Convergence of PARAREAL for a Vibrating String with Viscoelastic Damping

Martin J. GANDER, Thibaut LUNET, Aušra POGOŽELSKYTĖ

1 Model equation for a vibrating string

We consider an elastic string of length L, attached at its two end points, vibrating in a plane due to an initial deformation corresponding to a pinch in the middle, see Figure 1. Its deformation is represented by a scalar function $u(x,t)$, with $x \in [0,L]$, $t \in [0,T]$, and $u(0,t) = u(L,t) = 0$. This simple configuration is the basis for more complex problems that can model guitar and piano strings (see, e.g., [6, 2]), or similar musical instruments.

Generally, such a problem is modeled using the wave equation, possibly adding a first order damping term to produce the so called *Telegrapher's equation*,

$$\partial_{tt} u(x,t) = c^2 \, \partial_{xx} u(x,t) - R \, \partial_t u(x,t) \, , \tag{1}$$

with c the wave velocity and R a damping parameter; setting $R = 0$ leads back to the wave equation. The vibration period of the string is defined as

$$T_W := 2L/c \, . \tag{2}$$

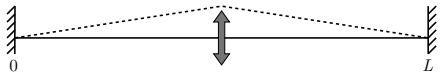

Fig. 1: Vibrating string attached at its two end points, with initial deformation after being plucked.

Martin J. GANDER
University of Geneva, e-mail: martin.gander@unige.ch

Thibaut LUNET
University of Geneva, e-mail: thibaut.lunet@unige.ch

Aušra POGOŽELSKYTĖ
University of Geneva, e-mail: ausra.pogozelskyte@unige.ch

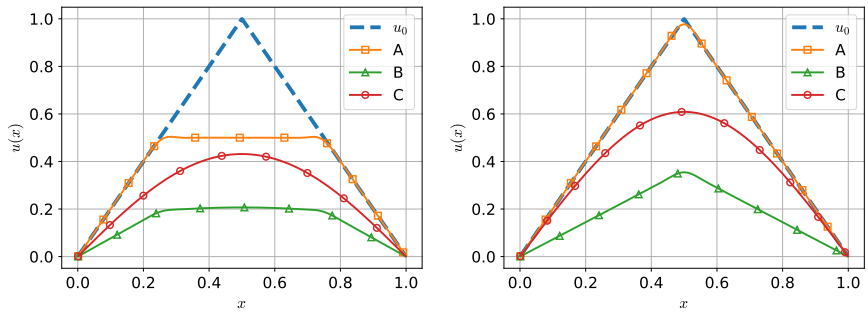

Fig. 2: Solution of the wave equation (A), the Telegrapher's equation with $R = 1$ (B), and the wave equation with viscoelastic damping $\gamma = 0.03$ (C). The dashed line is the initial condition, u_0, of all the equations for reference. Left: $t = \frac{9}{8} T_W$. Right: $t = T_W$.

Equation (1) does however not model the physical behavior of the string in Figure 1 accurately. To illustrate this, we show the numerical solution of (1) in Figure 2, at time $t = \frac{9}{8} T_W$ (left), and $t = T_W$ (right). Curve A represents the wave equation solution ($R = 0$): the string forms a central, unphysical plateau during the oscillation and comes back to the same plucked initial position after one oscillation ($t = T_W$). Adding the damping term ($R = 1$, curve B) reduces the amplitude of the vibration, but the shape still corresponds to the unphysical shape produced by the wave equation.

To correct this, we replace the damping term in (1) by a modified one,

$$\partial_{tt} u(x,t) = c^2 \, \partial_{xx} u(x,t) + \gamma \, \partial_{txx} u(x,t) \,, \tag{3}$$

where γ is a different damping parameter. We call this the *wave equation with viscoelastic damping*. A numerical solution of (3) is shown in Figure 2, curve C, which now looks closer to what we would expect from physics. The viscoelastic damping term in (3) is of prime importance when modeling string vibration. As shown in [5], $u(x,t)$ is a linear combination of string *modes*[1],

$$\xi_n(x) := \sin\left(\frac{\kappa \pi x}{L}\right), \quad \kappa \in \mathbb{N}^* \,, \tag{4}$$

with κ the mode number. In the physical world, when vibrating, each mode is damped at a different rate: high frequency modes (large κ) are damped quickly while low frequency modes (small κ) persist longer. The viscoelastic term can model this behaviour while the damping term in (1), also called *fluid term*[2], introduces the same damping for all modes.

