$\varepsilon_n = \sum_{j=1}^{n}(D^{-1}C)^{n-j}D^{-1}\rho_{j-1}.$$

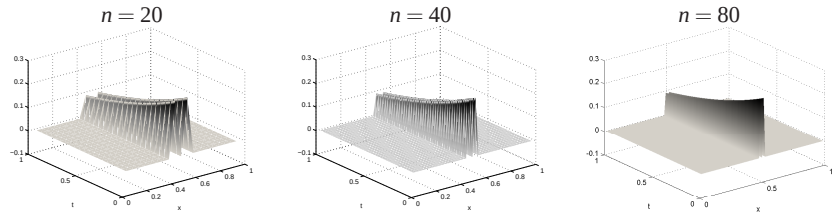

Fig. 3 Local truncation error of the CNPC method for a 1D two-subdomain problem with $u_t = u_{xx} + g(x,t)$, $\tau = h = 1/n$, where $n = 20, 40, 80$.

We now split ρ_n into the interface part $\hat{\rho}_n$ and the $O(h^2)$ part and treat them differently. The smoothness of $\hat{\rho}_n$ in time allows us to prove the following lemma.

Lemma 3. *Let $\hat{\varepsilon}_n = \sum_{j=1}^{n} (D^{-1}C)^{n-j} D^{-1} \hat{\rho}_{j-1}$ be the global error due to the interface. Then*

$$\|\hat{\varepsilon}_n\|_A \leq 4\tau^2 \cdot \max_{0 \leq l \leq n-1} \|X_2 A X_1 z_l\|_{A^{-1}} + O(\tau^4),$$

where $z_0 = -\mathscr{L}\partial_t u(\cdot, t_{1/2})$ and $z_l = -\mathscr{L}\partial_t^2 u(\cdot, t_l)$ for $l \geq 1$.

Since $\|u\|_{H^1(\Omega)}$ is spectrally equivalent to $\|u(\cdot)\|_A$, we can use Lemma 3 to obtain a bound for $\|\varepsilon_n\|_{H^1(\Omega)}$. To do so, we estimate

$$\|X_2 A X_1 z_l\|_{A^{-1}} = \|A^{-1/2}(I - X_1) A X_1 z_l\|_2 \leq \|A^{1/2} X_1 z_l\|_2 + \sqrt{\|X_1 A^{-1} X_1\|_2} \cdot \|A X_1 z_l\|_2.$$

But $X_1 A^{-1} X_1 = S_1^{-1}$, where S_1 is the Schur complement of A with respect to the interface. Thus, we can invoke the well-known Sobolev estimate [17, Lemma 4.11], cf. [2], which states that for a decomposition of Ω into shape-regular, conforming subdomains with diameter H, we have the condition number estimate

$$\kappa(S_1) := \|S_1\|_2 \|S_1^{-1}\|_2 \leq \frac{C}{Hh}.$$

Since A has been scaled in such a way that $\|S_1\|_2 = O(1)$, we conclude that $\|S_1^{-1}\|_2 \leq Ch^{-1}H^{-1}$. Additionally, since there are $O(h^{-1})$ points per interface and $O(H^{-1})$ interfaces, we have $\|X_1 z_l\|_2 = O(h^{-1/2}H^{-1/2})$. Combining these estimates leads to our main result.

Theorem 1. *Let Ω be partitioned into shape-regular, conforming subdomains Ω_i with diameter $\leq H$. Then for $\tau = \gamma h^\alpha$ for $\gamma > 0$ and $\alpha \geq 1$, the error of the CNPC method satisfies*

$$\|\varepsilon_n\|_{H^1(\Omega)} \leq \frac{Ch^\beta}{H}, \tag{4}$$

where $\beta = \min\{2\alpha - 1, 2\}$.

Thus, for a fixed number of subdomains, the method is second order if and only if $\alpha \geq 3/2$. For $\alpha = 1$, i.e., for $\tau = O(h)$, the method is only first order, unlike the classical CN method; this is due to the local inconsistency near subdomain interfaces.

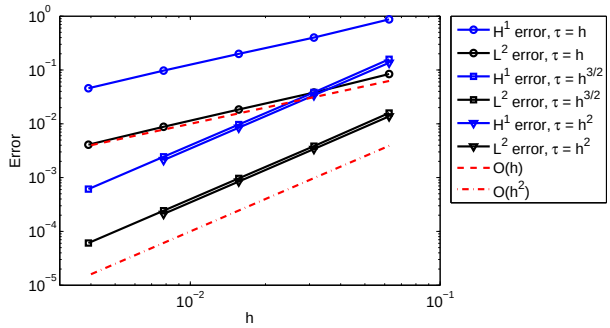

Fig. 4 Error of the CNPC scheme for the 2D heat equation $u_t - \Delta u = g(x,y,t)$ on $\Omega = (0,1)^2$.

4 Numerical results

We apply the CNPC method to solve

$$\partial_t u - \Delta u = g(x,y,t), \qquad (x,y) \in \Omega = (0,1) \times (0,1),$$

The domain Ω is decomposed into 4×4 equal subdomains, and the PDE discretized using a standard 5-point finite difference stencil in space. The initial conditions $u(x,y,0)$ and the source term $g(x,y,t)$ are chosen so that the exact solution is $u(x,y,t) = \sin(3\pi x)(1-e^{2y})(1-e^{y-1})\sqrt{1+t}$. Figure 4 shows the maximum L^2 and H^1 error of the method over the time interval $t \in (0,1)$, with $\tau = h^\alpha$ for $\alpha = 1, \frac{3}{2}, 2$. As predicted by Theorem 1, the error behaves like $O(h)$ for $\tau = h$, and $O(h^2)$ for $\alpha = \frac{3}{2}$ and 2. Moreover, we also see that using the finer time step $\tau = h^2$ only improves the error marginally when compared to $\tau = h^{3/2}$.

