Fast Parallelizable Galerkin Method for Initial and Boundary Value Problems

Sabin Anton[1,2], Davide Amato[2]

[1]*Astrodynamics and Space Missions Department, TU Delft, Delft, The Netherlands*
[2]*Department of Aeronautics, Imperial College London, London, United Kingdom*
Email: svanton@tudelft.nl, d.amato@imperial.ac.uk

Abstract –: Efficient numerical methods for the solution of Initial and Boundary Value Problems is of continued importance in astrodynamics are of continued importance in astrodynamics. This paper proposes a novel Galerkin method for the solution of IVPs and BVPs. The method works by projecting the solution onto a Chebyshev basis, and by finding the projection coefficients that zero the ODE residual through Newton-Rhapson iterations. The method is tested against classical Runge-Kutta methods and Modified Chebyshev-Picard Iterations (MCPI), which is a similar pseudo-spectral method for the solution of IVPs and BVPs. The Galerkin method is up to 300 times faster than state-of-the-art Runge-Kutta solvers for IVPs, and up to 80 times faster than MCPI in the solution of BVPs.

I. Introduction

Solution methods for IVPs and BVPs are of fundamental importance and continued relevance in astrodynamics. The sustained exponential increase in the number of launched payloads in recent years and the entry into service of several governmental and commercial sensor networks will result in a drastic increase in the amount of observations that will be processed by imminent Space Situational Awareness (SSA) systems. This fact underscores the continued need for efficient solution methods for initial and boundary value problems, which need to be scalable to future scenarios in which millions of objects will likely become detectable.

Classical methods for the solution of IVPs and BVPs rely on sequential schemes such as Runge-Kutta, multistep, and extrapolation methods [1]. All of these enjoy a long heritage in astrodynamics, and their error behaviour and performance is well understood. However, such methods do not use any assumption on the nature of the solution (such as the quasi-periodicity typical of solutions to the perturbed two-body problem), often do not naturally provide a continuous solution in the integration interval, and are non-trivial to parallelize. These drawbacks have been addressed in recent literature through Methods of Weighted Residuals (MWRs). In MWRs, the form of the solution is given, and the problem is shifted to finding a set of parameters that nulls the *residual*, i.e., the quantity obtained by plugging a solution guess in a set of Ordinary or Partial Differential Equations (ODEs/PDEs) [2]. The structure of MWRs allows relatively straightforward parallelization, and an appropriate choice of the basis functions on which the solution is expressed results in efficient methods, especially for quasi-periodic solutions with multiple time scales which are usually encountered in astrodynamics. The MWR that has found the widest applications in the astrodynamics community is MCPI [3], [4]. MCPI projects the solution on a Chebyshev basis, and solves the weak form of the ODE through a Picard iterative procedure. MCPI enforces the condition of zero residuals at a set of integration nodes, and can thus be seen as a type of pseudospectral method [2]. Recently, the Theory of Functional Connections (TFC) has been developed as a general method for functional interpolation in constrained problems. TFC has been applied to several problems in nonlinear dynamics and control, among which the solution of nonlinear ODEs [5], with high efficiency and accuracy. In the framework of MWRs, TFC is a least-squares method as the square of the residual of a set of ODEs is minimized.

In this work, we present a novel method for the solution of IVPs and BVP. Unlike MCPI, the method nulls the residual over the entire integration interval (instead than at discrete points), and uses Newton-Rhapson iterations instead of Picard iterations to solve the MWR equations. The novel method uses the Galerkin approach, in which the functions on which the residual and the solution are projected, i.e. the test and trial functions respectively, are identical (more precisely, they span the same Hilbert subspace). Galerkin spectral methods have found widespread application in solving Partial Differential Equations (PDEs) in fluid dynamics [6], and have been successfully applied in optimal control problems [7]. Galerkin methods have also been recently used to solve the Fokker-Planck equation, which gives a formally correct solution to the diffusion of uncertainty along time [8]. To the authors' knowledge, this is the first time that Galerkin methods have been used for the efficient solution of IVPs and BVPs.

In the paper, we provide a brief derivation of the Galerkin method for IVPs and BVPs, followed by an assessment of its performance against classical Runge-Kutta methods and MCPI. Although the current implementation is serial, we show that the Galerkin method is easy to parallelize and provide an estimate of the expected reduction in computational time.

II. Galerkin Method

A. IVP statement

The goal of the method is to solve IVPs

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{x}, t) \tag{1}$$

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0, \tag{2}$$

for all $t \in [t_0, t_\mathrm{f}]$, where $\boldsymbol{x}(t) : \mathbb{R} \to \mathbb{R}^N$ is the N-dimensional state vector expressing the solution, the Right-Hand Side (RHS) $\boldsymbol{f}(\boldsymbol{x}, t) : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$ is a smooth vector field, $\boldsymbol{x}_0$ is the initial state, and t_0, t_f are the initial and final times of integration. The integration interval $[t_0, t_f]$ is often long compared to the characteristic time scale of the system; for example, we are usually interested in computing the solution over an integration interval that is much larger than the orbital period. This makes the approximation of the solution through aperiodic functions commonly used in MWRs difficult, as typical IVP solutions are periodic and evolve over multiple time scales. We therefore split the integration interval into S sub-intervals $[t_i, t_{i+1}]$, $i \in \{0, \dots, S\}$ that are solved sequentially, where S (equivalently, the interval length) is chosen by the user. The method is applied to each of the sub-problems, with continuity enforced by setting the initial condition of the current sub-problem to the solution at the end of the previous, $\boldsymbol{x}_{0_{i+1}} = \boldsymbol{x}(t_{i+1})$. Finally, we scale the time variable such that we solve the following IVP:

$$\begin{aligned} \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\tau} &= \boldsymbol{g}(\boldsymbol{x}, \tau) \\ \boldsymbol{x}(-1) &= \boldsymbol{x}_0, \end{aligned} \tag{3}$$

where $\boldsymbol{g}(\boldsymbol{x}, \tau) : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$ is the scaled right-hand side. The problem above is then solved for all $\tau \in [-1, 1]$.