[1] Those modes are also the eigenfunctions of the one dimensional Laplacian with Dirichlet boundary conditions.

[2] Actually, a more accurate model for a vibrating guitar string in [6] considers both fluid and viscoelastic terms. But for simplicity, we will consider here only the viscoelastic term.

To do Fourier analysis, we take the initial condition to be a string mode (4). This leads to a closed form solution of (3),

$$u(x,t) = e^{-\mu_\kappa t}\left[\cos(\tilde{\omega}_\kappa t) + \frac{\omega_\kappa^2}{2}\sin(\tilde{\omega}_\kappa t)\right]\sin\left(\frac{\kappa\pi x}{L}\right), \tag{5}$$

with $\mu_\kappa := \gamma\frac{\kappa^2\pi^2}{2L^2}$, $\tilde{\omega}_\kappa := \omega_\kappa\sqrt{1-\left(\frac{\kappa\pi}{2}\frac{\gamma}{cL}\right)^2}$ and $\omega_\kappa := c\frac{\kappa\pi}{L}$, provided the mode number κ is low enough for the mode to still be oscillating. This is equivalent to the discriminant when solving (3) being negative, which means

$$\omega_\kappa^2 < \frac{4c^4}{\gamma^2} \quad\Longleftrightarrow\quad \kappa < \frac{2}{\pi}\frac{cL}{\gamma} = \frac{2}{\pi}\mathcal{W}, \tag{6}$$

where we introduced $\mathcal{W} := \frac{cL}{\gamma}$, which is the equivalent of a Reynolds (or Peclet) number for advection. Indeed, equation (3) is purely hyperbolic when $\mathcal{W} \to +\infty$ (vibration with no damping) and purely parabolic when $\mathcal{W} \to 0$ (no vibration). Furthermore, the number of vibrating modes is limited by the value of $\mathcal{W}$ (see (6)). For one given mode, $\mathcal{W}$ also defines, together with T_W (see (2)), the lifespan of the vibration

$$\tau_\kappa := \frac{1}{\mu_\kappa} = \frac{\mathcal{W}}{\kappa^2\pi^2}T_W, \tag{7}$$

which represents the time for the mode amplitude to be reduced to $36.8\% = e^{-1}$ of its initial value. As $\mathcal{W}$ gets larger, τ_κ increases, hence a numerical simulation may require more time steps to keep a good accuracy (for a more complete investigation, see Section 3). This can greatly increase computation time, so we now investigate the possibility of using time parallelization to speedup computations.

2 The Parareal algorithm

Time parallel time integration received sustained attention over the last decades, for a review, see [8]. Renewed interest in this area was sparked by the invention of the Parareal algorithm [13] for solving initial value problems of the form

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \mathcal{L}(\mathbf{u}(t),t), \quad \mathbf{u}(0) = \mathbf{u}_0, \quad t \in [0,T], \tag{8}$$

with $\mathcal{L}\colon \mathbb{R}^p \times \mathbb{R}^+ \to \mathbb{R}^p$, $\mathbf{u}(t) \in \mathbb{R}^p$, $\mathbf{u}_0 \in \mathbb{R}^p$, p being the total number of degrees of freedom, and T a positive real value. Problem (8) often arises from the spatial discretization of a (non-)linear system of partial differential equations (PDEs) through the method-of-lines.

For Parareal, one decomposes the global time interval $[0,T]$ into N time subintervals $[T_{n-1},T_n]$ of size ΔT, $n = 1,\ldots,N$, where N is the number of processes to be considered for the time parallelization. In the following, we denote by $\mathbf{U}_n$

the approximation of $\mathbf{u}$ at time T_n, i.e., $\mathbf{U}_n \approx \mathbf{u}(T_n)$. Let $\mathcal{F}^{n_F}_{T_{n-1}\to T_n}(\mathbf{U}_{n-1})$ denote the result of approximately integrating (8) on the time subinterval $[T_{n-1}, T_n]$ from a given starting value $\mathbf{U}_{n-1}$ using a fine propagator $\mathcal{F}$ and n_F time steps (with time step $\Delta t_F := (T_{n-1} - T_n)/n_F$). Similarly, Parareal also needs a coarse propagator $\mathcal{G}$ (using for example n_G time steps), which has to be much cheaper than $\mathcal{F}$ resulting in less accuracy (i.e. $n_F \gg n_G$).

