A Comparison of ODE Solvers for Dynamical Systems with Impacts

Spyridon Dallas
spyro.d.mechs@gmail.com

Konstantinos Machairas
kmach@central.ntua.gr

Evangelos Papadopoulos
egpapado@central.ntua.gr
ASME Member

Department of Mechanical Engineering
National Technical University of Athens
9 Heroon Polytechniou Str.
15780 Athens, Greece

In this paper, a method is developed that results in guidelines for selecting the best Ordinary Differential Equation (ODE) solver and its parameters, for a class of nonlinear hybrid system were impacts are present. A monopod interacting compliantly with the ground is introduced as a new benchmark problem, and is used to compare the various solvers available in the widely used Matlab ODE Suite. To provide result generality, the mathematical description of the hybrid system is brought to a dimensionless form, and its dimensionless parameters are selected in a range taken from existing systems and corresponding to different levels of numerical stiffness. The effect of error tolerance and phase transition strategy is taken into account. The obtained system responses are evaluated using solution speed and accuracy criteria. It is shown that hybrid systems represent a class of problems that cycle between phases in which the system of the Equations of Motion (EOM) is stiff (interaction with the ground), and phases in which it is not (flight phases); for such systems, the appropriate type of solver was an open question. Based on this evaluation, both general and case-specific guidelines are provided for selecting the most appropriate ODE solver. Interestingly, the best solver for a realistic test case turned out to be a solver recommended for numerically nonstiff ODE problems.

I. INTRODUCTION

In the simulation phase of a mechanical system, a representative model of the system is chosen, the dynamical equations describing its behavior are derived as a system of Ordinary Differential Equations (ODEs), and finally an ODE solver is selected from a collection of algorithms and packages, to solve the equations. The large number of methodologies available for solving such an Initial Value Problem (IVP) can be explained by the heterogeneity of the various realistic dynamical problems, and the great difficulty to be all solved numerically by a single algorithm. Notably, mechanical systems that experience repeated impacts, such as legged robots [1] and on-orbit manipulators [2], can present a very rich repertoire of behaviors, making the selection of the proper solver a tedious and challenging task. What makes a good selection is the adequately high speed of the solver for a given accuracy, as well as the small deviation from the true response – usually not known in analytical form.

In this work, we focus on mechanical systems with impacts and especially on hopping single-legged robots, which we consider as simple yet representative for a whole class of systems that include compliant elements in their structures and experience repeated impulsive impacts. Recently, the robotics community has been demonstrating advanced examples of efficient, agile and intelligent legged robots, followed by a large increase in related research output, including theoretical and experimental results accompanied mostly by simulation experiments. However, the criteria used for selecting an ODE solver in the simulations are obscure in most of the works, while even for problems of similar structure, very different solvers are chosen. For instance, focusing on legged robotics simulations, one can find works employing solvers recommended for stiff problems – Matlab ode23s in [3] – and also for nonstiff problems – Matlab ode113 in [4] – for solving similar problems. To the authors' knowledge, the community lacks a set of guidelines to evaluate, compare and choose the proper solver for a given problem in this class. The proper choice of a solver is time consuming, since one has to first find the solvers that converge to a solution, and then choose the most effective one. Taking into account demanding procedures like fixed point searching in legged robots [5] that can take hours or days, choosing the appropriate solver, and solving a problem efficiently with the desired level of accuracy is essential in hard problems such as those involving mechanical impacts.

During the last decades, important theoretical works in the field of numerical analysis showed how different methodologies for solving ODE systems can be implemented into algorithms for solving different kinds of problems. Detailed literature reviews are included in [6] and [7]. Seminal works early set the basis for the systematic evaluation of ODE solvers, while the numerous references found therein reveal the long history of attempts been made in the field [8], [9], and [10]. The initial analyses and results distinguished problems mainly into stiff and nonstiff, while difficulties like discontinuities were not included.

Collections of simple realistic problems were proposed as test suites – sets of benchmark problems – for evaluating the various methodologies and software packages, according to several criteria for accuracy and computational complexity [6], [8], [9], [11], [12], [13]. Until today, these are constantly being extended with new problems along with their solutions to serve as reference databases. A few recent examples of problem sets and frameworks available for evaluation of ODE solvers can be found in [7], [14], [15], [16], [17] and [18]. In addition to these general problem sets, several works focused on evaluating and comparing existing algorithms for solving specific problems in the fields of biology and chemistry [19], combustion chemistry [20], atmospheric chemistry [21], and astrochemical kinetics [22].

A special kind of algorithms and methods were also proposed, which were able to detect when the problem is stiff or nonstiff, and accordingly switch between families of methods to improve their overall performance [7], [23], [24]. Hybrid problems that

include both stiff and nonstiff segments are a fertile ground for this kind of methods, however they have not been fully exploited yet. In general, there are only a few examples concerning ODE solving for hybrid systems. Various solvers were studied on a nonstiff continuous, a stiff continuous, and a stiff hybrid model in [25], while a comparison was conducted for three Matlab ODE solvers for a model that changed from nonstiff to stiff, and back to nonstiff in [25].

In spite of the theoretical results published, the numerous software packages developed, and the large number of application examples presented already, the ODE solver selection problem remains open until today especially in the cases of complex hybrid systems that include impacts. Additional evaluation methodologies and results for this class of systems are welcome in the community, providing insight, practical guidelines, and tests with up to date software packages.

The purpose of this work is to provide basic guidelines regarding the numerical solution of dynamical systems with impacts. More specifically, we aim to determine which method can provide better accuracy and speed of solution when solving the respective system of ODEs in a wide parameter space. Moreover, we aim to understand and evaluate the role of event location functions, [27] – functions built in ODE software packages to aid in programming the transition between phases in hybrid systems – on the speed and accuracy of the method. Despite the earlier preference of the community for implementations in Fortran, the Matlab ODE suite – presented in [28] and more recently described in [29] – soon became a very popular tool with a wide variety of powerful ODE solvers and example problem sets [18], [30]. Based on this, and also on the fact that Matlab has become ubiquitous in engineering studies, we chose it as the framework for our analysis.