B. Galerkin method for IVPs

The Galerkin method presented in this paper is a specific type of MWR. Define the residual associated to (3) as

$$\boldsymbol{r}(\boldsymbol{\xi}, \tau) \equiv \frac{\mathrm{d}\boldsymbol{\xi}(\tau)}{\mathrm{d}\tau} - \boldsymbol{g}(\boldsymbol{\xi}, \tau). \tag{4}$$

Clearly, if $\hat{\boldsymbol{x}}(\tau)$ is the exact solution to (3), then $\boldsymbol{r}(\hat{\boldsymbol{x}}, \tau) = \boldsymbol{0}$, whereas any other approximation to the solution will result in a non-zero residual.

In MWRs, we set the zero residual condition by enforcing it to be orthogonal to an ordered basis of known $M+1$ *test functions* $\{\boldsymbol{w}_0(\tau), \boldsymbol{w}_1(\tau), \dots, \boldsymbol{w}_M(\tau)\}, \boldsymbol{w} : \mathbb{R} \to \mathbb{R}^N$ under the inner product:

$$\begin{aligned} \langle \boldsymbol{w}_m(\tau), \boldsymbol{r}(\boldsymbol{x}, \tau) \rangle = 0 \quad \iff \\ \int_{-1}^{1} \boldsymbol{w}_m^\intercal(\tau) \left(\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\tau} - \boldsymbol{g}(\boldsymbol{x}, \tau) \right) \mathrm{d}\tau = 0, \end{aligned} \tag{5}$$

for all $m \in 0, 1, \dots, M$. Integrating (5) by parts leads to the *weak form* of the ODE (3):

$$\begin{aligned} &\boldsymbol{w}_m^\intercal(1)\boldsymbol{x}(1) - \boldsymbol{w}_m^\intercal(-1)\boldsymbol{x}_{0_i} \\ &\quad - \int_{-1}^{1} \frac{\mathrm{d}\boldsymbol{w}_m^\intercal(\tau)}{\mathrm{d}\tau} \boldsymbol{x}(\tau)\mathrm{d}\tau \\ &- \int_{-1}^{1} \boldsymbol{w}_m^\intercal(\tau)\boldsymbol{g}(\boldsymbol{x}, \tau)\mathrm{d}\tau = 0. \end{aligned} \tag{6}$$

Note that the value of the solution at the initial time $\boldsymbol{x}_{0_i} \equiv \boldsymbol{x}(-1)$ is known. Equation (6) is solved through Newton-Rhapson iterations by linearizing the RHS about the solution at iteration k. By changing notation and rearranging (6) we get

$$\begin{aligned} &\int_{-1}^{1} \left(\frac{\mathrm{d}\boldsymbol{w}_m^\intercal}{\mathrm{d}\tau} + \boldsymbol{w}_m^\intercal \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}^k} \right) \boldsymbol{x}^{k+1}\mathrm{d}\tau \\ &\quad -\boldsymbol{w}_m^\intercal(1)\boldsymbol{x}^{k+1}(1) = -\boldsymbol{w}_m^\intercal(-1)\boldsymbol{x}_{0_i} - \\ &\int_{-1}^{1} \boldsymbol{w}_m^\intercal \left[\boldsymbol{g}(\boldsymbol{x}^k, \tau) - \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}^k} \boldsymbol{x}^k \right] \mathrm{d}\tau. \end{aligned} \tag{7}$$

The solution is projected over a set of $L+1$ *trial functions* $\phi_\ell(t)$,

$$\boldsymbol{x}(\tau) = \sum_{\ell=0}^{L} \boldsymbol{\Phi}_\ell(\tau)\boldsymbol{c}_\ell, \tag{8}$$

where $\boldsymbol{c}_\ell$ are $N \times 1$ vectors of spectral coefficients, and $\boldsymbol{\Phi}_\ell(t) = \boldsymbol{I}\phi_\ell(t)$ is an $N \times N$ diagonal matrix of ℓ-th order trial functions. The Galerkin assumption consisting of assuming identical test and trial functions, $\phi_m(t) = w_m(t)\ \forall m,\ L = M$, is now applied [2]. Plugging (8) into (7) we obtain N scalar equations to find the spectral coefficients at iteration $k+1$:

$$\begin{aligned} &\sum_{\ell=0}^{L} \int_{-1}^{1} \left(\frac{\mathrm{d}\boldsymbol{\phi}_m^\intercal}{\mathrm{d}\tau} + \boldsymbol{\phi}_m^\intercal \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}^k} \right) \boldsymbol{\Phi}_\ell \boldsymbol{c}_\ell^{k+1}\mathrm{d}\tau \\ &\quad - \sum_{\ell=0}^{L} \boldsymbol{\phi}_m^\intercal(1)\boldsymbol{\Phi}_\ell(1)\boldsymbol{c}_\ell^{k+1} = -\boldsymbol{\phi}_m^\intercal(1)\boldsymbol{x}_{0_i} \\ &- \int_{-1}^{1} \boldsymbol{\phi}_m^\intercal \left[\boldsymbol{g}\left(\boldsymbol{x}^k, \tau\right) - \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}^k} \sum_{\ell=0}^{L} \boldsymbol{\Phi}_\ell \boldsymbol{c}_\ell^k \right] \mathrm{d}\tau. \end{aligned} \tag{9}$$