The Parareal algorithm consists of a prediction step and a correction iteration. In the prediction step, Parareal computes an initial guess for the starting values $\mathbf{U}^0_n$ at the beginning of each time subinterval using the coarse propagator,

$$\mathbf{U}^0_0 = \mathbf{u}_0, \quad \mathbf{U}^0_n = \mathcal{G}^{n_G}_{T_{n-1}\to T_n}(\mathbf{U}^0_{n-1}), \;\; n = 1, \ldots, N\,. \tag{9}$$

A correction iteration is then applied in Parareal, using the fine propagator $\mathcal{F}$ on each time subinterval concurrently,

$$\mathbf{U}^k_n = \mathcal{F}^{n_F}_{T_{n-1}\to T_n}(\mathbf{U}^{k-1}_{n-1}) + \mathcal{G}^{n_G}_{T_{n-1}\to T_n}(\mathbf{U}^k_{n-1}) - \mathcal{G}^{n_G}_{T_{n-1}\to T_n}(\mathbf{U}^{k-1}_{n-1})\,, \tag{10}$$

where $\mathbf{U}^k_n$ denotes the approximation of $\mathbf{u}$ at time T_n at the k-th iteration of Parareal ($k = 1, \ldots, K$, $n = 1, \ldots, N$). While the application of $\mathcal{F}$ can be performed independently for each time subinterval, Parareal remains limited by the sequential nature of the coarse integration performed by $\mathcal{G}^{n_G}_{T_{n-1}\to T_n}$ in (10). Parareal will thus reduce the total computational time compared to a direct time-serial integration only if the application of $\mathcal{G}$ is cheap enough and if the total number of Parareal iterations K is small. A more complete description of parallel speedup for Parareal can be found in [1].

While this algorithm works well for parabolic problems, it is known to struggle when the problem of interest is close to hyperbolic (see, e.g., [16, 7, 15]). In our case, this happens when $\mathcal{W}$ becomes large in (3), similarly to what one obtains for the Navier-Stokes equations, as $\mathcal{W}$ plays the same role as the Reynolds number. In the latter case, keeping the same accuracy for the fine solver when the Reynolds number increases is very important (c.f., [14, Sec. 3.5]), as the use of the fine solver with incorrect mesh resolution can lead to a misinterpretation of Parareal convergence results (see, e.g., [10, Sec. 4]). Hence in the next section, we investigate the accuracy of our space time discretization and its link to $\mathcal{W}$ and other parameters.

3 Space mesh requirement for fixed error tolerance

We solve (3) numerically using a uniform spatial mesh with n_x points ($x_j := j\,\Delta x$, with $\Delta x := \frac{L}{n_x+1}$). Denoting by $u_j(t) \approx u(x_j, t)$ and $\dot{u}_j(t) \approx \partial_t u(x_j, t)$, we define

$$\mathbf{v}(t) := [u_1(t), ..., u_{n_x}(t), \dot{u}_1(t), ..., \dot{u}_{n_x}(t)]^\top. \tag{11}$$

We use second-order centered finite differences, which leads to a tridiagonal square matrix A of size n_x. Applying the method-of-lines to (3) yields

$\mathcal{W}$	$\kappa = 1$			$\kappa = 2$			$\kappa = 4$		
	n_x	n_t	ϵ	n_x	n_t	ϵ	n_x	n_t	ϵ
100	305	619	1.008%	432	219	1.007%	610	78	1.009%
1000	965	19555	1.012%	1365	6916	1.012%	1929	2444	1.014%
10000	3050	618060	1.015%	4314	218550	1.014%	6100	77258	1.014%