To obtain answers to the above questions a system comprised of a vertically hopping monopod, and a nonlinear Hunt-Crossley ground model was selected. To ensure the generality of the results, the system was brought in a dimensionless form with the values of the four dimensionless parameters chosen to belong in a range characterizing existing monopods. Using a numerical stiffness criterion, we investigated the effect of changing each dimensionless parameter at a time on the stiffness of the system. Furthermore, each of the above cases was solved at three different levels of absolute and relative error tolerance (1e-3, 1e-6, 1e-9), to determine which of the seven ODE solvers can cope better in each tolerance level. The results produced by the seven solvers, in the three error tolerance levels, for a range of numerical stiffness depending on the four dimensionless parameters, with and without the use of event functions for phase transition, were evaluated for speed and accuracy. The solvers that performed better for each criterion in each case were rewarded by a point reward system. Based on the point results, one can choose the desired error tolerance, decide whether to use the built-in event location functions, and finally narrow down the solver candidates to one or two that are appropriate for a specific problem.

This work also introduces a new benchmark problem for impacts including a continuous realistic ground model to the existing IVP problem sets, an evaluation method comprised of three ranking criteria, and an investigation on the effects of event location functions on simulations of hybrid dynamical systems with impacts. While most works in the field concentrate on solver comparisons for multiple problems of different nature, this study focuses on a particular problem – considered representative of a class of problems that include repeated impacts – and spans a large percentage of its parameter space, based on realistic values taken from existing mechanical systems.

It is expected that the general conclusions and suggestions of this paper will be useful not only for the presented class of systems, but also for systems with compliant mechanisms in which repeated impacts occur.

II. HYBRID SYSTEM DYNAMICS

A. System description and dynamics

The physical system described in this section is considered hybrid, as its differential equations change accordingly to the succeeding phases that are presented during its movement; phases with and without impact. The system consists of a simple unactuated springy monopod that hops vertically, on a nonlinear compliant ground, under the influence of gravity, see Figure 1. The hopper has toe mass m_1, body mass m_2, free length l_0, and leg stiffness k. Hopping is characterized by the succession of two discrete phases: the *stance phase* occurring when the monopod interacts with the ground, and the *flight phase* that takes place from the instant that the monopod takes-off from the ground until right before it touches the ground again. As long as the monopod stays in contact with the ground, a normal impulsive contact force F_g is applied from the ground to the toe, as the result of an infinitesimal displacement of the ground at the contact point, see Figure 1.

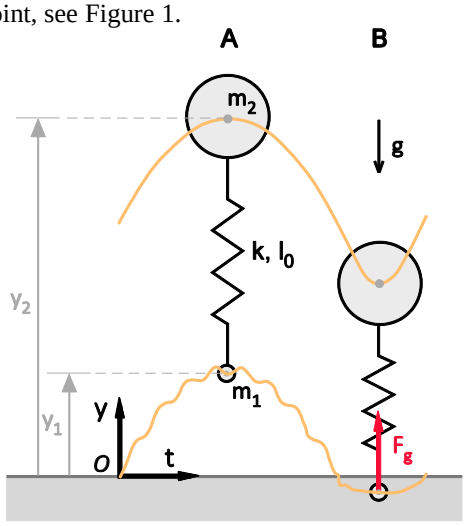

Figure 1. Two snapshots of a vertically hopping monopod at different phases. (A) flight phase, and (B) stance phase.

Among the various impact models proposed in literature [31], [42] the continuous, non-adhesive Hunt-Crossley model was selected for describing the forces exerted from the ground to the toe of the hopper during impact [43],

$$F_g = k_g \left| y_{go} - y_g \right|^n - b_g \dot{y}_g \left| y_{go} - y_g \right|^n \quad (1)$$

where k_g is the ground stiffness, b_g is its damping, $n = 1.5$ is the Hertz coefficient for non-adhesive contact, with y_g being the vertical displacement from the ground level y_{go}, and $\dot{y}_g$ the velocity of the ground contact point. The ground level is set at:

$$y_{go} = 0 \tag{2}$$

The equations of motion (EoM) of the monopod hopper are derived using the Euler-Lagrange method, with generalized coordinates the vertical displacements y_1, y_2 of the toe mass m_1 and the body mass m_2 respectively. They are presented in a unified form for both the flight and the stance phase, as:

$$m_1\ddot{y}_1 + k\left(l_0 + y_1 - y_2\right) + m_1 g = iF_g \tag{3}$$

$$m_2\ddot{y}_2 + k\left(-l_0 - y_1 + y_2\right) + m_2 g = 0 \tag{4}$$

where g is the acceleration of gravity and i is a variable responsible for the transition between phases, so that:

$$i = \begin{cases} 0 & for\ y_1 > 0 \\ 1 & for\ y_1 \le 0 \end{cases} \tag{5}$$

Also, during the stance phase, the displacement and velocity of the toe and of the ground contact point coincide so that:

$$y_1 = y_g \tag{6}$$

$$\dot{y}_1 = \dot{y}_g \tag{7}$$

Taking into account equations (1), (2), (6), and (7), the EoM can be rewritten with respect to the state variables as:

$$m_1\ddot{y}_1 + ib_g\left|y_1\right|^{1.5}\dot{y}_1 + \left(k + ik_g\left|y_1\right|^{0.5}\right)y_1 - ky_2 + m_1 g + kl_0 = 0 \tag{8}$$

$$m_2\ddot{y}_2 - ky_1 + ky_2 + m_2 g - kl_0 = 0 \tag{9}$$

B. Dimensionless EoM

To reduce the number of free parameters and provide generality, the mathematical description of the hopping monopod is written in a dimensionless form using the pi theorem of Buckingham [32]. The dimensionless variables of the system are:

$$y_1^* = \frac{y_1}{l_0} \tag{10}$$

$$y_2^* = \frac{y_2}{l_0} \tag{11}$$

$$t^* = \frac{t}{s} \tag{12}$$

where the * as a superscript denotes a dimensionless variable. The time parameter s is chosen to be equal to the period of the two mass free oscillation during the flight phase:

$$s = 2\pi\sqrt{\frac{m_1 m_2}{(m_1 + m_2)k}} \tag{13}$$

Also, from equations (10), (11), and (12), the dimensionless vertical velocities and accelerations of masses m_1, m_2 are derived as:

$$\dot{y}_1^* = \frac{s}{l_0}\dot{y}_1 \tag{14}$$

$$\dot{y}_2^* = \frac{s}{l_0}\dot{y}_2 \tag{15}$$

$$\ddot{y}_1^* = \frac{s^2}{l_0}\ddot{y}_1 \tag{16}$$

$$\ddot{y}_2^* = \frac{s^2}{l_0}\ddot{y}_2 \tag{17}$$

Replacing the dimensional variables in equations (8), (9) with the dimensionless ones of (10)-(12) and (14)-(17) and after some algebraic manipulation, the dimensionless description of the monopod hopper on compliant ground is:

$$\begin{aligned} &\ddot{y}_1^* + ib^*\left|y_1^*\right|^{1.5}\dot{y}_1^* + \left(4\pi^2 - m_1^*\right)\left(1 + ik^*\left|y_1^*\right|^{0.5}\right)y_1^* - \\ &-\left(4\pi^2 - m_1^*\right)y_2^* + + f^* + \left(4\pi^2 - m_1^*\right) = 0 \end{aligned} \tag{18}$$

$$\ddot{y}_2^* - m_1^*\dot{y}_1^* + m_1^*\dot{y}_2^* + f^* - m_1^* = 0 \tag{19}$$

where b^* is the dimensionless damping ratio, m_1^* is the dimensionless toe mass, k^* is the dimensionless ground stiffness and f^* is the dimensionless force, given by:

$$b^* = \frac{2\pi b_g l_0^{1.5}}{m_1}\left(\frac{m_1 m_2}{(m_1 + m_2)k}\right)^{0.5} \tag{20}$$

$$m_1^* = \frac{4\pi^2 m_1}{(m_1 + m_2)} \tag{21}$$

$$k^* = \frac{k_g l_0^{0.5}}{k} \tag{22}$$

$$f^* = \frac{4\pi^2 m_1 m_2 g}{(m_1 + m_2)kl_0} \tag{23}$$

C. Dimensionless parameter space and numerical stiffness

The performance of various numerical integration methods is to be determined under different impact conditions. These depend on the initial conditions for (18), and (19), and on the values of the parameters (20)-(23). To determine the dimensionless parameter space, plausible parameter values were gathered from existing monopods [33], [34], [35], [36], and Hunt - Crossley ground models [37]. Using (20)-(23), the range for the values of the four dimensionless parameters were derived to be:

$$b^* \in [550,\ 38000] \tag{24}$$

$$m_1^* \in [0.8,\ 20] \tag{25}$$

$$k^* \in [1,\ 480] \tag{26}$$

$$f^* \in [0.01,\ 0.3] \tag{27}$$

The numerical stiffness of the system of differential equations (18), (19) is a function of the above four dimensionless parameters. It is well known that to solve a system of linear differential equations accurately, the integration step size must be much smaller with respect to the period of the fastest phenomenon. The faster a phenomenon is, the larger the real part of the corresponding eigenvalue is. For a system with both very fast and very slow phenomena, a small integration step is needed, while the time of integration is very long, resulting in a numerically stiff system. Many definitions and criteria have been formulated over the years for numerical stiffness [25], [38]. In this work, the stiffness ratio (SR) is used as a representative stiffness criterion defined as:

$$SR = \frac{\max_j \left(\left| \mathrm{Re}\left(\lambda_j \right) \right| \right)}{\min_j \left(\left| \mathrm{Re}\left(\lambda_j \right) \right| \right)} \tag{28}$$

where λ_j is a system eigenvalue. The higher SR ($SR >> 1$) is, the stiffer the system of differential equations becomes.

To determine the stiffness of the differential equations (18), (19) with the dimensionless parameters in the range (24)-(27), the system is linearized in the vicinity of the equilibrium point given by:

$$\mathbf{q}_\mathbf{e}^* = \left[-\left(\frac{4\pi^2 f^*}{k^* m_1^*} \right)^{\frac{2}{3}} \quad 0 \quad -\left(\frac{4\pi^2 f^*}{k^* m_1^*} \right)^{\frac{2}{3}} \quad -\frac{f^*}{m_1^*} + 1 \quad 0 \right]^T \tag{29}$$

The characteristic equation of the system is found for the flight phase $(i = 0)$:

$$\lambda^2 \left(\lambda^2 + 4\pi^2 \right) = 0 \tag{30}$$

and for the linearized impact phase $(i = 1)$:

$$\lambda^4 + \frac{4\pi^2 b^* f^*}{k^* m_1^*} \lambda^3 + \left[4\pi^2 + 1.5\left(4\pi^2 - m_1^* \right) k^* \left(\frac{4\pi^2 f^*}{k^* m_1^*} \right)^{\frac{1}{3}} \right] \lambda^2 +$$

$$+ \frac{4\pi^2 b^* f^*}{k^*} \lambda + 1.5\left(4\pi^2 f^* \right)^{\frac{1}{3}} \left(k^* m_1^* \right)^{\frac{2}{3}} \left(4\pi^2 - m_1^* \right) = 0 \tag{31}$$

Since time was made dimensionless using the period of the free oscillation during the flight phase s, the eigenvalues during flight are independent of the dimensionless parameters, and therefore are omitted from the stiffness analysis that follows.

Focusing on (31), first a nominal dimensionless parameter set is chosen: $b^* = 1951$, $m_1^* = 1.88$, $k^* = 22.8$ and $f^* = 0.051$. The eigenvalues $\lambda_1, \lambda_2, ..., \lambda_4$ are computed starting from the nominal set, and continuing by changing one dimensionless parameter value at a time, for three different values in the intervals (24)-(27). For all these sets, we compute the corresponding numerical stiffness using *SR* as defined in (28).

III. ODE SOLVERS AND THEIR PARAMETERS

Given the dimensionless EoM (18)-(19), the desired parameter set, and the desired initial conditions, the ODE solvers available in Matlab are employed to yield the solution. The solvers and the integration algorithm that they implement are presented in Table 1, or more extensively in [28].

The performance of each solver depends on the selected *error tolerance*, and in turn, this tolerance determines the step size of the integrator. If solutions of large magnitudes are expected, then the relative error tolerance must be set.