As these N scalar equations must be simultaneously satisfied for all m, we can write the overall equation for the Newton-Rhapson iteration in matrix form:

$$\left(\boldsymbol{P} - \boldsymbol{B}_r + \boldsymbol{J}^k\right) \boldsymbol{c}^{k+1} = -\boldsymbol{b}_l - \boldsymbol{s}^k + \boldsymbol{J}^k \boldsymbol{c}^k, \tag{10}$$

where the block matrices $\boldsymbol{P}$, $\boldsymbol{B}_r$, $\boldsymbol{J}^k$ are where $\boldsymbol{P}$, $\boldsymbol{B}_r$, $\boldsymbol{J}^k$ are $N(M+1) \times N(M+1)$ matrices formed by $N \times N$

blocks,

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{P}_{00} & \boldsymbol{P}_{01} & \cdots & \boldsymbol{P}_{0M} \\ \boldsymbol{P}_{10} & \boldsymbol{P}_{11} & \cdots & \vdots \\ \vdots & & \ddots & \vdots \\ \boldsymbol{P}_{M0} & \cdots & & \boldsymbol{P}_{MM} \end{bmatrix} \tag{11}$$

$$\boldsymbol{B}_r = \begin{bmatrix} \boldsymbol{B}_{r_{00}} & \boldsymbol{B}_{r_{01}} & \cdots & \boldsymbol{B}_{r_{0M}} \\ \boldsymbol{B}_{r_{10}} & \boldsymbol{B}_{r_{11}} & \cdots & \vdots \\ \vdots & & \ddots & \vdots \\ \boldsymbol{B}_{r_{M0}} & \cdots & & \boldsymbol{B}_{r_{MM}} \end{bmatrix} \tag{12}$$

$$\boldsymbol{J}^k = \begin{bmatrix} \boldsymbol{J}^k_{00} & \boldsymbol{J}^k_{01} & \cdots & \boldsymbol{J}^k_{0M} \\ \boldsymbol{J}^k_{10} & \boldsymbol{J}^k_{11} & \cdots & \vdots \\ \vdots & & \ddots & \vdots \\ \boldsymbol{J}^k_{M0} & \cdots & & \boldsymbol{J}^k_{MM} \end{bmatrix} \tag{13}$$

with the blocks being

$$\boldsymbol{P}_{m\ell} = \int_{-1}^{1} \frac{\mathrm{d}\phi_m}{\mathrm{d}\tau} \phi_\ell \boldsymbol{I} \mathrm{d}\tau \tag{14}$$

$$\boldsymbol{B}_{r_{m\ell}} = \phi_m(1)\phi_\ell(1)\boldsymbol{I} \tag{15}$$

$$\boldsymbol{J}^k_{m\ell} = \int_{-1}^{1} \phi_m \phi_\ell \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}^k} \mathrm{d}\tau. \tag{16}$$

and $\boldsymbol{b}_l, \boldsymbol{s}^k, \boldsymbol{c}^k$ are $N(M+1) \times 1$ vectors,

$$\boldsymbol{b}_l = \begin{bmatrix} \boldsymbol{\Phi}_0(-1)\boldsymbol{x}_{0_i} \\ \boldsymbol{\Phi}_1(-1)\boldsymbol{x}_{0_i} \\ \vdots \\ \boldsymbol{\Phi}_M(-1)\boldsymbol{x}_{0_i} \end{bmatrix} \tag{17}$$

$$\boldsymbol{s}^k = \begin{bmatrix} \int_{-1}^{1} \boldsymbol{\Phi}_0 \boldsymbol{g}(\boldsymbol{x}^k, \tau)\mathrm{d}\tau \\ \int_{-1}^{1} \boldsymbol{\Phi}_1 \boldsymbol{g}(\boldsymbol{x}^k, \tau)\mathrm{d}\tau \\ \vdots \\ \int_{-1}^{1} \boldsymbol{\Phi}_M \boldsymbol{g}(\boldsymbol{x}^k, \tau)\mathrm{d}\tau \end{bmatrix} \tag{18}$$

$$\boldsymbol{c}^k = \begin{bmatrix} \boldsymbol{c}^k_0 \\ \boldsymbol{c}^k_1 \\ \vdots \\ \boldsymbol{c}^k_M \end{bmatrix}. \tag{19}$$

Note that $\boldsymbol{P}, \boldsymbol{B}_r$, and $\boldsymbol{b}_l$ do not change during the iterations and can be precomputed.

Each integral in the terms of (10) is computed through a Gauss-Legendre-Lobatto (GLL) quadrature,

$$\int_{-1}^{1} f(\tau)\mathrm{d}\tau \approx \sum_{j=0}^{M} v_j f(\tau_j)$$
$$v_j = \frac{2}{(1-\tau_j^2)\left(\left.\frac{\mathrm{d}p_M}{\mathrm{d}\tau}\right|_{\tau=\tau_j}\right)^2}, \tag{20}$$

where τ_j is the j-th root of the M-th order Legendre polynomial $p_M(\tau)$. The Newton-Rhapson iteration (10) is performed until the coefficient vector converges under a specified tolerance ϵ,

$$\frac{\|\boldsymbol{c}^{k+1} - \boldsymbol{c}^k\|}{\|\boldsymbol{c}^k\|} < \epsilon. \tag{21}$$

C. Galerkin method for BVPs

Similarly to Section II.A, we define the BVP as finding the solution $\boldsymbol{x}(\tau)$ for all $\tau \in [-1, 1]$ to

$$\frac{\mathrm{d}^2\boldsymbol{x}}{\mathrm{d}\tau} = \boldsymbol{g}(\boldsymbol{x}, \tau) \tag{22}$$

$$\boldsymbol{x}(-1) = \boldsymbol{x}_1,\ \boldsymbol{x}(1) = \boldsymbol{x}_2, \tag{23}$$

in which $\boldsymbol{x}_1, \boldsymbol{x}_2$ are known boundary conditions and the same non-dimensionalization as in Section II.A has been applied. For the BVPs, preliminary numerical testing has shown that the splitting of the total solution interval into sub-intervals is unnecessary.