Table 1: Mesh resolution needed for one percent relative error when varying n_x according to $\mathcal{W}$ and κ, with $\sigma = 10$. The number of time steps n_t is set to simulate $[0, T]$ with $T \approx \tau_\kappa$.

$$\frac{d\mathbf{v}}{dt} = \begin{pmatrix} 0 & I \\ c^2 A & \gamma A \end{pmatrix} \mathbf{v} = L\mathbf{v}\,, \tag{12}$$

with I the identity matrix of size n_x. For the time integration, we use a second order SDIRK2 scheme, integrating up to $T = \tau_\kappa$ with n_t time steps ($t_i := i\Delta t$ with $\Delta t := T/n_t$). We keep a constant CFL number,

$$\sigma := \frac{c\,\Delta t}{\Delta x} = 10\,. \tag{13}$$

We define the relative numerical error as

$$\epsilon := \max_{t \in \{0, t_1, \ldots, t_{n_t}\}} \frac{\left\|\mathbf{u}(t) - \mathbf{u}_{\text{theory}}(t)\right\|_2}{\|\mathbf{u}(0)\|_2}\,, \tag{14}$$

with $\mathbf{u}(t)$ the part of $\mathbf{v}(t)$ containing only u values, and $\mathbf{u}_{\text{theory}}(t)$ the analytic solution from (5) evaluated at the grid points x_j.

Looking at similar problems (e.g. advection-diffusion [9]), one can expect that when we keep a fixed mesh resolution in space (and in time), the error increases with κ and $\mathcal{W}$. Hence, we assume that the minimal value of n_x for which the error ϵ is lower than a given tolerance follows a law of the form

$$n_{x,\min} = C\kappa^\alpha \mathcal{W}^\beta. \tag{15}$$

We compute $n_{x,\min}$ for different values of κ and $\mathcal{W}$ using a trial and error procedure, and then the parameters C, α and β are determined by least square regression. Setting $\epsilon \leq 0.01$ (i.e., less than 1% error) with $\sigma = 10$, we find for our space time discretization

$$n_{x,\min} \approx 30.5\sqrt{\kappa\,\mathcal{W}}\,. \tag{16}$$

In Table 1, we use (16) to give the values (n_x, ϵ) for several combinations of κ and $\mathcal{W}$, which confirms well our empirical law (16). Furthermore, we also indicate the number of time steps n_t required to compute the whole time interval $[0, \tau_\kappa]$ (i.e., the time period during which the mode vibrates, with τ_κ defined in (7)). This shows an important increase in the problem size with $\mathcal{W}$, since generally $n_t \gg n_x$, which motivates time parallelization for such problems.

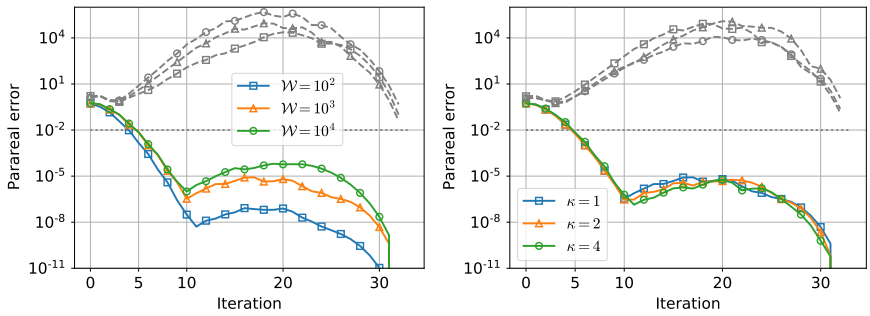

Fig. 3: Convergence of Parareal *for the wave equation with viscoelastic damping, using* $N = 32$ *and* $m = 8$. *Left:* $\kappa = 1$, *varying* $\mathcal{W}$. *Right:* $\mathcal{W} = 1000$, *varying* κ.