On the other hand, if solutions in the vicinity of zero are expected, then the absolute error tolerance must be determined, since then the relative error tolerance tends to infinity. If the solution of a problem is expected both in the vicinity and away from zero, as in the case of the hopping robot toe and main body trajectories, then both error tolerances must be set. To study the effect of error tolerance in the ODE solver performance, the system of differential equations is solved in three different levels of error tolerance, (1e-3), (1e-6) and (1e-9). The error tolerance level in the Matlab ODE suite is set with 'RelTol' and 'AbsTol' parameters.

Table 1. Description of solvers provided by the ODE Matlab suite.

ODE solver	Algorithm Implemented	Recommended for/comments
ode23	Runge - Kutta 23	Nonstiff problems
ode45	Runge - Kutta 45	Nonstiff problems
ode113	Adams - Bashforth - Moulton	Nonstiff problems
ode15s	Numerical differentiation formulas	Stiff problems. Quasi constant step size.
ode23s	Rosenbrock	Stiff problems. Second order.
ode23t	Trapezoidal rule	Moderately stiff problems.
ode23tb	Two stage Runge - Kutta	Stiff problems. First stage, trapezoidal rule. Second stage, second order backward differentiation formula.

During impacts, there is a fast transition from one set of EoM to another. During numerical integration, the *phase transition strategy* employed to describe the physics of the problem, is conjectured also to affect the performance of the ODE solver. The differential equations describing the hopping monopod change when the toe contacts or leaves the ground. This transition can be implemented either with an "if…else…end" statement as part of the EoM, or with the Matlab event location function, as described in [39]. As the system of (18)-(19) is numerically solved, the first transition strategy checks in every time step the value of y_1^*; if it is positive, then i takes the value 0, else it takes the value 0. With this strategy, the toe may penetrate slightly the ground before the ground force is exerted, depending on the size of the time step. With the second transition strategy, the EoM are solved in loops. Every time an event is detected, here when $y_1^* = 0$, the numerical integration stops, and the solutions of the system are accurately found for $y_1^* = 0$. Changing appropriately the EoM depending on the phase of the motion, and using as initial conditions the solution of the system at $y_1^* = 0$, the numerical integration is reinitiated until the next event detection. The procedure continues until the differential equations describing the hybrid system are solved in the desired time interval. With the event transition strategy, the ground forces are exerted on the toe exactly right after $y_1^* = 0$, however by constantly restarting the numerical solver the accuracy of the solution deteriorates. In the case of impacts, where phase transitions exist, it is important to determine how the use of event functions affects the ODE solver performance, in comparison to the simple "if…else…end" statement transition strategy.

IV. PERFORMANCE CRITERIA

To better understand the choice of performance criteria, first the nature of the response is described, see Figure 2. The toe mass m_1 is in general much smaller than the monopod body mass m_2, therefore it oscillates more intensely in both the flight and stance phases. As a result, during the numerical integration of (18)-(19), it is expected that errors in the toe displacement numerical solution with respect to the true solution, are going to appear sooner and be larger than those for the main body. This is confirmed by comparing the top (body) and bottom (toe) responses in Figure 2.

Since the mathematical description of the hybrid system during stance is nonlinear, no analytical solution exists to serve as a benchmark. That is evident even in one degree of freedom systems using the Hunt - Crossley impact formulation when gravitational forces are included [44]. Therefore, a *reference solution* is needed for benchmarking. This reference is obtained as the average of the solutions provided by solvers ode23, ode45, ode113 and ode15s, in a very fine error tolerance (1e-12). These solvers were used for the reference solution, because they can provide the desired error tolerance (1e-12) solution in reasonable time. As discussed earlier, the dimensionless toe displacement $y^*_{1,ref}$ is of critical importance compared to that of the main body. Therefore, the accuracy of each solver is evaluated using $y^*_{1,ref}$ only.

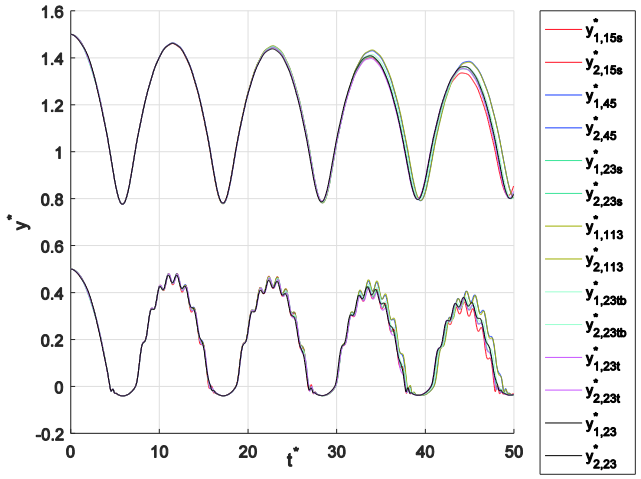

Figure 2. Response of hopping monopod by various solvers, for error tolerance (1e-3) and transition by "if…else…end" statement.

In evaluating an ODE solver's performance, an important criterion is the deviation of the solution compared to the reference one. Therefore, a suitable criterion is the divergence time (*DT)* that corresponds to the first dimensionless integration time at which the normalized error of $y^*_{1,solver}$ from the reference solution $y^*_{1,ref}$ becomes significant, or:

$$\frac{\left|y^*_{1,solver}-y^*_{1,ref}\right|}{\left|y^*_{1,ref}\right|}\geq\varepsilon \tag{32}$$

where ε is chosen to be equal to 0.05 for error tolerance (1e-3), and 0.01 for (1e-6) and (1e-9), also see Figure 3. This divergence is expected to occur during the flight phase, where no damping exists. Since the error is in relative form, (32) is evaluated only away enough from zero, i.e. for $|y_{ref}|\geq 0.05$. The larger *DT* for some solution $y^*_{1,solver}$ is, the more accurate the solution is, see Figure 3 and Figure 4.