By applying the MWR method with the Galerkin approach to the residual of as done in Section II.B, we arrive at an analogous equation for the Newton-Rhapson iteration for BVPs:

$$\begin{aligned}(\boldsymbol{P}_2 - \boldsymbol{J}^k + \boldsymbol{B}_{2v_r} - \boldsymbol{B}_{2v_l})\boldsymbol{c}^{k+1} \\ = \boldsymbol{b}_{2_r} - \boldsymbol{b}_{2_l} + \boldsymbol{s}^k - \boldsymbol{J}^k\boldsymbol{c}^k,\end{aligned} \tag{24}$$

where the coefficient vectors $\boldsymbol{c}^{k+1}, \boldsymbol{c}^k$, the term $\boldsymbol{s}^k$, and the Jacobian matrix $\boldsymbol{J}^k$ have the same meaning as in Section II.B, $\boldsymbol{P}_2, \boldsymbol{B}_{2v_r}, \boldsymbol{B}_{2v_l}$ are $N(M+1) \times N(M+1)$ matrices formed by assembling M times along rows and columns the following $N \times N$ blocks:

$$\boldsymbol{P}_{2_{m\ell}} = \int_{-1}^{1} \frac{\mathrm{d}^2\phi_m}{\mathrm{d}\tau^2} \phi_\ell \boldsymbol{I} \mathrm{d}\tau \tag{25}$$

$$\boldsymbol{B}_{2v_{r\,m\ell}} = \phi_m(1)\frac{\mathrm{d}\phi_\ell}{\mathrm{d}\tau}(1)\,\boldsymbol{I} \tag{26}$$

$$\boldsymbol{B}_{2v_{l\,m\ell}} = \phi_m(-1)\frac{\mathrm{d}\phi_\ell}{\mathrm{d}\tau}(-1)\,\boldsymbol{I}, \tag{27}$$

and $\boldsymbol{b}_{2_r}, \boldsymbol{b}_{2_l}$, are the $N(M+1) \times 1$ vectors

$$\boldsymbol{b}_{2_l} = \begin{bmatrix} \frac{\mathrm{d}\boldsymbol{\Phi}_0}{\mathrm{d}\tau}(-1)\boldsymbol{x}_1 \\ \frac{\mathrm{d}\boldsymbol{\Phi}_1}{\mathrm{d}\tau}(-1)\boldsymbol{x}_1 \\ \vdots \\ \frac{\mathrm{d}\boldsymbol{\Phi}_M}{\mathrm{d}\tau}(-1)\boldsymbol{x}_1 \end{bmatrix} \tag{28}$$

$$\boldsymbol{b}_{2_r} = \begin{bmatrix} \frac{\mathrm{d}\boldsymbol{\Phi}_0}{\mathrm{d}\tau}(1)\boldsymbol{x}_2 \\ \frac{\mathrm{d}\boldsymbol{\Phi}_2}{\mathrm{d}\tau}(1)\boldsymbol{x}_2 \\ \vdots \\ \frac{\mathrm{d}\boldsymbol{\Phi}_M}{\mathrm{d}\tau}(1)\boldsymbol{x}_2 \end{bmatrix}. \tag{29}$$

Analogously to the IVP case, the terms in (25)–(29) do not change during the iterations and can be precomputed. Equation (24) is iterated until the convergence criterion (21) is satisfied.

Table 1. IVP test case initial conditions.

e	a	i	Ω	ω	θ
0	6678 km				
0.5	13 356 km	30°	0°	0°	10°
0.72	22 260 km				

D. Computational cost of Newton-Rhapson iterations

The matrix $\boldsymbol{J}^k$ in (10) requires computing the Jacobian of the dynamics at many integration nodes, which is generally expensive as the Jacobian is approximated numerically in a general case. In addition, the naïve solution of (10) at each iteration involves inverting the matrix on the left-hand side, which has computational cost $\mathcal{O}(N^3(M+1)^3)$.

The first issue is addressed by adopting Chebyshev polynomials as the orthogonal basis for both test and trial functions. The orthogonality and product-sum properties of Chebyshev polynomials are employed to significantly reduce the evaluation cost of the integrals in (16).

The second issue is addressed by noting that only the Jacobian matrix $\boldsymbol{J}^k$ changes at each iteration in the system matrix $(\boldsymbol{P} - \boldsymbol{B}_r + \boldsymbol{J}^k)$. It is therefore desirable to write the inverse of the system matrix as a function of $(\boldsymbol{P} - \boldsymbol{B}_r)^{-1}$ (which can be precomputed) and $\boldsymbol{J}^k$. This is achieved through the expansion of the system matrix inverse into a power series through the Woodbury matrix identity [9].

E. Initial guess

For IVPs, the initial guess is provided through a low-fidelity solution of (3), which can be provided either through an approximate analytical solution or through a low-fidelity solution computed with a low-order Runge-Kutta method. In this work, the initial solution is provided by an integration with a Runge-Kutta 5(4) method at a coarse tolerance. The initial guess for the coefficients is found through interpolation of the coarse solution with Chebyshev polynomials. For BVPs, preliminary numerical testing has shown that no initial guess for the solution is necessary, and the coefficients at the first iteration can be set to zero. However, if an initial guess to the solution is available then convergence will likely be improved by interpolating it in Chebyshev polynomials and providing it as an initial guess.

III. Performance Analysis

A. Initial Value Problems

In classical sequential numerical integration methods, one usually tunes the accuracy of the method by adjusting a single parameter, which is either the step size (for fixed-step integrators) or the local truncation error tolerance (for variable-step integrators). Although sequential methods with variable order exist, changing the order of the method usually requires significant modifications. In spectral and pseudospectral methods, one can tune the accuracy such as the Galerkin method described here by adjusting either the order of the method (i.e., the number of basis functions) or the sub-interval size (analogous to the step size in sequential methods). These approaches are also called p- and h-refinement in literature on spectral methods [2]. In this section, the accuracy and computational cost of the methods for the solution of typical IVP in astrodynamics are investigated as a function of the spectral order and sub-interval size. As mentioned previously, the m-th basis function $\phi_m(\tau)$ is the Chebyshev polynomial of the first kind of order m. We test the Galerkin method against MCPI, and Runge-Kutta methods of 4$^{\text{th}}$ and 8$^{\text{th}}$ order implemented in the Julia `DifferentialEquations.jl` package [10]. We consider initial conditions for three test cases of varying initial eccentricity, which are reported in Table 1. The physical model considered includes a 200×200 normalized geopotential and the NRLMSISE-00 atmospheric model, with a spacecraft ballistic coefficient of $0.045\,\mathrm{kg\,m^{-2}}$. The equations of motion for in either Cartesian coordinates (Cowell) or modified equinoctial elements (MEEs) are integrated by all the methods. Evaluations of the RHS $\boldsymbol{g}$ are adopted as the metric of computational cost. As the Jacobian $\partial \boldsymbol{g}/\partial \boldsymbol{x}$ is evaluated through central finite differences, a single Jacobian evaluation is counted as $2N = 12$ function calls.