4 Numerical experiment with Parareal

We apply Parareal to the wave equation with viscoelastic damping (3), using N processors. We use the same space-time discretization as in Section 3, such that n_x and the time step Δt_F of the fine solver are fully determined by $\mathcal{W}$ and the mode number κ of the initial condition (4) (see Table 1). We denote by m the ratio between the coarse and the fine time step, i.e. $m = \Delta t_G / \Delta t_F$, and we set the number of time-steps per time-interval for the coarse and fine solver (n_F and n_G) such that $n_G \geq 1$ and that the final time of simulation T is close to τ_κ. Finally we compute the error of Parareal for each iteration as

$$E^k := \max_{n\in\{1,\dots,N\}} \frac{\left\|\mathbf{U}_n^k - \mathbf{U}_n^F\right\|}{\|\mathbf{U}_0\|}, \tag{17}$$

where $\mathbf{U}_n^F$ is the solution at the end of each time sub-interval obtained by the fine solver run sequentially.

In Figure 3, we plot the Parareal error at each iteration for different values of the parameters κ and $\mathcal{W}$. We observe two convergence dynamics: for the first few iterations, the error decreases super-linearly and rapidly goes below the fine solver accuracy of 1%, in around five iterations. Then, after about 10 iterations, divergence sets in and a bump forms until the last iteration when Parareal must converge to the fine solution after $k = N$ iterations [11]. This bump is due to the amplification of higher frequency modes in the Parareal iteration, and if Parareal is initialized with a random initial guess, we get the grey dashed convergence (or more divergence) curves in Figure 3, which shows how important the initialization here is and that Parareal struggles to generically solve such close to being hyperbolic problems. While this bump is not so much influenced by low initial modes κ, it does increase with $\mathcal{W}$. Especially when N gets large, this turns out to be problematic for larger values of $\mathcal{W}$, i.e., when the problem becomes more hyperbolic, even with a good coarse initial approximation, see Figure 4.

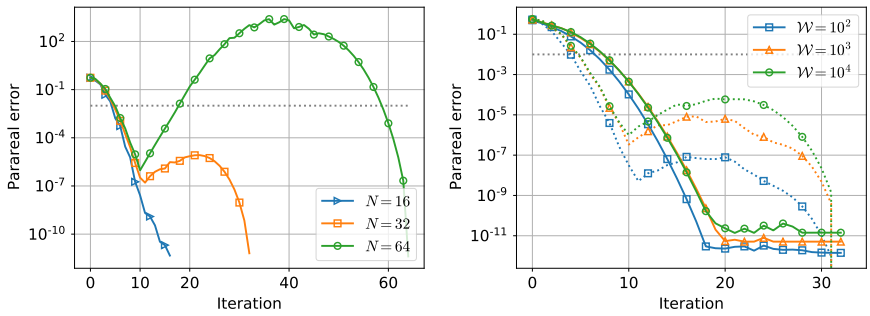

Fig. 4: Influence of N and the coarse solver's phase error on PARAREAL convergence. Left: $\mathcal{W} = 1000$, $m = 8$, varying N. Right: $N = 32$, $n_F = n_G$, and larger damping parameter for coarse solver, results from Figure 3 (left) in dotted lines and $\mathcal{W}$ is indicated for the fine solver.

A similar bump has been observed in [10] for the advection equation, and it is due to the amplification of high frequency error components in the PARAREAL iteration [10], because of the hyperbolic nature of advection: the PARAREAL correction step amplifies these high frequencies, which are present even in a smooth low frequency initial guess due to round-off error. The more processors one uses, the more these high frequency components are amplified, and the higher the bump becomes, even with a smooth low frequency initial guess. A theoretical way to avoid this problem is to impose very high regularity [3]. A more practical way is to reduce the number of processors N in order to limit the amplification induced by the PARAREAL iterations. For our problem, we show in Figure 4 (left) the impact of reducing N, and as expected, the bump is reduced and even disappears for low values of N. However, this limits the number of processors that can be used, and is thus also less useful in practice for parallel computations.

It has been shown for advection in [15] that removing the phase error of the coarse solver greatly improves convergence of PARAREAL. In order to simulate a coarse solver almost free of phase error, we consider using the same space-time discretization for both the coarse and fine solver, but with a larger damping parameter for the coarse solver. This is not useful in practice either since it makes the coarse solver as expensive as the fine solver (see [12] that can make using the same grids practical), but gives us further theoretical insight. We set the ratio between the coarse and fine damping parameter (around 5) such that the error between the two is equal to the one obtained when $\Delta t_G = m\,\Delta t_F$ with $m = 8$ (results in Figure 3 (left)). We plot the convergence of PARAREAL in Figure 4 (right), and see that now the convergence for the first iterations is slightly slower than in Figure 3, but the bump is no longer present in the later iterations.