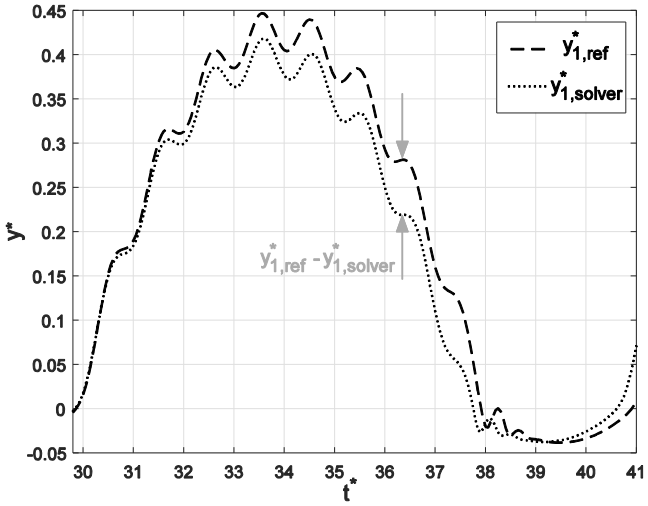

Figure 3. Absolute error between the reference toe displacement and the toe displacement provided by an ODE solver.

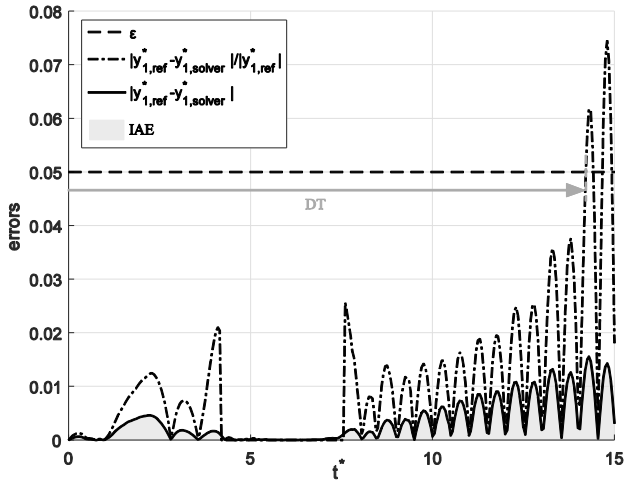

Figure 4. Correlation of absolute and relative error from reference with *DT*, and *IAE* criteria.

While the solution of an ODE solver may become inaccurate during the flight phase considering (32), during the stance phase damping exists that reduces the deviation from the reference. For this reason, a criterion representative for both phases, and valid in the entire dimensionless time integration interval, is introduced. This criterion is the *Integral of the Absolute Error* (*IAE*) of the solution from the reference calculated using the trapezoidal method:

$$IAE=\int_{t_0^*}^{t_f^*}|e|\,dt=\frac{h}{2}\left(|e_0|+|e_N|\right)+h\sum_{m=1}^{N-1}|e_i| \tag{33}$$
$$e_i=y^*_{1,solver,m}-y^*_{1,ref,m},\ h=t^*_{m+1}-t^*_m,\ m=0,\ldots,N$$

and shown in Figure 4. The smaller the *IAE* is for the dimensionless time interval of integration $[t_0^*,t_f^*]$, the greater the overall accuracy of the ODE solver is.

Finally, to evaluate the performance of ODE solvers with respect to the time needed for obtaining a solution, the *Solution Duration* (*SD*) criterion is introduced, defined as the time it takes a solver to integrate the EoM in real time. Its value is measured using the *tic* and *toc* Matlab commands; the *tic* starts a timer right before initiating the solution and the *toc* records the elapsed real time after the solution becomes available. The fastest a solver, the smaller is the value of its *SD.*

V. RESULTS

The system of differential equations (18)-(19) is solved in the time interval $\left[t_0^*,\ t_f^*\right]=\left[0,\ 50\right]$ as an IVP in the explicit form:

$$\dot{\mathbf{q}}^* = f\left(t^*, \mathbf{q}^*\right) \tag{34}$$

where $\mathbf{q}^* = \left[y_1^*,\ \dot{y}_1^*,\ y_2^*,\ \dot{y}_2^*\right]^T$. The initial toe height is taken as half the spring free length, the body initial height as one and a half free lengths, while the hopper starts from rest. Then, the initial conditions in the vector form are the following:

$$\mathbf{q}_\mathbf{0}^* = \left[0.5,\ 0,\ 1.5,\ 0\right] \tag{35}$$

The system (34) is solved first for the nominal dimensionless parameter set $b^* = 1951$, $m_1^* = 1.88$, $k^* = 22.8$, $f^* = 0.051$, shown in bold in Table 2. The nominal set of parameters was chosen close to those presented in [37]. After the nominal parameters are set, (34) is solved again changing a parameter at a time, choosing from three equally spaced points in the intervals (24)-(27), resulting in twelve more combinations. A total of thirteen dimensionless parameter sets shown in Table 2 result in different impact conditions (i.e. oscillation frequency, impact damping, clearance) and numerical stiffnesses as shown by the value of *SR*.

Table 2. Stiffness ratio (*SR*) values for a range of dimensionless parameter values.

#	b^*	m_1^*	k^*	f^*	*SR*
1	1951	20.00	22.8	0.051	35
2	1951	10.40	22.8	0.051	172
3	1951	1.88	22.8	0.010	1328
4	550	1.88	22.8	0.051	3564
5	**1951**	**1.88**	**22.8**	**0.051**	**7010**
6	1951	1.88	22.8	0.155	17805
7	1951	0.80	22.8	0.051	31044
8	1951	1.88	22.8	0.300	36144
9	19275	1.88	22.8	0.051	50072
10	1951	1.88	240.5	0.051	72942
11	38000	1.88	22.8	0.051	176750
12	1951	1.88	480.0	0.051	181160
13	1951	1.88	1.0	0.051	234440

The system of equations (34), for most of the thirteen dimensionless parameter sets, seems to be fairly stiff during the stance phase, as for most cases in Table 2, the *SR* is of the order of $10^3 - 10^5$. At this point note that the flight phase eigenvalues are not affected by the dimensionless parameters, i.e. hybrid systems, represent a class of systems that are stiff during the stance phase and nonstiff during the flight phase. For such a class of systems, which type of solvers (stiff or nonstiff) performs better is an open question.

The method described here and the obtained results, provide interesting answers to this question. To this end, the seven ODE solvers in Table 1 are used to provide solutions, in three error tolerance levels (1e-3), (1e-6), (1e-9), and for two transition strategies, one with a simple "if…else…end" statement and the other using Matlab's event function. In this way, 14 experiments are conducted for each of the three tolerance levels and for each of the 13 parameter sets, resulting in a total of 546 experiments.