The position error as a function of time computed with respect to an accurate reference solution is shown in Fig. 1 and Fig. 2 for equations in Cartesian coordinates and MEEs, respectively. The computational cost associated to the propagations is shown in Fig. 3 and Fig. 4 for the two sets of equations considered. The sub-interval size and spectral order for the Galerkin and MCPI methods, and the truncation error tolerances for the Runge-Kutta methods, have been chosen to obtain similar errors at the final propagation time, as to enable a fair comparison in computational cost amongst the different numerical methods. When integrating equations in Cartesian coordinates, both MCPI and the Galerkin method significantly outperform the Runge-Kutta solvers for $e = 0, e = 0.5$. The Galerkin method achieves excellent performance, with a number of function calls up to 15 times smaller than the Runge-Kutta solvers and four times smaller than the MCPI method. This advantage disappears for the highly eccentric case ($e = 0.7$), with both Runge-Kutta solvers outperforming the Galerkin method and MCPI. This is due to the increased nonlinearity of the dynamics, which significantly degrades the spectral convergence properties of MWRs. The issue is circumvented by integrating dynamics expressed in MEEs. As Fig. 2 shows, when expressing the dynamics

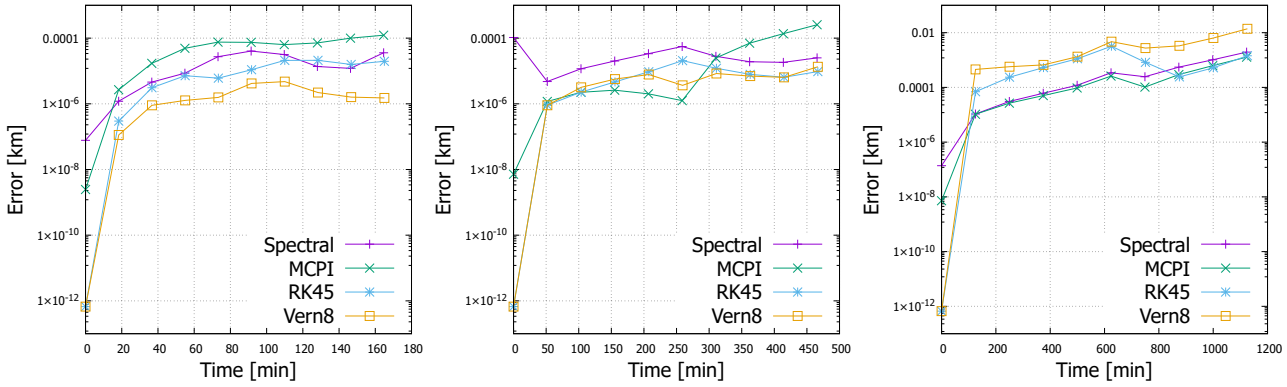


Fig. 1. Position error as a function of time for test cases in Table 1 with initial eccentricities $e = 0$ (left panel), $e = 0.5$, (center panel), $e = 0.72$ (right panel), propagated in Cartesian coordinates (Cowell's method).

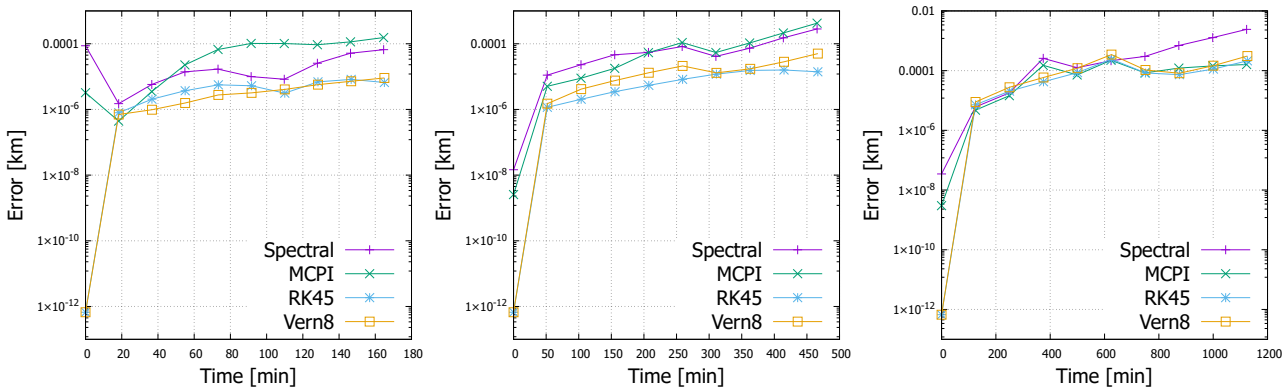


Fig. 2. Position error as a function of time for test cases Table 1 with initial eccentricities $e = 0$ (left panel), $e = 0.5$, (center panel), $e = 0.72$ (right panel), propagated in MEEs.

in MEEs the Galerkin method outperforms the Runge-Kutta solvers by an order of magnitude in function calls.

The performance improvement of the Galerkin method with respect to MCPI can be explained by its better spectral convergence behaviour. Fig. 5 shows the behaviour of the relative spectral error, defined according to (21), along iterations for the Galerkin method and MCPI. It can be seen that the Galerkin method exhibits faster convergence, requiring less iterations to achieve the same error compared to MCPI.

A comprehensive investigation of the performance of the Galerkin method compared to the classical Dormand-Prince 7(8) (DOPRI78) single-step solver [11] has been performed by computing the ratio of function calls required by each method for varying sub-interval size and final absolute error for the test cases in Table 1. The ratio of function calls required by DOPRI78 with respect to those required by the Galerkin method for the same position error is shown in Fig. 6. The speedup is more apparent when the dynamics are smoother, i.e., for circular orbits (for which speedups of up to 300 are achievable) and for dynamics expressed in equinoctial elements. Even for the most challenging case ($e = 0.7$), the Galerkin method is up to 60 times faster than DOPRI78.