Table 1 shows the error of the method for $\tau = h$ and $\tau = h^{3/2}$, when Ω is decomposed into $N \times N$ subdomains with $N = 1/H$. We see that for $\tau = h$, the estimate (4) is sharp; indeed, the errors are approximately constant along the diagonals, except for the column $N = 2$. For $\tau = h^{3/2}$, the estimate is too conservative, as the error does not deteriorate as the number of subdomains increases. This appears to be a 2D effect, since the estimate is sharp for $\tau = h^{3/2}$ in the 1D case. Thus, there appears to be a subtle interplay between temporal and spatial interpolation errors that gives rise to this "superconvergence" behavior.

Conclusions and outlook. The CNPC method allows one to solve diffusion problems in parallel to second-order accuracy without iterating, provided $\tau = O(h^{3/2})$ or smaller. For 3D problems, the Schur complement (3) becomes much denser; one alternative is to use a two-level approach, by first correcting the face values using explicit edge and vertex values, and then correct the edge and vertex values using the face values. The error analysis for this variant, as well as for more general equations (e.g. the advection-diffusion equation), will be the subject of a future paper.

Table 1 Maximum L^2 error for the 2D example .

	$\tau = h$				$\tau = h^{3/2}$			
	Subdomains per direction ($N = 1/H$)				Subdomains per direction ($N = 1/H$)			
$n = 1/h$	2	4	8	16	2	4	8	16
16	7.540e-02	2.347e-01	3.300e-01		5.888e-02	6.585e-02	7.165e-02	
32	2.265e-02	1.399e-01	2.330e-01	3.185e-01	1.448e-02	1.397e-02	1.392e-02	1.402e-02
64	1.291e-02	7.602e-02	1.382e-01	2.391e-01	3.607e-03	3.425e-03	3.296e-03	3.168e-03
128	6.941e-03	4.006e-02	7.405e-02	1.397e-01	9.010e-04	8.513e-04	8.107e-04	7.571e-04
256	3.597e-03	2.053e-02	3.838e-02	7.426e-02	2.252e-04	2.124e-04	2.015e-04	1.860e-04

References

1. Bennequin, D., Gander, M., Halpern, L.: A homographic best approximation problem with application to optimized Schwarz waveform relaxation. Math. Comp. **78**(265), 185–223 (2009)
2. Brenner, S.C.: The condition number of the Schur complement in domain decomposition. Numer. Math. **83**, 187–203 (1999)
3. Cai, X.C.: Additive Schwarz algorithms for parabolic convection-diffusion equations. Numer. Math. **60**, 41–61 (1991)
4. Dawson, C.N., Du, Q., Dupont, T.F.: A finite difference domain decomposition algorithm for numerical solution of the heat equation. Math. Comp. **57**, 63–71 (1991)
5. Farhat, C., Lesionne, M., Le Tallec, P., Pierson, K., Rixen., D.: FETI-DP: a dual-primal unified FETI method — part I: a faster alternative to the two-level FETI method. Internat. J. Numer. Methods Engrg. **50**, 1523–1544 (2001)
6. Gander, M.J., Hairer, E.: Nonlinear convergence analysis for the parareal algorithm. In: Domain Decomposition Methods in Science and Engineering XVII, pp. 45–56. Springer (2008).
7. Gander, M.J., Halpern, L.: Optimized Schwarz waveform relaxation for advection reaction diffusion problems. SIAM J. Numer. Anal. **45**, 666–697 (2007)
8. Gander, M.J., Stuart, A.: Space-time continuous analysis of waveform relaxation for the heat equation. SIAM J. Sci. Comput. **19**(6), 2014–2031 (1998)
9. Giladi, E., Keller, H.B.: Space-time domain decomposition for parabolic problems. Numer. Math. **93**, 279–313 (2002)
10. Janssen, J., Vandewalle, S.: Multigrid waveform relaxation on spatial finite element meshes: the continuous case. SIAM J. Sci. Comput. **17**, 133–155 (1996)
11. Kuznetsov, Y.A.: Overlapping domain decomposition methods for parabolic problems. In: Domain Decomposition Methods in Science and Engineering. AMS (1992)
12. Kwok, F.: A parallel predictor-corrector crank–nicolson method for the solution of parabolic equations. submitted (2012)
13. Lions, J.L., Maday, Y., Turinici, G.: A parareal in time discretization of PDE's. C.R. Acad. Sci. Paris, Série I **332**, 661–668 (2001)
14. Meurant, G.: Numerical experiments with domain decomposition methods for parabolic problems on parallel computers. In: Domain Decomposition Methods for Partial Differential Equations. SIAM (1991)
15. Rempe, M.J., Chopp, D.L.: A predictor-corrector algorithm for reaction-diffusion equations associated with neural activity on branched structures. SIAM J. Sci. Comput. **28**, 2139–2161 (2006)
16. Shi, H.S., Liao, H.L.: Unconditional stability of corrected explicit-implicit domain decomposition algorithms for parallel approximation of heat equations. SIAM J. Numer. Anal. **44**(4), 1584–1611 (2006)
17. Toselli, A., Widlund, O.B.: Domain Decomposition Methods — Algorithms and Theory, *Springer Series in Computational Mathematics*, vol. 34. Springer, Berlin Heidelberg (2005)