The solutions are evaluated in speed and accuracy using the *SD, DT* and *IAE* criteria. The best performance of all solvers, in all parameter sets and transition strategies is awarded with 10 points and the worst with 1 point. Intermediate performances are rewarded with points obtained using a linear interpolation in the interval [1, 10]. As a result for every transition strategy, tolerance level and criterion, if a solver has the worst performance for all of 13 parameter sets, it will accumulate the total of 1x13 points. On the other hand, if it has the best performance for all of 13 parameter sets, it will accumulate 10x13=130 points. With this point system, only results of the same error tolerance are compared. The total points each ODE solver obtains for every criterion and error tolerance level, and for all thirteen dimensionless parameter sets, were collected and displayed in two tables, one for the "if…else…end" transition strategy (Table 3) and the other for the event function transition strategy (Table 4). The three solvers that accumulated most points for every tolerance and criterion, were ranked 1 to 3, with 1 being the best.

Table 3. ODE solver point ranking with respect to *SD, DT, IAE* for various tolerances and phase transition by an *"if…else…end" statement.*

Error Tolerance (1e-3)						
	SD		*DT*		*IAE*	
Criteria / *Solvers*	*Points*	*Rank*	*Points*	*Rank*	*Points*	*Rank*
ode15s	102.77	-	42.30	-	102.31	-
ode45	**123.96**	**2**	**75.90**	**1**	**121.71**	**1**
ode23s	96.16	-	46.49	-	115.66	-
ode113	**119.37**	**3**	**55.94**	**2**	**116.15**	**3**
ode23tb	106.99	-	**47.91**	**3**	**117.59**	**2**
ode23t	105.41	-	47.00	-	111.62	-
ode23	**125.44**	**1**	43.84	-	115.04	-

Error Tolerance (1e-6)						
	SD		*DT*		*IAE*	
Criteria / *Solvers*	*Points*	*Rank*	*Points*	*Rank*	*Points*	*Rank*
ode15s	120.84	-	**99.63**	**1**	**122.79**	**2**
ode45	**127.78**	**1**	**36.46**	**3**	120.60	-
ode23s	84.16	-	18.90	-	119.39	-
ode113	**127.07**	**2**	27.50	-	**122.66**	**3**
ode23tb	106.98	-	18.94	-	121.07	-
ode23t	104.77	-	**79.06**	**2**	119.96	-
ode23	**124.22**	**3**	15.78	-	**122.92**	**1**

Error Tolerance (1e-9)						
Criteria / *Solvers*	*SD*		*DT*		*IAE*	
	Points	*Rank*	*Points*	*Rank*	*Points*	*Rank*
ode15s	**127.21**	**3**	15.02	-	113.75	-
ode45	**128.99**	**2**	**87.01**	**1**	**127.55**	**1**
ode23s	83.44	-	**36.94**	**2**	**119.54**	**3**
ode113	**129.19**	**1**	**28.02**	**3**	**121.88**	**2**
ode23tb	109.19	-	15.26	-	102.74	-
ode23t	107.39	-	19.97	-	116.67	-
ode23	123.37	-	23.61	-	118.61	-

Table 4. ODE solver point ranking with respect to *SD, DT, IAE* for various tolerances and phase transition by an *event function*.

Error Tolerance (1e-3)						
Criteria / *Solvers*	*SD*		*DT*		*IAE*	
	Points	*Rank*	*Points*	*Rank*	*Points*	*Rank*
ode15s	87.50	-	45.85	-	100.76	-
ode45	**112.91**	**2**	**74.44**	**1**	**122.23**	**1**
ode23s	85.26	-	46.31	-	**120.02**	**2**
ode113	**103.12**	**3**	**67.55**	**2**	**119.85**	**3**
ode23tb	98.19	-	**47.87**	**3**	118.86	-
ode23t	93.09	-	45.52	-	106.04	-
ode23	**115.76**	**1**	43.49	-	119.84	-

Error Tolerance (1e-6)						
Criteria / *Solvers*	*SD*		*DT*		*IAE*	
	Points	*Rank*	*Points*	*Rank*	*Points*	*Rank*
ode15s	116.37	-	**98.45**	**1**	**123.16**	**3**
ode45	**125.51**	**1**	**53.29**	**3**	**124.23**	**2**
ode23s	76.14	-	18.90	-	119.61	-
ode113	**123.70**	**2**	27.50	-	**128.16**	**1**
ode23tb	100.29	-	18.85	-	120.84	-
ode23t	95.42	-	**76.44**	**2**	120.07	-
ode23	**119.12**	**3**	15.83	-	122.61	-

Error Tolerance (1e-9)						
Criteria / *Solvers*	*SD*		*DT*		*IAE*	
	Points	*Rank*	*Points*	*Rank*	*Points*	*Rank*
ode15s	**126.23**	**3**	15.02	-	110.67	-
ode45	**128.39**	**2**	**112.28**	**1**	**128.77**	**1**
ode23s	75.89	-	**36.91**	**3**	**119.41**	**3**
ode113	**128.96**	**1**	**44.91**	**2**	**126.56**	**2**
ode23tb	102.33	-	15.26	-	102.23	-
ode23t	98.66	-	19.97	-	116.53	-
ode23	119.39	-	23.58	-	118.65	-

VI. DISCUSSION

From the results of Table 3 and Table 4, general guidelines can be extracted for application to the class of hybrid problems with short-duration impacts and long-duration flight phases, as that of a hopping monopod.

Looking at Table 3 and Table 4, to our surprise, ode45 and ode113 were among the best three performing ODE solvers most of the time. The ode15s performed distinctively well mainly for error tolerance (1e-6), while ode23s showed good results two times using the "if…else…end" transition strategy, and three times for an event function based transition strategy. The ode23 showed that could provide fast solutions for error tolerance (1e-3) to (1e-6), but was in most cases overruled in accuracy by other solvers. The ode23tb mainly fared well for crude error tolerance (1e-3), while ode23t only once per table with respect to the *DT* criterion. Therefore, although in many simulations of legged robots the preference is for stiff solvers to address hybrid problems with impacts, such as the ode15s or the ode23s, [40], [41], this preference does not seem to be justified based on the obtained results.