B. *Boundary Value Problems*

The performance of the Galerkin method in the solution of BVPs is assessed on the solution of a single-revolution perturbed Lambert's problem for a heliocentric transfer from $\boldsymbol{r}_1 = [1, 0, 0]^\intercal$au to $\boldsymbol{r}_2 = [0, 2, 0]^\intercal$au. Three cases are considered, corresponding to times of flight of 3, 4, and 5 years, respectively. The only perturbation considered is solar radiation pressure (SRP) with a cannonball model, and an SRP ballistic coefficient of 0.1 kg m^{-2}. The accuracy of the solution is tuned by adjusting the spectral order M between 5 and 50, with a fixed Newton-Rhapson tolerance $\epsilon = 10^{-13}$. The performance of the Galerkin method against MCPI is compared in Table 2. Depending on the test case, the Galerkin method is from 5 to 80 times faster than MCPI in terms of function calls. This is due to much faster convergence of the coefficients during the Newton-Rhapson iterations.

C. *Remarks on parallelization*

All the results presented so far refer to a fully serial implementation of the Galerkin method, with no regards for

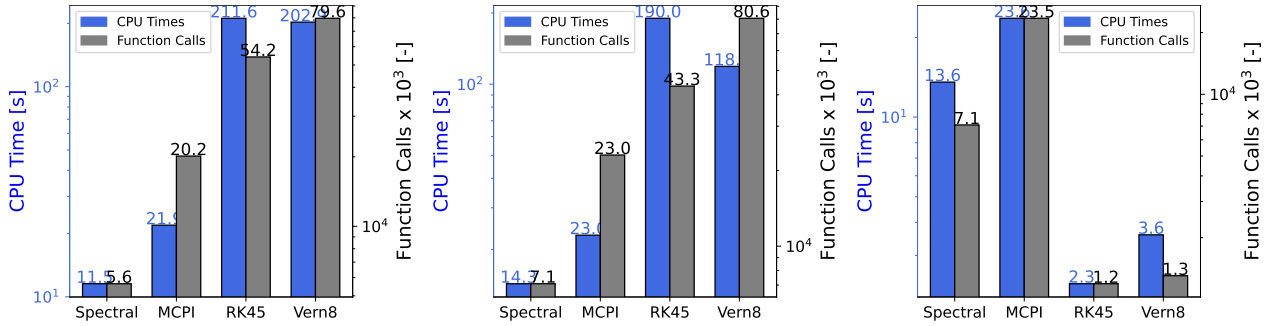


Fig. 3. CPU time and function calls required for the propagation of the three test cases in Table 1 with initial eccentricities $e = 0$ (left panel), $e = 0.5$, (center panel), $e = 0.72$ (right panel), propagated in Cartesian coordinates (Cowell's method).

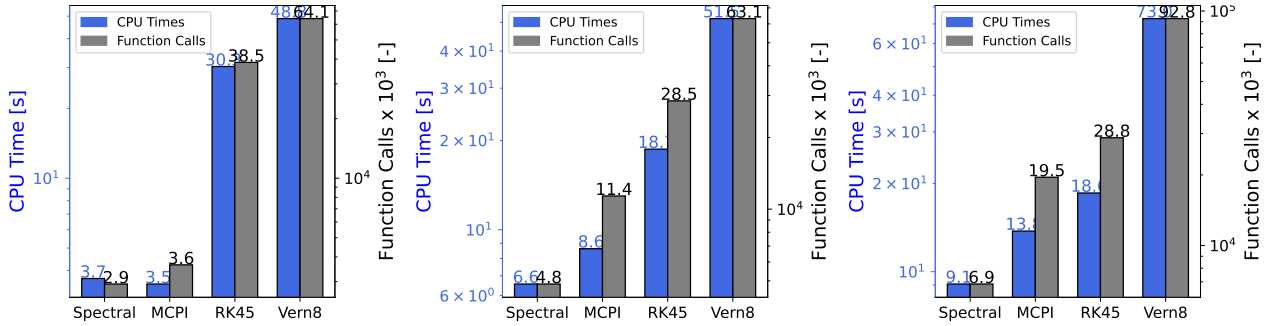


Fig. 4. CPU time and function calls required for the propagation of the three test cases in Table 1 with initial eccentricities $e = 0$ (left panel), $e = 0.5$, (center panel), $e = 0.72$ (right panel), propagated in MEEs.

parallelization. However, an additional advantage of the method is that it requires little effort to parallelize in the solution of both IVPs and BVPs. For physical models of mid- to high-fidelity, the evaluation of the Jacobian $\partial \boldsymbol{g}/\partial \boldsymbol{x}$ in (16) and of the RHS $\boldsymbol{g}$ in (18) are the most computationally intensive tasks and are thus ideal candidates for parallelization. These must be computed independently at $M+1$ GLL nodes during the evaluation of the integrals, and can be parallelized. For expansion orders between 100 and 500 and using the physical model described in Section III.A, at least 70% of the computational time is spent on the evaluation of the RHS and Jacobian. Assuming that these are parallelized over 8 threads, this would result in a further 60% speedup for the method.

IV. Conclusions

A new Galerkin spectral method for the solution of typical Initial and Boundary Value Problems (IVPs/BVPs) in astrodynamics has been developed. The method belongs to the class of methods of weighted residuals, which null the projections of the residual of an ODE (the quantity obtained by plugging a guess solution into the ODE) over a set of test functions. The solution is expanded over a set of trial functions, which coincide with the test functions in the Galerkin approach. The full derivation of the method has been shown for Initial Value Problems, however it can be adapted to Boundary Value Problems with little modifications.

The performance of the method has been tested on IVPs by solving orbit propagation problems with a high-fidelity geopotential model, and has been compared against that of Runge-Kutta solvers and of the Modified-Chebyshev Picard Iterations (MCPI) method, a pseudo-spectral method that similarly achieves a global solution of a set of ODEs. The Galerkin method is up to 300 times faster than the state-of-the-art DOPRI78 sequential solver (for the same accuracy), and 4 times faster than MCPI.