To conclude, we have shown that under the condition that the fine solver has a sufficient mesh accuracy for the problem considered (determined by κ and $\mathcal{W}$), PARAREAL with a smooth low frequency initial guess obtained from the coarse solver converges for the first few iterations when applied to low frequency modes, which

have the longest vibration time (see (7)). However divergence occurs afterward, due to the amplification of higher frequency modes by the PARAREAL iteration. We have then shown that removing the phase error between the coarse and fine solver can improve PARAREAL convergence. Designing inexpensive coarse solvers for hyperbolic-type problems that do not produce phase error with the fine solver may allow PARAREAL to become more efficient for such problems: for direct constructions using dispersion correction for the advection equation in 1D, see [4], and for a rapid coarse solver based on the same mesh as the fine solver but solved by diagonalization for general hyperbolic problems, see [12].

References

1. Aubanel, E.: Scheduling of tasks in the Parareal algorithm. Parallel Computing **37**(3), 172–182 (2011)
2. Chabassier, J., Chaigne, A., Joly, P.: Modeling and simulation of a grand piano. The Journal of the Acoustical Society of America **134**(1), 648–665 (2013)
3. Dai, X., Maday, Y.: Stable Parareal in time method for first- and second-order hyperbolic systems. SIAM Journal on Scientific Computing **35**(1), A52–A78 (2013)
4. De Sterck, H., Falgout, R.D., Friedhoff, S., Krzysik, O.A., MacLachlan, S.P.: Optimizing MGRIT and Parareal coarse-grid operators for linear advection. arXiv preprint arXiv:1910.03726 (2019)
5. Derveaux, G.: Modélisation numérique de la guitare acoustique. (2002)
6. Derveaux, G., Chaigne, A., Joly, P., Bécache, E.: Time-domain simulation of a guitar: Model and method. The Journal of the Acoustical Society of America **114**(6), 3368–3383 (2003)
7. Gander, M.J.: Analysis of the Parareal algorithm applied to hyperbolic problems using characteristics. Bol. Soc. Esp. Mat. Apl. **42**, 21–35 (2008)
8. Gander, M.J.: 50 years of time parallel time integration. In: T. Carraro, M. Geiger, S. Körkel, R. Rannacher (eds.) Multiple Shooting and Time Domain Decomposition Methods, pp. 69–114. Springer (2015)
9. Gander, M.J., Lunet, T.: A Reynolds number dependent convergence estimate for the Parareal algorithm. In: International Conference on Domain Decomposition Methods, pp. 277–284. Springer (2018)
10. Gander, M.J., Lunet, T.: Toward error estimates for general space-time discretizations of the advection equation. Computing and Visualization in Science **23**(1), 1–14 (2020)
11. Gander, M.J., Vandewalle, S.: Analysis of the Parareal time-parallel time-integration method. SIAM J. Sci. Comput. **29**(2), 556–578 (2007)
12. Gander, M.J., Wu, S.L.: A diagonalization-based parareal algorithm for dissipative and wave propagation problems. SIAM Journal on Numerical Analysis **58**(5), 2981–3009 (2020)
13. Lions, J.L., Maday, Y., Turinici, G.: A "Parareal" in time discretization of PDE's. C. R. Math. Acad. Sci. Paris **332**(7), 661–668 (2001)
14. Lunet, T., Bodart, J., Gratton, S., Vasseur, X.: Time-parallel simulation of the decay of homogeneous turbulence using parareal with spatial coarsening. Computing and Visualization in Science **19**(1), 31–44 (2018)
15. Ruprecht, D.: Wave propagation characteristics of Parareal. Computing and Visualization in Science **19**(1-2), 1–17 (2018)
16. Staff, G.A., Rønquist, E.M.: Stability of the Parareal algorithm. In: R. Kornhuber, et al (eds.) Domain Decomposition Methods in Science and Engineering, Lecture Notes in Computational Science and Engineering, vol. 40, pp. 449–456. Springer (2005)