Comparing the results from Tables 3 and 4, in almost all cases using the built-in event function, the *SD* of all solvers deteriorated. For instance, for error tolerance (1e-3) ode15s scored 102.77 points for *SD* in the "if…else…end" case (Table 3), while it only scored 87.50 points for *SD* in the event function case, for the same error tolerance (Table 4). The same observation applies for the performance of all solvers in *SD*, comparing results for the same error tolerance, and for both phase transition strategies. The performance of ode45 and ode113 with respect to the *DT* and *IAE* criteria was a little higher in comparison to the "if…else…end" case; the performance of all other solvers was mixed with respect to these criteria.

Based on the above observations, the guideline that emerges is that if someone is simply interested in obtaining the response of a similar hybrid system with impacts, the ode45 or the ode113 should be tried first, with the ode15s and ode23s to be considered next. In the case that the event function is considered for phase transition, it is pointed out that the performance of the ODE solver used will deteriorate in solution speed, without necessarily improving in accuracy.

Using Table 3 and Table 4, case-specific guidelines can be also proposed. For instance, assume one is interested in obtaining the response of a hybrid system with impacts, with phase transition using an "if…else…end" statement, and error tolerance (1e-6). Assume also that *SD* is twice as important as *IAE*, while *DT* is unimportant. In such a case, one may use the results provided in Table 3 under the legend *Error tolerance (1e-6)* to calculate a weighted sum of total points p_{tot} for every solver, defined by:

$$p_{tot} = w_{SD}\, p_{SD} + w_{DT}\, p_{DT} + w_{IAE}\, p_{IAE} \tag{36}$$

where p_{SD}, p_{DT} and p_{IAE} are the points obtained and w_{SD}, w_{DT} and w_{IAE} the weighting factors of the ODE solver in the corresponding to the subscript criteria. For the example case study, the weighting factors are assigned values $w_{SD} = 2$, $w_{DT} = 0$ and $w_{IAE} = 1$. Using (36), the calculated weighted sum is shown in Table 5. This table also indicates that the best ODE solver for this case is ode113, a solver recommended for nonstiff problems.

Table 5. Ranking of ODE solvers, for error tolerance (1e-6), transition by an "if…else…end" statement, and weighting factors $w_{SD} = 2$, $w_{DT} = 0$, and $w_{IAE} = 1$.

Solver	p_{tot}	*Rank*
ode15s	364.47	-
ode45	376.16	2
ode23s	287.71	-
ode113	**376.80**	**1**
ode23tb	335.03	-
ode23t	329.50	-
ode23	371.36	3

VII. CONCLUSIONS

Hybrid systems represent a class of problems that cycle between phases when the system EoM is stiff (interaction with the ground) and phases when it is nonstiff (flight phases). As the question of selecting the best solver for such a system was open, in this paper a method was proposed to provide guidelines for selecting an ODE solver and its parameters for such systems. A monopod hopper interacting compliantly with the ground was introduced as a new benchmark problem, and used to compare the solvers available in the widely used Matlab ODE Suite, according to three criteria for solution speed, and accuracy. To provide generality to the results, the mathematical description of the model was brought to a dimensionless form, and its dimensionless parameters were varied in a range taken from existing systems and corresponding to different levels of numerical stiffness. The effects of error tolerance and phase transition strategy were also studied. Finally, guidelines were provided, for selecting the appropriate ODE solver, both overall and case-specific. Interestingly, the best solver for a realistic case turned out to be a solver recommended for numerically nonstiff problems.

ACKNOWLEDGMENT

Funding for this research by the "IKY Fellowships of Excellence for Postgraduate Studies in Greece – Siemens Programme" in the framework of the Hellenic Republic – Siemens Settlement Agreement is acknowledged.

REFERENCES

[1] Park, H. W., Wensing, P. M., & Kim, S., 2017, "High-speed bounding with the MIT Cheetah 2: Control design and experiments." *The International Journal of Robotics Research*, 0278364917694244.

[2] Paraskevas, I., and Papadopoulos, E., 2016, "Parametric sensitivity and control of on-orbit manipulators during impacts using the Centre of Percussion concept." *Control Engineering Practice* 47: 48-59.

[3] Vasilopoulos, V., Paraskevas, I. and Papadopoulos, E., 2014, "All-terrain Legged Locomotion Using a Terradynamics Approach," in *2014 International Conference on Intelligent Robots and Systems (IROS '14)*, Chicago, Illinois, Sept. 14–18.

[4] Blum, Y., Lipfert, S. W., Rummel, J. and Seyfarth, A., 2010, "Swing leg control in human running," *Bioinspiration & biomimetics*, **5**(2).

[5] Koutsoukis, K. and Papadopoulos, E., 2016, "On Passive Quadrupedal Bounding with Translational Spinal Joint," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '16)*, Deajeon, Korea, October 9-14.

[6] Byrne, G. D. and Hindmarsh, A. C., 1987, "Stiff ODE solvers: A review of current and coming attractions," *Journal of Computational Physics*, **70**(1), pp. 1-62.

[7] Petcu, D., 2004, "Software issues in solving initial value problems for ordinary differential equations," *Creative Math*, **13**, pp. 97-110.

[8] Hull, T. E., Enright, B. M. and Sedgwick, A. E., 1972, "Comparing numerical methods for ordinary differential equations," *SIAM Journal on Numerical Analysis*, **9**(4), pp. 603-637.

[9] Enright, H. W., Hull, T. E. and Lindberg, B., 1975, "Comparing numerical methods for stiff systems of ODE's," *BIT Numerical Mathematics*, **15**(1), pp. 10-48.

[10] Krogh, F. T., 1973, "On testing a subroutine for the numerical integration of ordinary differential equations," *Journal of the ACM (JACM)*, **20**(4), pp. 545-562.

[11] Enright, W. H. and Pryce, J. D., 1987, "Two FORTRAN packages for assessing initial value methods," *ACM Transactions on Mathematical Software (TOMS)*, **13**(1), pp. 1-27

[12] Shampine, L. F., 1981, "Evaluation of a test set for stiff ODE solvers," *ACM Transactions on Mathematical Software (TOMS)*, **7**(4), pp. 409-420.