The performance of the Galerkin method on BVPs has been tested by solving perturbed heliocentric Lambert's problems, in which the method has been shown to be up to 80 times faster than MCPI.

V. Acknowledgments

DA thanks Max I. Hallgarten La Casta for preparing a LaTeX version of this article template.

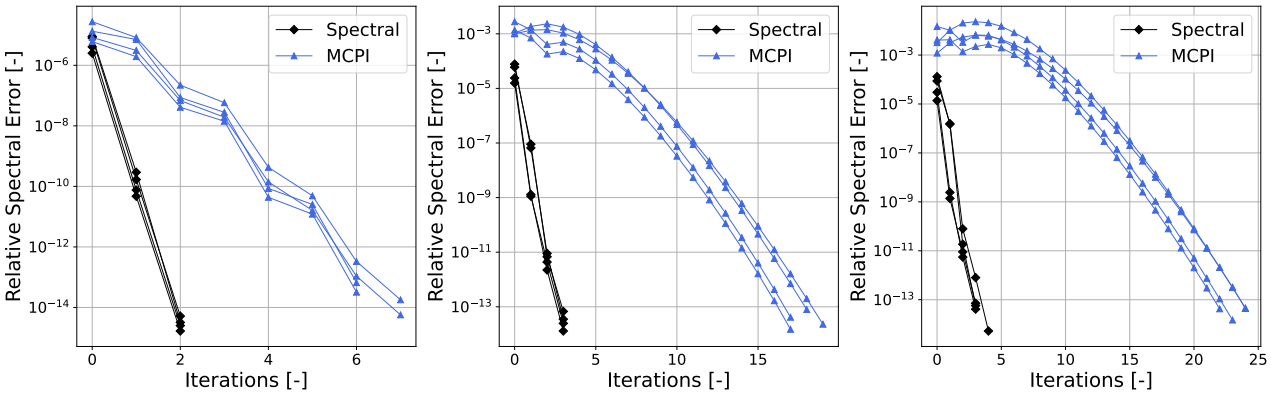
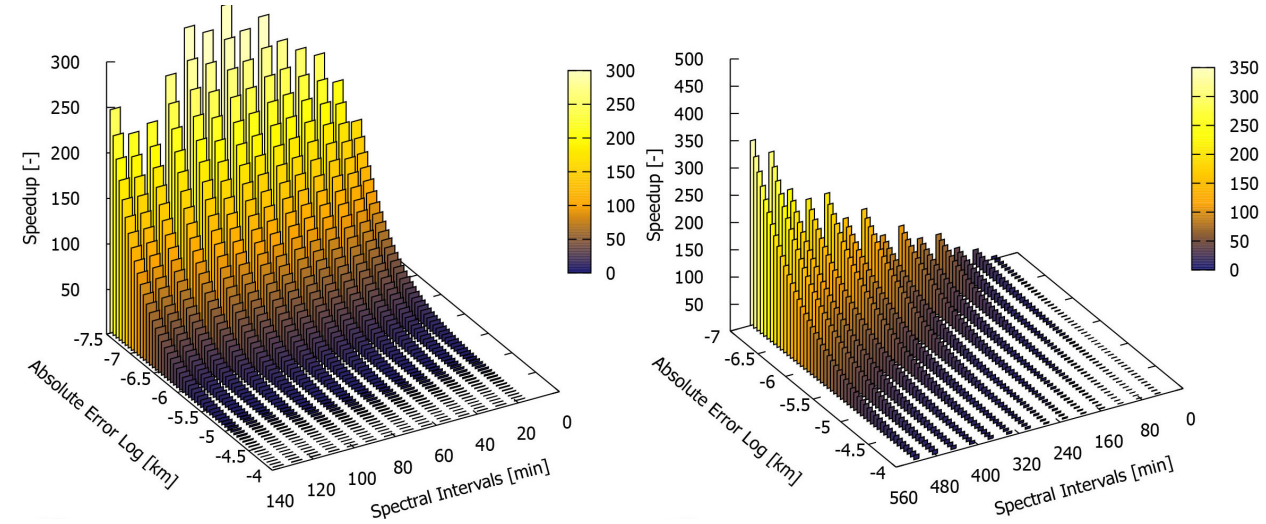


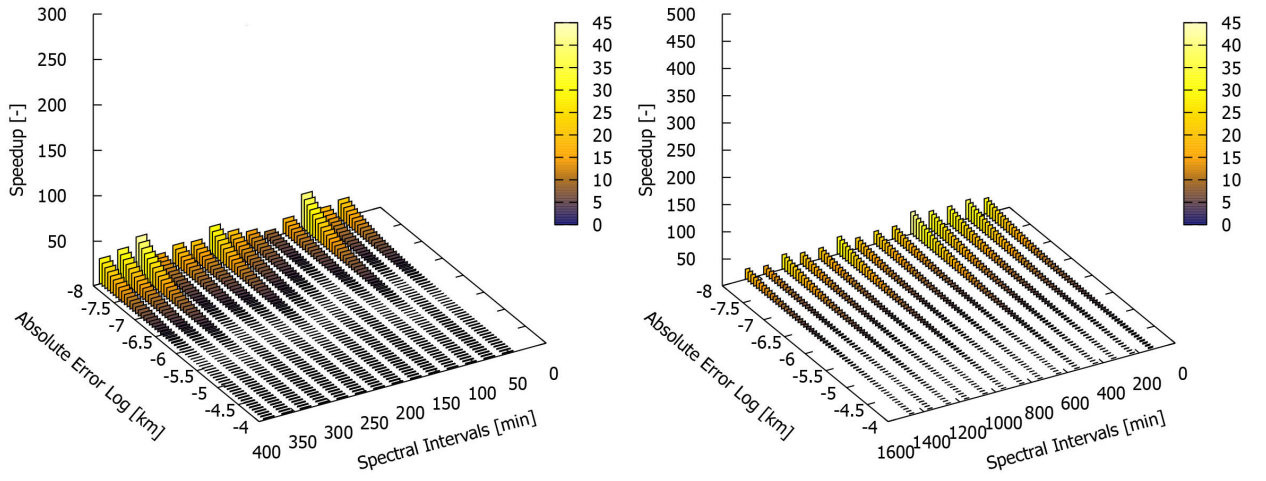
Fig. 5. Spectral convergence behaviour for the Galerkin and MCPI methods for the three test cases in Table 1 with initial eccentricities $e = 0$ (left panel), $e = 0.5$, (center panel), $e = 0.72$ (right panel), propagated in MEEs.

Table 2. Comparison of function calls required to solve Lambert's problems for varying times of flight for the Galerkin method against MCPI.

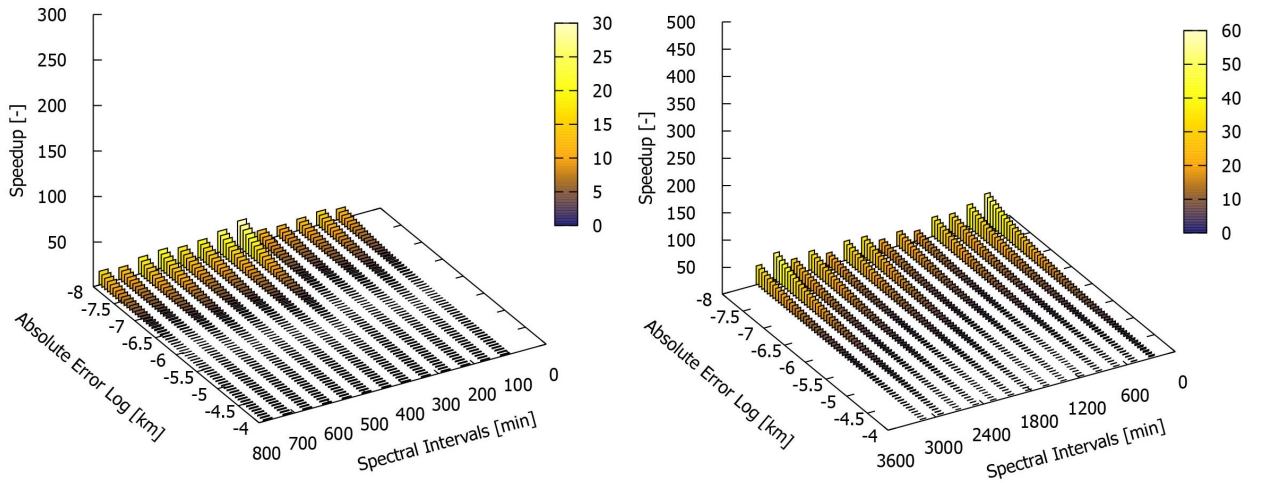
Time of flight	3 years	4 years	5 years
Galerkin Abs. error (au)	1.97×10^{-8}	3.30×10^{-8}	2.01×10^{-7}
MCPI Abs. Error (au)	1.98×10^{-8}	6.76×10^{-8}	5.87×10^{-7}
Galerkin FCalls	220	286	264
MCPI FCalls	1166	2761	20 647



(a) $e = 0$



(b) $e = 0.5$



(c) $e = 0.7$

Fig. 6. Function call ratio $\mathrm{FC_{Gal}}/\mathrm{FC_{DP7}}$ for varying spectral interval size and final absolute error for the test cases in Table 1. Left and right panels refer to equations in Cartesian coordinates and MEEs, respectively.

VI. References

[1] E. Hairer, S. P. Nørsett and G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff Problems* (Springer Series in Computational Mathematics 8), 2nd rev. ed. Heidelberg ; London: Springer, 2009, 528 pp., ISBN: 978-3-540-56670-0 978-3-642-05163-0 978-3-540-78862-1.

[2] G. E. Karniadakis and S. J. Sherwin, *Spectral/Hp Element Methods for Computational Fluid Dynamics* (Numerical Mathematics and Scientific Computation), 2nd ed. New York: Oxford university press, 2005, ISBN: 978-0-19-852869-2.

[3] X. Bai and J. L. Junkins, "Modified Chebyshev-Picard Iteration Methods for Solution of Boundary Value Problems," *The Journal of the Astronautical Sciences*, vol. 58, no. 4, pp. 615–642, 2011.

[4] X. Bai and J. L. Junkins, "Modified Chebyshev-Picard Iteration Methods for Solution of Initial Value Problems," *The Journal of the Astronautical Sciences*, vol. 59, no. 1, pp. 327–351, 2012.

[5] D. Mortari, H. Johnston and L. Smith, "High Accuracy Least-Squares Solutions of Nonlinear Differential Equations," *Journal of Computational and Applied Mathematics*, vol. 352, pp. 293–307, 2019.

[6] C. D. Cantwell, D. Moxey, A. Comerford *et al.*, "Nektar++: An Open-Source Spectral/Hp Element Framework," *Computer Physics Communications*, vol. 192, pp. 205–219, 2015.

[7] M. P. Neuenhofen, E. C. Kerrigan and Y. Nie, "Numerical Comparison of Collocation vs Quadrature Penalty Methods," in *2023 62nd IEEE Conference on Decision and Control (CDC)*, 2023, pp. 4285–4290.

[8] G. Acciarini, C. Greco and M. Vasile, "Uncertainty Propagation in Orbital Dynamics via Galerkin Projection of the Fokker-Planck Equation," *Advances in Space Research*, vol. 73, no. 1, pp. 53–63, 2024.

[9] W. W. Hager, "Updating the Inverse of a Matrix," *SIAM Review*, vol. 31, no. 2, pp. 221–239, 1989.

[10] C. Rackauckas and Q. Nie, "DifferentialEquations.jl – a performant and feature-rich ecosystem for solving differential equations in Julia," *Journal of Open Research Software*, vol. 5, no. 1, 2017.

[11] J. Dormand and P. Prince, "A family of embedded Runge-Kutta formulae," *Journal of Computational and Applied Mathematics*, vol. 6, no. 1, pp. 19–26, 1980.