[13] Nowak, U. and Gebauer, S. , 1997, "A new test frame for ordinary differential equation solvers," *ZIB*.

[14] Mazzia, F., Iavernaro, F. and Magherini, C., 2008, "Test set for initial value problem solvers," Department of Mathematics, University of Bari.

[15] https://cran.r-project.org/web/packages/deTestSet/index.html.

[16] https://github.com/mauro3/IVPTestSuite.jl.

[17] Auer, E. and Rauh, A. , 2012, "VERICOMP: a system to compare and assess verified IVP solvers," *Computing*, **94**(2-4), pp. 163-172.

[18] https://www.mathworks.com/help/matlab/math/choose-an-ode-solver.html.

[19] Gattwald, B. A. and Wanner, G., 1982, "Comparison of numerical methods for stiff differential equations in biology and chemistry," *Simulation*, **38**(2), pp. 61-66.

[20] Radhakrishnan, K., 1984, "Comparison of numerical techniques for integration of stiff ordinary differential equations arising in combustion chemistry," NASA Technical Paper 2372.

[21] Sandu, A., Verwer, J. G., Van Loon, M., Carmichael, G. R., Potra, F. A., Dabdub, D. and Seinfeld, J. H., 1997, "Benchmarking stiff ode solvers for atmospheric chemistry problems-I. implicit vs explicit," *Atmospheric Environment*, **31**(19), pp. 3151-3166

[22] Nejad, L. A., 2005, "A comparison of stiff ODE solvers for astrochemical kinetics problems," *Astrophysics and Space Science*, **299**(1), pp. 1-29.

[23] Petzold, L., 1983, "Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations," *SIAM journal on scientific and statistical computing*, **4**(1), pp. 136-148.

[24] Shampine, L. F., 1977, "Stiffness and nonstiff differential equation solvers, II: detecting stiffness with Runge-Kutta methods," *ACM Transactions on Mathematical Software (TOMS)*, **3**(1), pp. 44-53.

[25] Felgner, F., Liu, L., Frey, G., 2011, "Vergleich numerischer Löser zur Simulation steifer und hybrider Systeme.", *Proceedings of the Kongress Automation*, ISBN 978-3-18-092143-3, VDI-Berichte 2143, pp. 303-306, Baden-Baden, Germany.

[26] Abelman S. and Patidar, K. C., 2008, "Comparison of some recent numerical methods for initial-value problems for stiff ordinary differential equations," *Computers & Mathematics with Applications*, **55**(4), pp. 733-744.

[27] Shampine, L. and Thompson, S., 2000, "Event location for ordinary differential equations," *Computers & Mathematics with Applications*, **39**(5-6), pp. 43-54.

[28] Shampine, L. and Reichelt, M., 1997, "The MATLAB ODE Suite," *SIAM Journal on Scientific Computing*, **18**(1), pp. 1-22.

[29] Shampine, L. F., Gladwell, I. and Thompson, S., 2003, "Solving ODEs with matlab," Cambridge University Press.

[30] Ashino, R., Nagase, M. and Vaillancourt, R., 2000, "Computers & Mathematics with Applications," *Behind and beyond the MATLAB ODE suite*, **40**(4), pp. 491-512.

[31] Gilardi, G. and Sharf, I., 2002, "Literature survey of contact dynamics modeling," *Mechanism and Machine Theory*, **37**(10), pp. 1213-1239. DOI: 10.1016/S0094-114X(02)00045-9

[32] Buckingham, E., 1914, "On physically similar systems; illustrations of the use of dimensional equations," *Physical Review*, **4**(4), pp. 345-376. DOI: 10.1103/PhysRev.4.345

[33] Cherouvim, N. and Papadopoulos, E., 2008, "The SAHR Robot - Controlling Hopping Speed and Height Using a Single Actuator," *Applied Bionics and Biomechanics*, **5**(3), pp. 149-156. DOI: 10.1080/11762320802564218

[34] Ahmadi, M. and Buehler, M., 2006, "Controlled Passive Dynamic Running Experiments With the ARL-Monopod II," *IEEE Transaction on robotics*, **22**(5), pp. 974-986. DOI: 10.1109/TRO.2006.878935

[35] Raibert, M.,1986, "Legged Robots that Balance," The MIT Press, pp. 145-146.
[36] Okubo, H., Nakano, E. and Handa, M., 1996, "Design of a Jumping Machine Using Self-energizing Spring," in *IROS*. DOI: 10.1109/IROS.1996.570659
[37] Vasilopoulos, V. and Paraskevas, I., Papadopoulos E., 2015, "Control and Energy Considerations for a Hopping Monopod on Rough Compliant Terrains," in *ICRA*, Washington. DOI: 10.1109/ICRA.2015.7139832
[38] Spijker, M., 1996, "Stiffness in numerical initial-value problems," *Journal of Computational and Applied Mathematics*, **72**(2), pp. 393-406. DOI: 10.1016/0377-0427(96)00009-X
[39] https://www.mathworks.com/help/matlab/math/ode-event-location.html.
[40] Machairas, K., and Papadopoulos, E., 2016, "An Active Compliance Controller for Quadruped Trotting," in *MED*, Athens.
[41] Saha, S., Fiorini, P. and Shah, S., 2006, "LANDING MECHANISMS FOR HOPPING ROBOTS: CONSIDERATIONS AND PROSPECTS," in *ASTRA*, Noordwijk.
[42] Khulief Y.A., 2012, "Modeling of Impact in Multibody Systems: An Overview.", ASME. *J. Comput. Nonlinear Dynam*, 8(2) : 021012 - 021012 - 15. DOI: 10.1115/1.4006202
[43] Hunt K. H., Crossley F. E., 1975, "Coefficient of Restitution Interpreted as Damping in Vibroimpact.", ASME. *J. Appl. Mech*., **42**(2), pp. 440-445. DOI:10.1115/1.3423596.
[44] Papetti S., Avanzini F. and Rocchesso D., 2011, "Numerical Methods for a Nonlinear Impact Model: A Comparative Study With Closed-Form Corrections," in IEEE Transactions on Audio, Speech, and Language Processing, **19**(7), pp. 2146-2158, DOI: 10.1109/TASL.2011.2118204